CAR-NF: A classifier based on specific rules with high netconf

Article *in* Intelligent Data Analysis · January 2012

DOI: 10.3233/IDA-2011-0510

CAR-NF: A Classifier based on Specific Rules with High Netconf

R. Hernández-León[1,2,*] Jesús A. Carrasco-Ochoa[1]
José Fco. Martínez-Trinidad[1]
J. Hernández-Palancar[2]

[1]Computer Science Department
National Institute of Astrophysics, Optics and Electronics
Luis Enrique Erro ♯1, Sta. María Tonantzintla, Puebla, CP: 72840, Mexico.
{raudel,ariel,fmartine}@inaoep.mx
[2]Advanced Technologies Application Center
7a ♯21812, Siboney, Playa, CP: 12200, Havana, Cuba.
{rhernandez,jpalancar}@cenatav.co.cu

October 18, 2010

Abstract

In this paper, an accurate classifier based on Class Association Rules (CARs), called CAR-NF, is proposed. CAR-NF introduces a new strategy for computing CARs, using the Netconf as measure of interest, that allows to prune the CAR search space for building specific rules with high Netconf. Moreover, we propose and prove a proposition that supports the use of a Netconf threshold value equal to 0.5 for mining the CARs. Additionally, a new way for ordering the set of CARs based on their rule sizes and Netconf values is introduced in CAR-NF. The ordering strategy together with the "Best K rules" satisfaction mechanism allows CAR-NF to have better accuracy than CBA, CMAR, CPAR, TFPC and HARMONY classifiers, the best classifiers based on CARs reported in the literature.

*Corresponding author

Key words: Data mining, Supervised Classification, Class Association Rules, Association Rule Mining

1 Introduction

The Classification Association Rule Mining (CARM) or associative classification, introduced in [22], integrates Classification Rule Mining (CRM) [7, 25] and Association Rule Mining (ARM) [1, 19]. This integration involves mining a special subset of association rules, called Class Association Rules (CARs).

Associative classification aims to mine a set of CARs from a class-transaction dataset; where a CAR describes an implicative co-occurring relationship between a set of items (itemset) and a class, expressed as "$\langle item_1, \ldots, item_n \rangle \Rightarrow class$". A classifier based on this approach usually consists of an ordered CAR list l, and a mechanism for classifying unseen transactions using l.

Associative classification has been used in different tasks, for example: text classification [9, 39], determination of DNA splice junction types [6], text segmentation [10], automatic image annotation [29], mammalian mesenchymal stem cell differentiation [34] and prediction of protein-protein interaction types [23], among others.

Currently, all classifiers based on CARs use the Support and Confidence measures for computing and ordering the set of CARs [15, 21, 22, 30, 31, 38]. In CARM, similar to ARM, it is assumed that a set of items $I = \{i_1, i_2, \ldots, i_n\}$, a set of classes C, and a set of transactions D are given, where each transaction $t \in D$ consists of an itemset and a class. The Support of an itemset $X \subseteq I$ (denoted as $Sup(X)$) is the fraction of transactions in D containing X (see Eq. 1). A CAR is an implication of the form $X \Rightarrow c$ where $X \subseteq I$ and $c \in C$. A CAR with k items (including the class *i.e.* $|X| = k-1$) will be called a k-CAR. The rule $X \Rightarrow c$ is held in D with certain Support s and Confidence α, where s

(see Eq. 2) is the fraction of transactions in D that contains $X \cup \{c\}$, and α (see Eq. 3) is the probability of finding c in transactions that also contain X, which represents how "strongly" the rule antecedent X implies the rule consequent c. A CAR $X \Rightarrow c$ satisfies or covers a transaction t if $X \subseteq t$.

$$Sup(X) = \frac{|D_X|}{|D|} \tag{1}$$

where D_X is the set of transactions in D containing X.

$$Sup(X \Rightarrow c) = Sup(X \cup \{c\}) \tag{2}$$

$$Conf(X \Rightarrow c) = \frac{Sup(X \Rightarrow c)}{Sup(X)} \tag{3}$$

Many studies [1, 19, 40] have pointed out the combinatorial number of association rules that could be obtained when a small Support threshold is used. To address this problem, recent works [15, 35, 36, 37] prune the CAR search space stopping the growth of the rule when a CAR satisfies the Support and Confidence thresholds, it means that CARs satisfying both thresholds are not extended anymore leading to obtain general (small) rules. This strategy has some drawbacks, for example:

- Many branches of the CAR search space could be explored in vain because the CAR search space is not pruned when a CAR satisfies only the Support threshold, instead the CAR is extended until it also satisfies the Confidence threshold, which could never happen.

- If a CAR $X \Rightarrow c$ is obtained then a CAR $X^{'} \Rightarrow c$ with $X \subset X^{'}$ can not be obtained, it does not allow to generate specific (large) rules, some of which could be more interesting (*i.e.* "with higher Confidence").

In order to overcome these drawbacks, in this paper we introduce the use of Netconf measure instead of support and confidence for computing the rules with a new pruning strategy.

Although previous works [22, 36] have shown that associative classification seems to achieve better accuracy than other classification approaches such as decision trees, rule induction and probabilistic methods, associative classification still has some weaknesses that must be addressed, for example:

- The Confidence measure detects neither statistical independence nor negative dependence among items, because Confidence does not take into account the Support of the consequent [5].
- Previous CAR pruning strategies do not allow to generate specific (large) rules, some of which could be more interesting than general (small) rules.
- Threshold values used to compute the set of CARs are not supported.

The main contribution of this paper is an accurate classifier, called CAR-NF, which has better performance than the best classifiers based on CARs reported in the literature CBA, CMAR, CPAR, TFPC and HARMONY classifiers, all of them following the Support-Confidence framework. CAR-NF introduces a new pruning strategy to generate the set of CARs, which allows to find specific rules with high Netconf. Additionally, CAR-NF introduces a new way for ordering the set of CARs, based on the size of the CARs and their Netconf value. Moreover, we determine an appropriate Netconf threshold value, supported by a proposition, that avoids ambiguity at the classification stage.

This paper is organized as follows: The next section describes the related work. The third section introduces our classifier. In the fourth section the experimental results are shown. Finally the conclusions and future works are given in section five.

2 Related work

Classification Association Rule Mining was first presented in [3] where it was used for two specific tasks: reducing telecommunication order failures and detecting redundant medical tests. Later, several classifiers based on CARs have been developed. In general, these classifiers can be divided in two groups according to the strategy used for computing the set of CARs:

1. Two Stage classifiers. In a first stage all the CARs satisfying predefined Support and Confidence thresholds are mined and later, in a second stage, a classifier is built by selecting a subset of CARs. The second stage involves a coverage analysis where each CAR is examined in order to find a small set of CARs that fully covers the training set. Some classifiers following this approach are CBA (Classification Based on Associations) [22], CMAR (Classification based on Multiple Association Rules) [21] and MCAR (Multi-Class Classification based on Association Rule) [31]. The CBA classifier uses an apriori-like based algorithm for CAR generation [1]. CMAR and MCAR are similar to CBA but for generating the set of CARs they use FP-growth [17] and Eclat [40] algorithms respectively.

2. Integrated classifiers. In these classifiers the subset of CARs is generated during the classification stage. Some classifiers following this approach are: TFPC (Total From Partial Classification) [14, 15, 37], HARMONY (Highest confidence clAssification Rule Mining fOr iNstance-centric classifYing) [32]; and induction systems such as FOIL (First Order Inductive Learner) [26], PRM (Predictive Rule Mining) and CPAR (Classification based on Predictive Association Rules) [38].

Once a subset of CARs has been generated, regardless of the used strategy, the CARs are ordered. There are five main ordering schemes reported in the literature:

a) CSA (Confidence - Support - Antecedent size): The CSA ordering scheme combines Confidence, Support and the size of the rule antecedent. CSA sorts the rules in a descending order according to the Confidence. Those CARs that share a common Confidence value are sorted in a descending order according to the Support, and in case of tie, CSA sorts the rules in ascending order according to the size of the rule antecedent [21, 22].

b) ACS (Antecedent size - Confidence - Support): The ACS ordering scheme is a variation of CSA. But it takes into account the size of the rule antecedent as first ordering criterion followed by the Confidence and the Support [14].

c) WRA (Weighted Relative Accuracy): The WRA ordering scheme, proposed in [20], assigns to each CAR a weight (based on Support and Confidence) and then sorts the set of CARs in a descending order according to the assigned weights. The WRA has been used to order lists of CARs in several CAR classifiers [14, 36, 37]. Given a rule $A \Rightarrow B$ the WRA is computed as follows:

$$WRA(A \Rightarrow B) = Sup(A)(Conf(A \Rightarrow B) - Sup(B))$$

d) LAP (Laplace Expected Error Estimate): The LAP ordering scheme was introduced by Clark and Boswell [12] and it has been used to order the list of CARs in some CAR classifiers [36, 38]. Given a rule $A \Rightarrow B$, in [38] the LAP is defined as follows:

$$LAP(A \Rightarrow B) = \frac{Sup(A \Rightarrow B) + 1}{Sup(A) + \mid C \mid}$$

where C is the set of predefined classes.

e) χ^2 (Chi-Square): The χ^2 ordering scheme is a well known technique in statistics, which can be used to determine whether two variables are independent or related. After computing an additive χ^2 value for each CAR (also based on Support and Confidence), this value is used to sort the set of CARs in a descending order [21].

All these ordering schemes take into account the Confidence measure. But, as we have mentioned in the introduction, this measure has some drawbacks. We will come back to this point in the next subsection.

Once a classifier has been built, usually presented as a list of sorted rules, there are three main satisfaction (or covering) mechanisms for classifying unseen data.

1. **Best rule**: This mechanism selects the first ("best") rule in the order that satisfies the transaction to be classified (unseen data), and then the class associated to the selected rule is assigned to this transaction [22].

2. **Best K rules**: This mechanism selects the best K rules (for each class) that satisfy the transaction to be classified and then the class of the new transaction is determined using these K rules, according to different criteria [36].

3. **All rules**: This mechanism selects all rules that satisfy the transaction to be classified and then these rules are used to determine the class of the new transaction [21].

Algorithms following the "Best rule" mechanism could suffer biased classification or overfitting since the classification is based on only one rule. On the other hand, the "All rules" mechanism includes rules with low ranking for classification and this could affect the accuracy of the classifier. Since the "Best K rules" mechanism has been the most widely used for CAR classifiers, reporting the best results, we will use it in our work.

2.1 Drawbacks of the Confidence measure

As we mentioned above, all the classifiers developed for CARM use the Confidence measure for mining the set of CARs. However, several authors have pointed out some drawbacks of this measure that could lead us to discover many more rules than it should [5, 8, 27, 28]. In particular, the presence of items with high Support can lead us to obtain misleading rules (see Example 1) because higher-Support items appear in many transactions and they could be predicted by any itemset [5].

Example 1 Let's assume that $Sup(X) = 0.5$, $Sup(Y) = 0.7$, $Sup(X \Rightarrow Y) = 0.3$ and the Confidence threshold is set to 0.5. By Eq. 3, $Conf(X \Rightarrow Y) = 0.6$. We are tempted to choose $X \Rightarrow Y$ as an interesting rule, but there is a problem. Y occurs in 70% of the transactions, but as the rule only has 60% of Confidence it does worse than just randomly guessing. In this case, $X \Rightarrow Y$ is a misleading rule.

This is a key weakness of the Confidence measure, and it is particularly evident in Census data, where many items are very likely to occur with or without other items (e.g. the Census dataset employed in [8], where $|I| = 2166$ and there are many items with Support above 95%).

In [24], the authors defined a good accuracy measure (ACC), as a measure that separates strong rules from weak rules, assigning them high and low values respectively. Additionally, the authors suggested several desirable properties that a good ACC should satisfy. These properties are the following:

Property 1 *If* $Sup(A \Rightarrow B) = Sup(A)Sup(B)$ *then* $ACC(A \Rightarrow B) = 0$

This property claims that any good accuracy measure must test the independence [5].

Property 2 $ACC(A \Rightarrow B)$ *monotonically increases with* $Sup(A \Rightarrow B)$ *when all other parameters remain the same.*

The property 2 can be interpreted as follows: Suppose a dataset D and two rules $A \Rightarrow B$ and $A^{'} \Rightarrow B^{'}$ such that $Sup(A) = Sup(A^{'})$ and $Sup(B) = Sup(B^{'})$. If the fraction of transactions in D that contains $A \cup B$ ($Sup(A \Rightarrow B)$) is greater than the fraction of transactions in D that contains $A^{'} \cup B^{'}$ ($Sup(A^{'} \Rightarrow B^{'})$) then $ACC(A \Rightarrow B) > ACC(A^{'} \Rightarrow B^{'})$ (which means that $A \Rightarrow B$ is stronger than $A^{'} \Rightarrow B^{'}$).

Property 3 $ACC(A \Rightarrow B)$ *monotonically decreases when* $Sup(A)$ *(or* $Sup(B)$*) increases and all other parameters remain the same.*

An ACC satisfying property 3 avoids to obtain misleading rules because its value does not increase by only increasing the consequent (or antecedent) Support.

An ACC satisfying properties 2 and 3 has local maxima when $Sup(A \Rightarrow B) = Sup(A)$ or $Sup(A \Rightarrow B) = Sup(B)$ and it has a global maximum when $Sup(A \Rightarrow B) = Sup(A) = Sup(B)$.

Now we will show that $Conf(A \Rightarrow B)$ (see Eq. 4), which has been used in all the algorithms for CARM, does not satisfy simultaneously all these properties:

$$Conf(A \Rightarrow B) = \frac{Sup(A \Rightarrow B)}{Sup(A)} \tag{4}$$

Proposition 1 $Conf(A \Rightarrow B)$ *does not satisfy the property 1.*

Proof. Here is a counterexample: Consider the transactional dataset shown in table 1(a), where rows represent transactions and columns represent items.

Table 1(b) shows the Supports of the itemsets $\{i_1\}$, $\{i_2\}$ and $\{i_1, i_2\}$. Since $Sup(\{i_1\})Sup(\{i_2\}) = 0.25 = Sup(\{i_1, i_2\})$, $\{i_1\}$ and $\{i_2\}$ are statistically independent and hence the Confidence of $\{i_1\} \Rightarrow \{i_2\}$ should be 0. However, $Conf(\{i_1\} \Rightarrow \{i_2\}) = 0.25/0.5 = 0.5 \neq 0$ □

Proposition 2 *$Conf(A \Rightarrow B)$ satisfies the property 2*

Proof. Trivial according to Eq. (4). □

Proposition 3 *$Conf(A \Rightarrow B)$ satisfies the property 3 for $Sup(A)$*

Proof. Trivial according to Eq. (4). □

Proposition 4 *$Conf(A \Rightarrow B)$ does not satisfy the property 3 for $Sup(B)$.*

Proof. Since $Sup(B)$ does not appear in Eq. (4) then the property 3 is not satisfied for $Sup(B)$. □

In summary, the Confidence measure detects neither statistical independence (property 1) nor negative dependence between items because the Support of the consequent is not considered in its definition. Thus we can see that, according [24], the Confidence measure is not a good accuracy measure for separating strong rules from weak rules.

In [5] the authors analyzed several measures (Conviction, Interest or Lift, χ^2 and Certainty factor), as an alternative to the Confidence measure, for estimating the strength of an association rule. Some of these measures overcome the drawbacks of the Confidence measure but only Interest and Certainty factor fulfill the properties $1 - 3$ suggested in [24]. However, both Interest and Certainty factor have other limitations.

The Interest measure has a not bounded range [5], therefore differences among its values are not meaningful and for this reason, it is difficult to define an Interest threshold. Moreover, the Interest measure is symmetric (Eq. 5) but this almost never happens in practice.

$$Int(A \Rightarrow B) = \frac{Sup(A \Rightarrow B)}{Sup(A)Sup(B)} = \frac{Sup(A \cup B)}{Sup(A)Sup(B)} = \frac{Sup(B \Rightarrow A)}{Sup(B)Sup(A)} = Int(B \Rightarrow A) \quad (5)$$

On the other hand, Certainty factor is defined by Eq. 6.

$$CF(A \Rightarrow B) = \begin{cases} \frac{Conf(A \Rightarrow B) - Sup(B)}{1 - Sup(B)} & if\ Conf(A \Rightarrow B) > Sup(B) \\ \frac{Conf(A \Rightarrow B) - Sup(B)}{Sup(B)} & if\ Conf(A \Rightarrow B) < Sup(B) \\ 0 & otherwise \end{cases} \quad (6)$$

Negative values of Certainty factor mean negative dependence, while positive values mean positive dependence and 0 means independence. However, the value that Certainty factor takes depends on the Support of the consequent (the class in our case). When $Conf(A \Rightarrow B)$ is close to $Sup(B)$, even if the difference of $Conf(A \Rightarrow B)$ and $Sup(B)$ is close to 0 but still positive, the Certainty factor measure shows a strong positive dependence when $Sup(B)$ is high (close to 1). For better understanding let's see the following example taken from [2]:

Example 2 Suppose that $Sup(A) = 0.5$ and $Sup(B) = 0.9$. If $Sup(A \Rightarrow B) = 0.45$ then A and B are independent according Certainty Factor, since:

$$Conf(A \Rightarrow B) = \frac{Sup(A \Rightarrow B)}{Sup(A)} = \frac{0.45}{0.5} = \frac{0.5 * 0.9}{0.5} = 0.9 = Sup(B)$$

$$\therefore CF(A \Rightarrow B) = 0$$

If $Sup(A \Rightarrow B) = 0.43$, the Certainty factor of $A \Rightarrow B$ is -0.044 by equation 6. This means that there is a slightly negative relationship between A and B. But if $Sup(A \Rightarrow B) = 0.47$, the Certainty factor of $A \Rightarrow B$ is 0.4 by equation

6. This shows that A and B are positively dependent. The difference between 0.43 and 0.45 is equal to the difference between 0.45 and 0.47. However, the Certainty factor obtains very different results.

In [2], the authors introduced a measure to estimate the strength of an association rule, called Netconf. This measure, defined in equation 7, has among its main advantages that it detects misleading rules produced by the Confidence. As a simple example, suppose that $Sup(X) = 0.4$, $Sup(Y) = 0.8$ and $Sup(X \Rightarrow Y) = 0.3$, therefore $Sup(\neg X) = 1 - Sup(X) = 0.6$ and $Sup(\neg X \Rightarrow Y) = Sup(Y) - Sup(X \Rightarrow Y) = 0.5$ (see table 2). If we compute $Conf(X \Rightarrow Y)$ we obtain 0.75 (a high Confidence value) but Y occurs in 80% of the transactions, therefore the rule $X \Rightarrow Y$ does worse than just randomly guessing, clearly, $X \Rightarrow Y$ is a misleading rule [5]. For this example, $Netconf(X \Rightarrow Y) = -0.083$ showing a negative dependence between the antecedent and the consequent. If we analyze the rule $\neg X \Rightarrow Y$ then $Conf(\neg X \Rightarrow Y) = 0.83 > 0.8 = Sup(Y)$, it means that rule $\neg X \Rightarrow Y$ does better than just randomly guessing. However, the Netconf value for rule $\neg X \Rightarrow Y$ is 0.083 showing a positive dependence between the antecedent and the consequent.

$$Netconf(A \Rightarrow B) = \frac{Sup(A \Rightarrow B) - Sup(A)Sup(B)}{Sup(A)(1 - Sup(A))} \tag{7}$$

The authors in [2] showed that Netconf measure overcomes the drawbacks of all above mentioned measures but they did not prove that Netconf is a good ACC measure by proving that it fulfils properties 1-3, we will come back to this point further.

Since Netconf overcomes the drawbacks of all above mentioned measures but it was not used before in CAR based classification, it motivated the classifier based on CARs presented in the next section.

3 CAR-NF classifier

The explanation of our classifier is divided in three subsections. First of all, in subsection 3.1, we prove that the Netconf measure completely fulfills the properties 1-3 above mentioned. Also, we show that Netconf does not have the drawbacks of other measures. Additionally, we propose and prove a proposition that supports the use of a Netconf threshold value equal to 0.5 for mining the CARs. An important part of the CAR-NF classifier is the algorithm for mining the set of CARs, in subsection 3.2 we introduce this algorithm based on the efficient use of equivalence classes and bit-to-bit operations for computing the set of CARs employing the Netconf measure. Finally, in section 3.3, we propose a new way for ordering the set of CARs that together with the "Best K rules" mechanism defines the CAR-NF classifier.

3.1 Netconf measure

In [2], the authors introduced a measure to estimate the strength of an association rule, called Netconf. As mentioned in subsection 2.1, Netconf overcomes the drawbacks of Confidence, Conviction, Interest or Lift, χ^2 and Certainty factor. Additionally, the authors of [2] show that the Netconf measure has some useful properties, for example:

- Netconf tests independence, therefore $Netconf(A \Rightarrow B) = 0 \Leftrightarrow Sup(A \Rightarrow B) = Sup(A)Sup(B)$.
- $Netconf(A \Rightarrow B) \neq Netconf(B \Rightarrow A)$ for $Sup(A) \neq Sup(B)$, it means that Netconf is not symmetric and it can indicate the strength of implication in both directions, not only the degree of dependence.
- $Netconf(A \Rightarrow B)$ takes values in [-1,1].

- Positive values of the Netconf measure represent positive dependencies, negative values of Netconf represent negative dependencies and a zero value represents independence.

However, the authors did not prove that Netconf satisfies the properties suggested in [24].

According to Eq. (7) and taking into account that the Support takes values in $[0, 1]$, it is easy to see that Netconf satisfies the properties 1 and 2, and also satisfies the property 3 for $Sup(B)$.

In order to show that Netconf completely satisfies property 3 we will prove the next proposition.

Proposition 5 *Netconf satisfies the property 3 for $Sup(A)$.*

Proof. In Eq. (7), let $Sup(A \Rightarrow B) = S_{ab}$, $Sup(B) = S_b$, $Sup(A) = x$ be the Support of $A \Rightarrow B$, B and A respectively, with S_{ab} and S_b constants satisfying $0 < S_{ab} \leq S_b < 1$ and $x \in (0, 1)$. We can rewrite the right member of Eq. (7) in terms of S_{ab}, S_b, and x, as follows:

$$f(x) = \frac{S_{ab} - S_b x}{x(1 - x)}$$

Now, we will prove that $f'(x) < 0$, which implies that $f(x)$ is strictly decreasing and therefore proposition 5 is true. Computing the first derivative and reducing terms we have:

$$f'(x) = \frac{-S_b x^2 + 2S_{ab}x - S_{ab}}{x^2(1 - x)^2}$$

Due to $0 < S_{ab} \leq S_b < 1$ then:

$$-S_b x^2 + 2S_{ab}x - S_{ab} \leq -S_{ab}x^2 + 2S_{ab}x - S_{ab} = -S_{ab}(x - 1)^2 < 0,$$

and $x^2(1-x)^2 > 0$, therefore, $f'(x) < 0$. □

With the proof of proposition 5, we have showed that Netconf fulfills the properties $1-3$. In particular, the fact that Netconf satisfies the property 3 is the main motivation for using it in our CAR-NF classifier. As it was shown in subsection 2.1, an accuracy measure satisfying property 3 avoids to obtain misleading rules. Thus if we return to the example 1 of subsection 2.1 and we evaluate the Netconf measure for $Sup(X) = 0.5$, $Sup(Y) = 0.7$ and $Sup(X \Rightarrow Y) = 0.3$, we obtain a Netconf value equal to -0.2, meaning that there is a negative dependence between X and Y and consequently this is a bad rule. In our CAR-NF classifier, we bet for CARs with high Netconf values (positive dependence between X and Y), therefore, the misleading rules are avoided. In this paper we propose to use Netconf for computing and ordering the set of CARs.

Previous works use different Support and Confidence thresholds for mining the set of CARs. The threshold values used in those works must be carefully defined because a huge volume of CARs could be generated. However, those threshold values are not supported. In our case, we choose a Netconf threshold that allows to obtain CARs with different antecedent, avoiding ambiguity at classification stage. In order to determine the appropriate Netconf threshold we introduce and prove the proposition 6, which states that for any itemsex X, only one CAR with X as antecedent can have a Netconf value greater than 0.5.

Proposition 6 *Let $C = \{c_1, c_2, \ldots, c_m\}$ be the set of predefined classes, for each itemset X we can obtain at most one rule $X \Rightarrow c_k$ ($c_k \in C$) with Netconf value greater than* 0.5.

Proof. Let us assume that there are two CARs $X \Rightarrow c_{k_1}$ and $X \Rightarrow c_{k_2}$ with $c_{k_1}, c_{k_2} \in C$ such that:

$$Netconf(X \Rightarrow c_{k_1}) > 0.5$$
$$Netconf(X \Rightarrow c_{k_2}) > 0.5$$

then, adding these inequalities we obtain the following statement,

$$Netconf(X \Rightarrow c_{k_1}) + Netconf(X \Rightarrow c_{k_2}) > 1 \quad (8)$$

From Eqs. 1 and 2 defined in section 1, we have that $Sup(c_{k_1}) \geq Sup(X \Rightarrow c_{k_1})$, $Sup(c_{k_2}) \geq Sup(X \Rightarrow c_{k_2})$, $Sup(X) \geq Sup(X \Rightarrow c_{k_1}) + Sup(X \Rightarrow c_{k_2})$ and $Sup(X) \in [0, 1]$, therefore the following inequalities are fulfilled,

$$\begin{array}{ccc} \frac{Sup(c_{k_1})-Sup(X\Rightarrow c_{k_1})}{1-Sup(X)} & \geq & 0 \\ \frac{Sup(c_{k_2})-Sup(X\Rightarrow c_{k_2})}{1-Sup(X)} & \geq & 0 \\ 1 & \geq & \frac{Sup(X\Rightarrow c_{k_1})+Sup(X\Rightarrow c_{k_2})}{Sup(X)} \end{array} \quad (9)$$

Since the three inequalities of Eq. 9 have the same direction, we can add them obtaining

$$1 + \frac{Sup(c_{k_1})-Sup(X\Rightarrow c_{k_1})}{1-Sup(X)} + \frac{Sup(c_{k_2})-Sup(X\Rightarrow c_{k_2})}{1-Sup(X)} \geq \frac{Sup(X\Rightarrow c_{k_1})}{Sup(X)} + \frac{Sup(X\Rightarrow c_{k_2})}{Sup(X)}$$

and moving some terms to the right side we have

$$1 \geq \frac{Sup(X\Rightarrow c_{k_1})}{Sup(X)} - \frac{Sup(c_{k_1})-Sup(X\Rightarrow c_{k_1})}{1-Sup(X)} + \frac{Sup(X\Rightarrow c_{k_2})}{Sup(X)} - \frac{Sup(c_{k_2})-Sup(X\Rightarrow c_{k_2})}{1-Sup(X)} \quad (10)$$

Now, working with the first and second terms of the right side of (10)

$$\frac{Sup(X \Rightarrow c_{k_1})}{Sup(X)} - \frac{Sup(c_{k_1}) - Sup(X \Rightarrow c_{k_1})}{1 - Sup(X)} =$$
$$= \frac{Sup(X \Rightarrow c_{k_1}) - Sup(X \Rightarrow c_{k_1})Sup(X) - Sup(c_{k_1})Sup(X) + Sup(X \Rightarrow c_{k_1})Sup(X)}{Sup(X)(1 - Sup(X))}$$
$$= \frac{Sup(X \Rightarrow c_{k_1}) - Sup(c_{k_1})Sup(X)}{Sup(X)(1 - Sup(X))}$$
$$= Netconf(X \Rightarrow c_{k_1}) \qquad (see \quad Eq.7)$$

Analogously, $\frac{Sup(X \Rightarrow c_{k_2})}{Sup(X)} - \frac{Sup(c_{k_2}) - Sup(X \Rightarrow c_{k_2})}{1 - Sup(X)} = Netconf(X \Rightarrow c_{k_2})$ and substituting in (10) we obtain

$$1 \geq Netconf(X \Rightarrow c_{k_1}) + Netconf(X \Rightarrow c_{k_2})$$

which contradicts (8). □

Based on proposition 6, if we select a Netconf threshold $\gamma \geq 0.5$, for each itemset X we can obtain at most one CAR $X \Rightarrow c$, $c \in C$ such that $Netconf(X \Rightarrow c) > \gamma$, and in this way, we can select CARs with different antecedent and consequently the ambiguity at the classification stage is avoided. It is important to notice that a Netconf value greater than 0.5 can be considered as a high Netconf value because the Netconf takes values in [-1,1], being the dependence between antecedent and consequent more positive when the Netconf value is closer to 1. In our classifier, we want to compute as many CARs as possible but avoiding the ambiguity in the classification stage. Since the number of CARs increases when the Netconf threshold decreases and the smallest Netconf threshold value that avoids ambiguity in the classification stage is 0.5, then in our classifier we use this value as Netconf threshold.

3.2 CAR-CA algorithm

In order to generate the set of CARs, we propose an algorithm, called CAR-CA, which is a modification of the frequent itemset mining algorithm CA [18], which according to the experiments shown in [18], outperforms other efficient algorithms for mining frequent itemsets, as Apriori (used in CBA), Fp-growth (used in CMAR), Eclat (used in MCAR) and TFP (used in TFPC).

CAR-CA uses a new equivalence relation to group the CARs and bit-to-bit operations for fast computing Supports and in this way to speedup CAR computing employing the Netconf measure.

In [40], for mining ARs the authors propose partitioning the itemset space into equivalence classes grouping itemsets of the same size k which have a common $(k-1)$-length prefix. An equivalence class grouping k-itemsets will be denoted as EC_k. In CAR-CA, unlike the algorithm proposed in [40] we consider each predefined class $c \in C$ as another item and we propose to divide the CAR space into equivalence classes defined by the following equivalence relation: "The CARs of size k that share the consequent (the same class) and the first $k-2$ items of the antecedent (which has $k-1$ items) belong to the same equivalence class". In Fig. 1 we show graphically this equivalence relation.

In order to take advantage of bit-to-bit operations we represent the dataset as an m x n binary matrix being m the number of transactions and n the number of items including the class item. The binary values 1 and 0 denote the presence or absence of an item in a transaction, respectively. Each column, associated to an item j, can be compressed and represented as an integer array I_j, as follows:

$$I_j = \{W_{1,j}, W_{2,j}, \ldots, W_{q,j}\},\ q = \lceil m/32 \rceil \tag{11}$$

where each integer of the array represents 32 transactions (in a 32 bit architecture).

Previous algorithms developed for CAR mining need extra operations or extra dataset scans to compute the Support of rule antecedents. Our proposal avoids these extra operations; for that, it iteratively generates a list L_{EC_k} representing the equivalence classes containing k-CARs, whose elements have the next format:

$$\langle c, AntPref_{k-2}, IA_{AntPref_{k-2}}, AntSuff \rangle, \tag{12}$$

where c is the consequent of the grouped CARs, $AntPref_{k-2}$ is the $(k-2)$-itemset that is common to all the antecedents of the grouped CARs (antecedent's prefix), $AntSuff$ is the set of all items j which can extend $AntPref_{k-2}$ (antecedent suffixes), where j is lexicographically greater than each item in the antecedent prefix, and $IA_{AntPref_{k-2}}$ is an array of pairs $(value, id)$, with $value > 0$ and $1 \leq id \leq q$, that is built by intersecting (using AND operations) the arrays I_j, where j belongs to $AntPref_{k-2}$. The IA arrays store the Support of the antecedent prefix of each equivalence class EC_k, which is used to compute the Support of the rule antecedent of each CAR in EC_k. If k is large, the number of elements of IA is small because the AND operations generate null integers, and null integers are not stored because they do not have influence neither over the Support nor over the Netconf. The procedure for obtaining IA is as follows: Let i and j be two items, then:

$$IA_{\{i\}\cup\{j\}} = \{(W_{k,i} \mathbin{\&} W_{k,j}, k) \mid (W_{k,i} \mathbin{\&} W_{k,j}) \neq 0, k \in [1, q]\} \tag{13}$$

now let X be an itemset and j be an item, then:

$$IA_{X\cup\{j\}} = \{(b \mathbin{\&} W_{k,j}, k) \mid (b, k) \in IA_X, (b \mathbin{\&} W_{k,j}) \neq 0, k \in [1, q]\} \tag{14}$$

In order to compute the Support of an itemset X with an integer-array IA_X, the expression (15) is used:

$$Sup(X) = \sum_{(b,k)\in IA_X} BitCount(b) \tag{15}$$

where $BitCount(b)$ is a function that calculates the Hamming weight of b. The Netconf (Eq. 7) can be easily computed taking into account the format used to store the equivalence classes, see (12), and using the equations (13), (14) and (15).

To illustrate the overall CAR mining process, suppose that we have the equivalence class $E = \langle c_1, i_1i_2, IA_{i_1i_2}, \{i_3, i_4, \ldots\}\rangle$, which stores CARs of size 4 ($E \in EC_4$) *i.e.* $i_1i_2i_3 \Rightarrow c_1$, $i_1i_2i_4 \Rightarrow c_1$, etc. For simplicity, we assume a 4-bit architecture and we show, in Fig. 2, the arrays I_{c_1}, I_{i_3} and I_{i_4} for only 16 transactions (four blocks of four transactions each one). Additionally, we show the pairs $(value, id)$ resulting from the intersection of the arrays I_{i_1} and I_{i_2} (see $IA_{i_1i_2}$ in Fig. 2).

In Fig. 3, we show the steps for building the equivalence classes of EC_5 from the equivalence class E. In the first step, see Fig. 3(a), the $IA_{i_1i_2i_3}$ array is obtained by intersecting (using AND operations) the I_{i_3} array with the values stored in $IA_{i_1i_2}$, only the blocks with equal id are intersected. The $IA_{i_1i_2i_3}$ array stores the support of $i_1i_2i_3$, which can be computed as $BitCount(6) = 2$ (Eq. 15). In the second step, see Fig. 3(b), the $IA_{i_1i_2i_3i_4c_1}$ array is obtained in analogous way and the support of the rule $i_1i_2i_3i_4 \Rightarrow c_1$ can be computed as $BitCount(4) = 1$. If $Netconf(i_1i_2i_3i_4 \Rightarrow c_1)$ satisfies the Netconf threshold then the equivalence class $\langle c_1, i_1i_2i_3, IA_{i_1i_2i_2}, \{i_4, \ldots\}\rangle$ is built.

Using integer arrays, CAR-CA avoids extra operations or extra dataset scans to compute rule antecedent Supports, which increases its efficiency. At first sight, the use of integer arrays seems to require more memory than other classifiers but it is not true. The number of elements of the integer arrays stored by CAR-CA decreases rapidly because they are built using AND operations, which generate a lot of null integers, and null integers are not stored by our algorithm.

The efficiency of CAR-CA is based on two main features: the efficient use of equivalence classes and the efficient use of bit-to-bit operations for computing the Netconf of a CAR.

Recent algorithms for mining the set of CARs [15, 35, 36, 37] prune the CAR search space each time a CAR satisfying the Support and Confidence thresholds is found, it produces general (small) rules, with high Confidence. In our algorithm, instead of pruning the CAR search space when a CAR satisfies the Netconf threshold, we propose the following pruning strategy: If a candidate CAR $X \Rightarrow c$ does not satisfy the Netconf threshold we do not extended the CAR anymore avoiding to explore this part of the CAR search space in vain, *i.e.*, we prune the CAR search space avoiding to generate candidate CARs from CARs that do not satisfy the Netconf threshold. Otherwise, if the candidate CAR $X \Rightarrow c$ satisfies the Netconf threshold we follow extending it while $Netconf(X \cup \{i\} \Rightarrow c)$ is greater than or equal to $Netconf(X \Rightarrow c)$, thus we allow to obtain more specific rules (large rules) with high Netconf.

The pseudo code of CAR-CA is shown in Algorithm 1.

In line 3 of Algorithm 1, the 1-itemsets are calculated. In line 5, the equivalence classes of size 2 for each class c are built. In lines $7 - 15$, each equivalence class of size 2 is processed using the *ECGen* function.

The *ECGen* function takes, as an argument, an equivalence class of size $k - 1$ and generates a set of equivalence classes of size k (see Algorithm 2). The equivalence classes generated by *ECGen* only contain CARs with Netconf greater than 0.5.

Algorithm 1: CAR-CA

Input: Training dataset in binary representation
Output: Set of CARs

$Answer = \emptyset$
$C = \{$Set of pre-defined classes$\}$
$L = \{$1-itemsets$\}$
forall $c \in C$ **do**
  ECGen($\langle\{c\}, \emptyset, NULL, \{L\}\rangle, L_{EC_2} = \emptyset$)
  $Answer = Answer \cup L_{EC_2}$
  $k = 3$
  $L_{EC_k} = \emptyset, L_{EC_{k+1}} = \emptyset, \ldots$
  while $L_{EC_{k-1}} \neq \emptyset$ **do**
    forall $ec \in L_{EC_{k-1}}$ **do**
      ECGen(ec, L_{EC_k}) // *ec is in* $< \ldots >$ *format*
    end
    $Answer = Answer \cup L_{EC_k}$
    $k = k + 1$
  end
end
return $Answer$

Algorithm 2: ECGen

Input: An EC in $\langle c, AntPref, IA_{AntPref}, AntSuff\rangle$ format
A set of equivalence classes $ecSet$
Output: The updated set of equivalence classes $ecSet$

forall $i \in AntSuff$ **do**
  $AntPref' = AntPref \cup \{i\}$
  $IA_{AntPref'} = IA_{AntPref \cup \{i\} \cup \{c\}}$
  $AntSuff' = \emptyset$
  forall ($i' \in AntSuff$) *and* (i' *lexicographically greater than* i) **do**
    $NC =$ Netconf($AntPref' \Rightarrow c$)
    $NC' =$ Netconf($\{AntPref' \cup \{i'\}\} \Rightarrow c$)
    if $NC' > 0.5$ *and* $NC' >= NC$ **then**
      $AntSuff' = AntSuff' \cup \{i'\}$
    end
  end
  if $AntSuff' \neq \emptyset$ **then**
    $ecSet = ecSet \cup \{\langle c, AntPref', IA_{AntPref'}, AntSuff'\rangle\}$
  end
end
return $ecSet$

3.3 Ordering and Classifying

Once the set of CARs has been generated, using the CAR-CA algorithm, the CAR list is sorted. As mentioned earlier, for classifying, we will use specific (large) rules with high Netconf; for this purpose, we propose sorting the set of CARs in a descending order according to the size of the CARs (the largest first) and in case of tie, we sort the tied CARs in a descending order according to their Netconf (the highest values first).

The intuition behind this ordering is that more specific rules should be preferred before more general rules because in general more specific rules have a higher Netconf than general rules. In case of tie in size, rules with high Netconf valued should be preferred before rules with low Netconf value. Remember that a rule with high Netconf value has a high positive dependence between its antecedent and its consequent (the class) therefore, it is a good predictor. For example, in table 3(a), we have a classifier with three CARs, which are sorted by the criterion of the most general first. Given the transaction $\{i_1, i_2, i_3, i_4, i_5, i_6\}$, using the "Best rule" mechanism, this transaction would be classified as belonging to class c_1 when intuitively class c_2 or c_3 would be more likely to be the correct class because the rules $\{i_1 i_2 i_3\} \Rightarrow c_2$ and $\{i_4 i_5 i_6\} \Rightarrow c_3$ take into account three of the six items of the transaction, while the rule $\{i_1\} \Rightarrow c_1$ only considers the item i_1. In table 3(b), we show the same three CARs but sorted with our ordering strategy, more specific rules first and in case of tie in size, highest Netconf values first. Therefore, given the transaction $\{i_1, i_2, i_3, i_4, i_5, i_6\}$, our classifier assigns the class c_3 because the rule $\{i_4 i_5 i_6\} \Rightarrow c_3$ has the same size that the rule $\{i_1 i_2 i_3\} \Rightarrow c_2$ but the former has a grater Netconf value.

For classifying unseen transactions, we decided to follow the "Best K rules" satisfaction mechanism, because, as it was explained above, the "Best rule" mechanism could suffer biased classification or overfitting since the classification is based on only one rule; and the "All rules" mechanism takes into account rules with low ranking, which affects the accuracy of the classifier. Algorithms 3 and 4 show the pseudo code of the training phase and classification phase respectively:

Algorithm 3: CAR-NF (training phase)

Input: training dataset db
Output: the classifier

$Answer = \emptyset$
$CARs$ = `CAR-CA(`db`)`
$Answer$ = `Ordering_CARs(`$CARs$`)`
return $Answer$

Algorithm 4: CAR-NF (classification phase)

Input: set of sorted $CARs$, unseen transaction t
Output: the assigned class

$Answer = \emptyset$
$BestK$ = `Select_BestK(`t`)`
$Answer$ = `Classify(`$BestK$`)`
return $Answer$

In the training phase (Alg. 3), the $CAR - CA$ function computes the set of CARs from the training dataset. After, the $Ordering\_CARs$ function sorts the set of CARs in a descending order according to the size of the CARs and if there is a tie, in a descending order according to the Netconf.

In the classification phase (Alg. 4), to classify an unseen transaction t, for each class, the "best K rules" covering t are selected ($Select\_BestK$ function), and a class is assigned according to the average of the Netconf values ($Classify$ function). If there is a tie, one of the tied classes is randomly assigned. If no rule covers t, unlike other evaluated classifiers which assign the majority class, our classifier refuses to classify t, and such abstentions are counted as errors. This is done in order to avoid hiding uncovered transactions.

4 Experimental results

In this section, we report some experimental results where the CAR-NF classifier is compared against the main classifiers based on CARs reported in the literature: CBA [22], CMAR [21], CPAR [38], TFPC [15] and HARMONY [32]. Other good classifiers like RCBT [16] and DDPMine [11] were not included in our experiments because of the authors of these works did not provide the programs of their algorithms, which can not be implemented based on the details provided in [11] and [16]. Besides, the first one was evaluated using only four gene expression datasets, which were not provided by the authors; and the second one was evaluated using only 8 unusual datasets from the UCI repository.

The codes of CBA, CMAR, CPAR and TFPC were downloaded from the Frans Coenen's homepage (http://www.csc.liv.ac.uk/~frans) and for HARMONY, we used the accuracy values reported in [32]. In classification, the accuracy of a classifier depends on the number of transactions correctly classified and is computed as:

$$Accuracy = \frac{T}{S}$$

where T is the number of transactions correctly classified, and S is the total number of transactions presented to the classifier.

All our experiments were done using ten-fold cross-validation, reporting the average over the ten folds. Our tests were performed on a PC with an Intel Core 2 Duo at 1.86 GHz CPU with 1 GB DDR2 RAM, running Windows XP SP2.

In the same way as in other works [15, 21, 22, 38], experiments were conducted using several datasets, 20 in our case. The chosen datasets (see characteristic in Table 4) were originally taken from the UCI Machine Learning

Repository [4], and their numerical attributes were discretized by the author of [13] using the LUCS-KDD discretized/normalized ARM and CARM Data Library. The discretization technique used in LUCS-KDD is different from those used in [15, 21, 22, 38]; thus, our results reported in tables 6, 7 and 8 are different from previous studies, even for the same classifier and the same dataset [33]. However, in table 9, we show a comparison of the accuracies obtained by our classifier, CAR-NF, against the best reported accuracies of all the other evaluated classifiers.

For CBA, CMAR, CPAR and TFPC classifiers we used the Confidence threshold set to 50% and the Support threshold set to 1%, as their authors suggested because, after testing these classifiers with different threshold values as it is shown in table 5, the best results, appearing bold faced, were obtained for Support = 1% and Confidence = 50%, i.e., we obtained the same results as those reported by Coenen in [15]. In [32], the authors of HARMONY obtained the best results using a support threshold of 50%. In CAR-NF we used the Netconf threshold set to 0.5 (equivalent to 75% if we map Netconf from $[-1, 1]$ to $[0, 1]$) based on proposition 6 and in our previous analysis in section 3.1.

For classifying a new transaction we used the "Best K rules" satisfaction mechanism (section 2), which selects the best K rules per class (we used K=5 as in the other evaluated classifiers) satisfying the transaction to be classified, and later, the K rules having the greatest average Netconf determine the class that will be assigned to the new transaction.

In Table 6, the results show that CAR-NF yields an average accuracy higher than the other evaluated classifiers, having in average a difference in accuracy (average difference) of more than 1.5% with respect to the classifier in the second place. Table 7 shows for each dataset the differences between the accuracy of each classifier and the accuracy of the classifier in the first place (accuracy differences), Table 8 shows the position obtained, from 1 to 6, by each classifier according to its accuracy value (ranking position), *i.e.* the best average in the first place, the second best average in the second place and so on.

Analyzing these tables, we can see that CBA had the worst performance in average accuracy and average difference w.r.t. the best classifier while it had a good performance in average ranking; this is because, although CBA had low accuracy for some datasets (e.g. letRecog, ionosphere and mushroom), it reached the first or second place in 10 of the 20 datasets, as opposed to CMAR, which had a good average accuracy but a poor average ranking.

For CAR-NF, we can see that it was the best in accuracy as well as in average difference w.r.t. the best classifier; obtaining the best average ranking position. The classifier with the second best performance was HARMONY. HARMONY was the second best in average accuracy and average difference w.r.t. the best classifier; and it shared with CBA the second place in average ranking position.

Although the original implementations of CBA, CMAR and CPAR use different discretization/normalization techniques, we consider interesting to show, in Table 9, a comparison of the accuracies obtained by CAR-NF against the best reported accuracies of all the evaluated classifiers. In the case of HARMONY, the authors did not report which technique was used for discretization/normalization. Notice that, for this comparison, we used only 15 datasets because there are not values reported, for all classifiers, in the other datasets.

Despite the discretization/normalization technique is not the same, CAR-NF obtains the best average accuracy being 2.1% better than the second best.

5 Conclusions

In this paper, we have proposed an accurate classifier based on CARs. This classifier, called CAR-NF, introduces a new strategy for computing CARs, using the

Netconf as measure of interest, which allows pruning the CAR search space for obtaining specific rules with high Netconf (greater than 0.5 which corresponds to greater than 0.75 if we map Netconf from $[-1,1]$ to $[0,1]$). We also prove that the Netconf measure satisfies several desirable properties that a good rule quality measure should satisfy; additionally, we propose and prove a proposition that supports the use of a Netconf threshold equal to 0.5 for generating rules that avoid ambiguity at the classification stage. Besides, we propose a new way for ordering the set of CARs using the CAR size and the Netconf value.

The experimental results show that CAR-NF has better performance than CBA, CMAR, CPAR, TFPC and HARMONY classifiers. In general, CAR-NF has the best classification accuracy.

As future work, we are going to study the problem of producing rules with multiple labels, it means rules with multiple classes in the consequent. This kind of rules could be useful for problems where some transactions can belong to more than one class.

Acknowledgment

This work was partly supported by the National Council of Science and Technology of Mexico under the project CB-2008-01-106443 and grant 228056.

References

[1] R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules. *In Proceedings of 20th International Conference on Very Large Data Bases*, pages 487–499, 1994.

[2] K. I. Ahn and J. Y. Kim. Efficient Mining of Frequent Itemsets and a Measure of Interest for Association Rule Mining. *Information and knowledge Management*, 3(3):245–257, 2004.

[3] K. Ali., S. Manganaris, and R. Srikant. Partial classification using association rules. *In Proceedings of the 3rd International Conference on Knowledge Discovery in Databases and Data Mining*, pages 115–118, 1997.

[4] A. Asuncion and D.J. Newman. UCI machine learning repository. *In http://www.ics.uci.edu/∼mlearn/MLRepository.html*, 2007.

[5] F. Berzal, I. Blanco, D. Sánchez, and M. A. Vila. Measuring the accuracy and interest of association rules: A new framework. *Intelligence Data Analysis*, 6(3):221–235, 2002.

[6] F. Berzal, J. C. Cubero, D. Sánchez, and J. M. Serrano. ART: A Hybrid Classification Model. *Machine Learning*, 54(1):67–92, 2004.

[7] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. Classification and Regression Trees. *Wadsworth Publishing Company*, 1984.

[8] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic Itemset Counting and Implication Rules for Market Basket Data. *SIGMOD*, 26(2):255–264, 1997.

[9] S. Buddeewong and W. Kreesuradej. A New Association Rule-Based Text Classifier Algorithm. pages 684–685, Washington, DC, USA, 2005. IEEE Computer Society.

[10] E. Cesario, F. Folino, A. Locane, G. Manco, and R. Ortale. Boosting text segmentation via progressive classification. *Knowl. Inf. Syst.*, 15(3):285–320, 2008.

[11] Hong Cheng, Xifeng Yan, Jiawei Han, and Philip S. Yu. Direct Discriminative Pattern Mining for Effective Classification. *In Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*, pages 169–178, 2008.

[12] P. Clark and R. Boswell. Rule Induction with CN2: Some Recent Improvments. *In Proceedings of European Working Session on Learning (ESWL'91)*, pages 151–163, 1991.

[13] F. Coenen. The LUCS-KDD discretised/normalised ARM and CARM Data Library. Department of Computer Science, The University of Liverpool, UK(2003). *In http://www.csc.liv.ac.uk/~frans/KDD/Software/LUCS-KDD-DN.*

[14] F. Coenen and P. Leng. An Evaluation of Approaches to Classification Rule Selection. *In Proceedings of the Fourth IEEE International Conference on Data Mining*, pages 359–362, 2004.

[15] F. Coenen, P. Leng, and L. Zhang. Threshold Tuning for Improved Classification Association Rule Mining. *Lecture Notes in Artificial Intelligence: Advances in Knowledge Discovery and Data Mining - PAKDD 2005*, 3518:216–225, 2005.

[16] Gao Cong, Kian-Lee Tan, Anthony K. H. Tung, and Xin Xu. Mining top-K covering rule groups for gene expression data. *In Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 670–681, 2005.

[17] J. Han, J. Pei, and Y. Yin. Mining Frequent Patterns without a Candidate Generation. *In Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 1–12, 2000.

[18] R. Hernández, J. Hernández, J. A. Carrasco, and J. Fco. Martínez. Algorithms for mining frequent itemsets in static and dynamic datasets. *Intelligent Data Analysis*, 14(3):419–435, 2010.

[19] J. Holt and M. S. Chung. Multipass algorithms for mining association rules in text databases. *Knowl. Inf. Syst.*, 3(2):168–183, 2001.

[20] N. Lavraĉ, P. Flach, and B. Zupan. Rule Evaluation Measures: A Unifying View. *In Proceedings of the 9th International Workshop on Inductive Logic Programming (ILP'99)*, pages 174–185, 1999.

[21] W. Li, J. Han, and J. Pei. CMAR: accurate and efficient classification based on multiple class-association rules. *In Proceedings of the IEEE International Conference on Data Mining, ICDM 2001*, pages 369–376, 2001.

[22] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. *In proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, pages 80–86, 1998.

[23] S. H. Park, J. A. Reyes, D. R. Gilbert, J. W. Kim, and S. Kim. Prediction of protein-protein interaction types using association rule based classification. *BMC Bioinformatics*, 10(1), 2009.

[24] G. Piatetsky-Shapiro. *Discovery, Analysis, and Presentation of Strong Rules*, pages 229–238. AAAI/MIT Press, Cambridge, MA, 1991.

[25] J. R. Quinlan. C4.5: programs for machine learning. *Morgan Kaufmann Publishers Inc.*, 1993.

[26] J. R. Quinlan and R. M. Cameron-Jones. FOIL: A midterm report. *In Proceedings of the European Conference on Machine Learning, ECML 1993*, 667:3–20, 1993.

[27] C. Silverstein, S. Brin, and R. Motwani. Beyond Market Baskets: Generalizing Association Rules to Dependence Rules. *Data Mining and Knowledge Discovery*, pages 39–68, 1998.

[28] M. Steinbach and V. Kumar. Generalizing the notion of confidence. *Knowl. Inf. Syst.*, 12(3):279–299, 2007.

[29] A. M. Teredesai, M. A. Ahmad, J. Kanodia, and R. S. Gaborski. CoMMA: a framework for integrated multimedia mining using multi-relational associations. *Knowl. Inf. Syst.*, 10(2):135–162, 2006.

[30] A. Thabtah, P. Cowling, and Y. Peng. Multiple labels associative classification. *Knowl. Inf. Syst.*, 9(1):109–129, 2006.

[31] F. Thabtah, P. Cowling, and Y. Peng. MCAR: multi-class classification based on association rule. *In Proceedings of the 3rd ACS/IEEE International Conference on Computer Systems and Applications*, pages 33+, 2005.

[32] J. Wang and G. Karypis. On Mining Instance-Centric Classification Rules. *IEEE Trans. on Knowl. and Data Eng.*, 18(11):1497–1511, 2006.

[33] Jianyong Wang and George Karypis. HARMONY: Efficiently mining the best rules for classification. *In Proc. of SDM*, pages 205–216, 2005.

[34] W. Wang, Y. J. Wang, nares-Alcántara R. Ba Z. Cui, and F. Coenen. Application of Classification Association Rule Mining for Mammalian Mesenchymal Stem Cell Differentiation. pages 51–61, Berlin, Heidelberg, 2009. Springer-Verlag.

[35] Y. J. Wang, Q. Xin, and F. Coenen. A Novel Rule Ordering Approach in Classification Association Rule Mining. *In Proceedings of the 5th in-*

ternational conference on Machine Learning and Data Mining in Pattern Recognition, MLDM 2007, pages 339–348, 2007.

[36] Y. J. Wang, Q. Xin, and F. Coenen. A Novel Rule Weighting Approach in Classification Association Rule Mining. *Data Mining Workshops, International Conference on*, 0:271–276, 2007.

[37] Y. J. Wang, Q. Xin, and F. Coenen. Hybrid Rule Ordering in Classification Association Rule Mining. *Trans. MLDM*, 1(1):1–15, 2008.

[38] X. Yin and J. Han. CPAR: Classification based on Predictive Association Rules. *In Proceddings of the SIAM International Conference on Data Mining*, 2003.

[39] Y. Yoon and G. G. Lee. Practical Application of Associative Classifier for Document Classification. In G. G. Lee, A. Yamada, H. Meng, and S. H. Myaeng, editors, *Information Retrieval Technology, Second Asia Information Retrieval Symposium, AIRS 2005, Jeju Island, Korea, October 13-15, 2005, Proceedings*, volume 3689 of *Lecture Notes in Computer Science*, pages 467–478. Springer, 2005.

[40] M. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. New Algorithms for Fast Discovery of Association Rules. *In 3rd International Conference on Knowledge Discovery and Data Mining*, pages 283–286, 1997.

Table 1: (a) Transactional dataset D and (b) Supports of several itemsets in D.

(a)

T_{id}	i_1	i_2	i_3
t_1	1	0	0
t_2	0	0	1
t_3	1	1	1
t_4	0	1	1

(b)

Itemset	Support
$\{i_1\}$	0.5
$\{i_2\}$	0.5
$\{i_1 i_2\}$	0.25

Table 2: Different ways in which two itemsets can appear in a dataset.

Transactions		Support
¬X	¬Y	0.1
¬X	Y	0.5
X	Y	0.3
X	¬Y	0.1

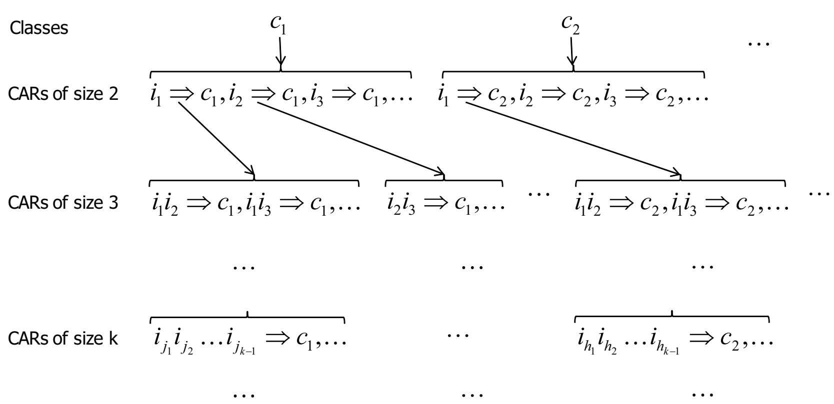


Fig. 1: CARs search space grouped in equivalence classes.

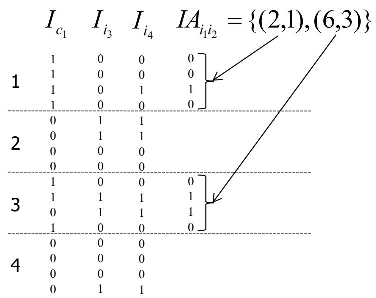
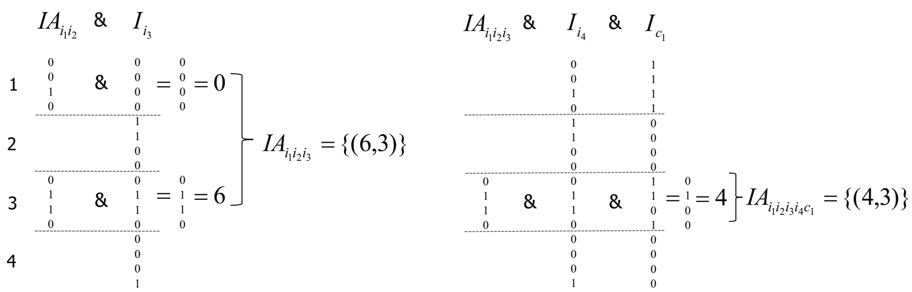


Fig. 2: Binary representation in a 4-bit architecture.



(a) Computing $IA_{i_1i_2i_3}$.

(b) Computing $IA_{i_1i_2i_3i_4c_1}$.

Fig. 3: Obtaining the equivalence classes of EC_5 from an equivalence class storing 4-CARs.

Table 3: Example of CARs ordering strategies.

(a) More general rules first

#	CAR	Netconf
1	$\{i_1\} \Rightarrow c_1$	0.75
2	$\{i_4i_5i_6\} \Rightarrow c_3$	0.75
3	$\{i_1i_2i_3\} \Rightarrow c_2$	0.70

(b) More specific rules first

#	CAR	Netconf
1	$\{i_4i_5i_6\} \Rightarrow c_3$	0.75
2	$\{i_1i_2i_3\} \Rightarrow c_2$	0.70
3	$\{i_1\} \Rightarrow c_1$	0.75

Table 4: Tested dataset characteristics.

Dataset	# instances	# items	# classes
adult	48842	97	2
anneal	898	73	6
breast	699	20	2
connect4	67557	129	3
dematology	366	49	6
ecoli	336	34	8
flare	1389	39	9
glass	214	48	7
heart	303	52	5
hepatitis	155	56	2
horseColic	368	85	2
ionosphere	351	157	2
iris	150	19	3
led7	3200	24	10
letRecog	20000	106	26
mushroom	8124	90	2
pageBlocks	5473	46	5
penDigits	10992	89	10
pima	768	38	2
waveform	5000	101	3

Table 5: Average accuracy of each algorithm over tested datasets, for different threshold values.

	% support / % confidence								
Classifier	1/40	1/50	1/60	10/40	10/50	10/60	20/40	20/50	20/60
CBA	70.26	**72.41**	69.38	68.42	68.76	66.35	41.67	41.73	40.68
CMAR	74.33	**77.63**	73.15	71.98	72.06	70.15	43.02	43.65	41.85
CPAR	74.12	**76.58**	73.26	71.44	71.68	70.01	44.21	44.30	42.92
TFPC	72.21	**75.46**	71.87	71.25	71.31	69.40	41.88	42.15	39.76

Table 6: Classification accuracy.

Dataset	CBA	CMAR	CPAR	TFPC	HARMONY	CAR-NF
adult	84.21	79.72	77.24	80.79	81.90	83.42
anneal	94.65	89.09	94.99	88.28	91.51	93.43
breast	94.09	88.84	92.95	89.98	92.42	85.26
connect4	66.67	64.83	65.15	65.83	68.05	62.18
dematology	80.00	82.92	80.08	76.30	62.22	78.78
ecoli	83.17	77.01	80.59	58.53	63.60	82.36
flare	84.23	83.30	64.75	84.30	75.02	86.31
glass	68.30	74.37	64.10	64.09	49.80	67.89
heart	57.33	55.36	55.03	51.42	56.46	56.79
hepatitis	57.83	81.16	74.34	81.16	83.16	85.87
horseColic	79.24	80.06	81.57	79.06	82.53	83.25
ionosphere	31.64	89.61	89.76	86.05	92.03	84.34
iris	94.00	92.33	94.70	95.33	93.32	96.67
led7	66.56	72.31	71.38	68.71	74.56	74.53
letRecog	28.64	26.25	28.13	27.57	76.81	71.14
mushroom	46.73	100.00	98.52	99.03	99.94	99.52
pageBlocks	90.94	87.98	92.54	89.98	91.60	92.44
penDigits	87.39	82.48	80.39	81.73	96.23	78.04
pima	75.03	72.85	74.82	74.36	72.34	77.65
waveform	77.58	72.22	70.66	66.74	80.46	74.68
Average	72.41	77.63	76.58	75.46	79.20	80.73

Table 7: Differences of accuracies respect to the best classifier.

Dataset	CBA	CMAR	CPAR	TFPC	HARMONY	CAR-NF
adult	0.00	4.49	6.97	3.42	2.31	0.79
anneal	0.34	5.90	0.00	6.71	3.48	1.56
breast	0.00	5.25	1.14	4.11	1.67	8.83
connect4	1.38	3.22	2.90	2.22	0.00	5.87
dematology	2.92	0.00	2.84	6.62	20.70	4.14
ecoli	0.00	6.16	2.58	24.64	19.57	0.81
flare	2.08	3.01	21.56	2.01	11.29	0.00
glass	6.07	0.00	10.27	10.28	24.57	6.48
heart	0.00	1.97	2.30	5.91	0.87	0.54
hepatitis	28.04	4.71	11.53	4.71	2.71	0.00
horseColic	4.01	3.19	1.68	4.19	0.72	0.00
ionosphere	60.39	2.42	2.27	5.98	0.00	7.69
iris	2.67	4.34	1.97	1.34	3.35	0.00
led7	8.00	2.25	3.18	5.85	0.00	0.03
letRecog	48.17	50.56	48.68	49.24	0.00	5.67
mushroom	53.27	0.00	1.48	0.97	0.06	0.48
pageBlocks	1.60	4.56	0.00	2.56	0.94	0.10
penDigits	8.84	13.75	15.84	14.5	0.00	18.19
pima	2.62	4.80	2.83	3.29	5.31	0.00
waveform	2.88	8.24	9.80	13.72	0.00	5.78
Average	11.66	6.44	7.49	8.61	4.88	3.35

Table 8: Ranking position based on accuracy.

Dataset	CBA	CMAR	CPAR	TFPC	HARMONY	CAR-NF
adult	1	5	6	4	3	2
anneal	2	5	1	6	4	3
breast	1	5	2	4	3	6
connect4	2	5	4	3	1	6
dematology	3	1	2	5	6	4
ecoli	1	4	3	6	5	2
flare	3	4	6	2	5	1
glass	2	1	4	5	6	3
heart	1	4	5	6	3	2
hepatitis	5	3	4	3	2	1
horseColic	5	4	3	6	2	1
ionosphere	6	3	2	4	1	5
iris	4	6	3	2	5	1
led7	6	3	4	5	1	2
letRecog	3	6	4	5	1	2
mushroom	6	1	5	4	2	3
pageBlocks	4	6	1	5	3	2
penDigits	2	3	5	4	1	6
pima	2	5	3	4	6	1
waveform	2	4	5	6	1	3
Average	3.05	3.90	3.60	4.45	3.05	2.80

Table 9: Comparison of the accuracies obtained by CAR-NF against the best accuracies reported by the methods CBA, CMAR, CPAR, TFPC and HARMONY.

Dataset	CBA-R	CMAR-R	CPAR-R	TFPC-R	HARMONY-R	CAR-NF
adult	84.20	80.10	76.70	80.80	81.90	83.42
anneal	97.90	97.30	98.40	88.30	91.51	93.43
ecoli	83.17	77.01	80.59	58.53	63.60	82.36
flare	84.20	84.30	64.75	84.30	75.02	86.31
glass	73.90	70.10	74.40	64.50	49.80	67.89
heart	81.90	82.20	82.60	51.40	56.46	56.79
hepatitis	81.80	80.50	79.40	81.20	83.16	85.87
horseColic	82.10	82.60	84.20	79.10	82.53	83.25
iris	94.70	94.00	94.70	95.30	93.32	96.67
led7	71.90	72.50	73.60	57.30	74.56	74.53
letRecog	28.64	25.50	28.13	26.40	76.81	71.14
mushroom	46.70	100.00	98.52	99.00	99.94	99.52
pageBlocks	90.90	90.00	92.54	90.00	91.60	92.44
pima	72.90	75.10	73.80	74.40	72.34	77.65
waveform	80.00	83.20	80.90	74.40	80.46	74.68
Average	76.99	79.63	78.88	73.66	78.20	81.73