Expert Systems with Applications 39 (2012) 11203–11211

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Classification based on specific rules and inexact coverage

Raudel Hernández-León [a,*], Jesús A. Carrasco-Ochoa [b], José Fco. Martínez-Trinidad [b], José Hernández-Palancar [a]

[a] Advanced Technologies Application Center (CENATAV), Havana, Cuba
[b] National Institute of Astrophysics, Optics and Electronics (INAOE), Puebla, Mexico

ARTICLE INFO

Keywords:
Data mining
Supervised classification
Class association rules
Association rule mining

ABSTRACT

Association rule mining and classification are important tasks in data mining. Using association rules has proved to be a good approach for classification. In this paper, we propose an accurate classifier based on class association rules (CARs), called CAR-IC, which introduces a new pruning strategy for mining CARs, which allows building specific rules with high confidence. Moreover, we propose and prove three propositions that support the use of a confidence threshold for computing rules that avoids ambiguity at the classification stage. This paper also presents a new way for ordering the set of CARs based on rule size and confidence. Finally, we define a new coverage strategy, which reduces the number of non-covered unseen-transactions during the classification stage. Results over several datasets show that CAR-IC beats the best classifiers based on CARs reported in the literature.

1. Introduction

Associative Classification, introduced in Liu, Hsu, and Ma (1998), is a well-known technique in data mining that aims to build a classifier based on class association rules (CARs) to determine the class of "unseen" transactions. Associative Classification integrates association rule mining (ARM) (Agrawal & Srikant, 1994) and classification rule mining (CRM) (Quinlan, 1993) to construct supervised classifiers also known as Associative Classifiers; a classifier based on this approach usually consists in an ordered list of CARs, which are used for classifying new transactions (Li, Han, & Pei, 2001; Liu et al., 1998; Yin & Han, 2003).

Nowadays, Associative Classification methods are increasingly applied in modern applications, such as prediction of protein–protein interaction types (Park, Reyes, Gilbert, Kim, & Kim, 2009), prediction of consumer behavior (Bae & Kim, 2010), text segmentation (Cesario, Folino, Locane, Manco, & Ortale, 2008), text classification (Buddeewong & Kreesuradej, 2005; Yoon & Lee, 2005), determination of DNA splice junction types (Berzal, Cubero, Sánchez, & Serrano, 2004), classification of mammograms (Dua, Singh, & Thompson, 2009), detection of breast cancer (Karabatak & Ince, 2009), obtention of minimal gene groups potentially responsible for the development of cancer (Dong & Li, 2005), analysis of real world geo-referenced census data (Ceci, 2006) and mammalian mesenchymal stem cell differentiation (Wang, Wang, Cui, & Coenen, 2009), among others.

In Associative Classification, similar to ARM, a set of items $I = \{i_1, i_2, \ldots, i_n\}$, a set of classes C, and a set of labeled transactions D, are given. Each transaction $t \in D$ contains a set of items $X \subseteq I$ and a class $c \in C$. The support of an itemset $X \subseteq I$ is the fraction of transactions in D containing X (see Eq. (1)). A CAR is a expression $X \Rightarrow c$, where $X \subseteq I$ and $c \in C$. The interest of a CAR is usually evaluated using the support and confidence measures. The support of a CAR $X \Rightarrow c$ (see Eq. (2)) is the fraction of transactions in D that contain $X \cup \{c\}$. The confidence of a CAR $X \Rightarrow c$ (see Eq. (3)) is the probability of finding c in transactions that contain X, which represents how "strongly" the rule antecedent X implies the rule consequent c.

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

where D_X is the set of transactions in D containing X and $|\cdot|$ represents the cardinality.

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

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

Some works (Agrawal & Srikant, 1994; Zaki, Parthasarathy, Ogihara, & Li, 1997) have mentioned that finding association rules is a hard problem that requires extensive computation capacity, especially when a small support threshold is used and the expected number of candidate ARs is too large. In order to deal with this problem into the context of CAR mining, some works (Coenen, Leng, & Zhang, 2005; Wang, Xin, & Coenen, 2007a, Wang, Xin, & Coenen, 2007b,

* Corresponding author. Tel.: +53 7 271 4787; fax: +53 7 273 0045.
E-mail addresses: rhernandez@cenatav.co.cu, raudel@ccc.inaoep.mx (R. Hernández-León), ariel@inaoep.mx (J.A. Carrasco-Ochoa), fmartine@inaoep.mx (J.F. Martínez-Trinidad), jpalancar@cenatav.co.cu (J. Hernández-Palancar).

0957-4174/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2012.03.057

2008) propose to prune the CAR space when a CAR satisfies predefined support and confidence thresholds, in this way general (small) rules are obtained. This strategy has two main drawbacks:

- Many branches of the CAR space could be explored in vain because the CARs are extended until they satisfy both support and confidence thresholds, which could never happen. For the support, this drawback is overcame by apriori-like algorithms that prune the search space using the anti-monotonic property of the support but the confidence measure does not satisfy the anti-monotonic property.
- Once a CAR $X \Rightarrow c$ is computed then the CAR $X' \Rightarrow c$ with $X \subset X'$ (a larger rule) never will be computed, this approach does not allow to generate specific (large) rules, some of which could be more interesting (*i.e.* these rules could have a higher confidence).

On the other hand, in previous works (Coenen et al., 2005; Li et al., 2001; Liu et al., 1998; Wang, Xin, & Coenen, 2008; Yin & Han, 2003), the concept of coverage of a transaction by a CAR is as follows: a CAR $X \Rightarrow c$ satisfies or covers a transaction t if $X \subseteq t$, which means that t must contain all the items of X; we will refer to this concept as exact coverage. Besides, in these works when no rule covers a transaction, the default class (majority class in all cases) is assigned and this could influence positively or negatively the results, hiding the real classification accuracy.

To solve these drawbacks, in this paper we introduce a new CAR based classifier called CAR-IC (Classification Association Rules with Inexact Coverage).

Preliminary results of this paper were published in Hernández, Carrasco, Martínez, and Hernández (2010). The main differences of this paper w.r.t. the conference paper (Hernández et al., 2010) are the following: (1) we prove the Propositions 1 and 2, which were only enunciated in Hernández et al. (2010); (2) we introduce and prove a new proposition, which guarantees that CAR-IC uses the minimum confidence threshold value that avoids ambiguity at the classification stage; (3) we introduce a new coverage strategy, which allows to reduce the number of non-covered unseen-transactions (default class assignations) improving the accuracy of CAR-IC; (4) the pruning strategy, the ordering strategy, the procedure for mining the CARs and the classification process are better explained and supported with examples, and (5) in the experiments, we included more datasets and new experiments to show the positive impact of the new coverage strategy in the classification stage, and to show the statistical significance of the results.

This paper is organized as follows: the next section describes the related work. In section three the CAR-IC classifier is introduced. Additionally, propositions that guarantee selecting a suitable confidence threshold value for computing rules that avoid ambiguity at the classification stage are presented and proved. In section four the experimental results, comparing CAR-IC against other state of the art classifiers based on CAR jointly with a discussion about these experiments are shown. Finally, conclusions as well as future work are given in section five.

2. Related work

The main classifiers based on CARs, reported in the literature, can be divided in two groups according to the approach used for obtaining the set of CARs:

1. Two Stage classifiers. In a first stage all CARs satisfying the support and confidence thresholds are mined and later, in a second stage, a classifier is built by selecting a small subset of CARs that fully covers the training set. CBA (Liu et al., 1998), CMAR (Li et al., 2001) and MCAR (Thabtah, Cowling, & Peng, 2005) follow this strategy.
2. Integrated classifiers. In these classifiers a small subset of CARs is directly generated. CPAR (Yin & Han, 2003), TFPC (Coenen et al., 2005) and HARMONY (Wang & Karypis, 2006) follow this strategy.

Regardless of the approach used for generating the set of CARs, there are five main schemes for ordering CARs:

(a) CSA (Confidence–Support–Antecedent size): The CSA ordering scheme (used by CBA (Liu et al., 1998)) sorts the rules in a descending order according to their confidence. Those CARs that share a common confidence value are sorted in a descending order according to their support, and in case of tie, CSA sorts the rules in ascending order according to the size of their rule antecedent. If all above criteria are identical, CSA sorts the rules following the order in which they were generated.

(b) ACS (Antecedent size–Confidence–Support): The ACS ordering scheme (used by TFPC (Coenen et al., 2005)) is a variation of CSA. But it takes the size of the rule antecedent as first ordering criterion, followed by the confidence and the support.

(c) WRA (Weighted Relative Accuracy): The WRA ordering scheme, proposed in Lavraĉ, Flach, and Zupan (1999) and used in three versions of the TFPC classifier (Wang et al., 2007a, 2007b, 2008), assigns to each CAR a weight based on its support and confidence and then sorts the set of CARs in a descending order according to these weights.

(d) LAP (Laplace Expected Error Estimate): The LAP ordering scheme was introduced by Clark and Boswell (1991) and it has been used to order CARs in CPAR (Yin & Han, 2003). This ordering scheme is similar to WRA but it computes the weights in a different way, also based on support and confidence.

(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, this value is used in CMAR (Li et al., 2001) to sort the set of CARs in a descending order.

In the literature there are three main satisfaction mechanisms (Li et al., 2001; Liu et al., 1998; Wang et al., 2008) to determine the class of an unseen transaction. Let t be a new transaction,

1. **Best rule**: This mechanism assigns to t the class associated with the first (“the best”) rule in the order that satisfies t (Liu et al., 1998).
2. **Best K rules**: This mechanism uses the best K rules in the order (for each class) that satisfy t for determining the class of t (Wang et al., 2008).
3. **All rules**: This mechanism uses all rules that satisfy t to determine the class of t (Li et al., 2001).

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 could include 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 used satisfaction mechanism for CAR based classifiers (Wang et al., 2007a, 2007b, 2008), and it has reported the best results, we will use it in our classifier.

3. CAR-IC classifier

We divide the explanation of the proposed classifier in 3 subsections. First of all, in Subsection 3.1, we propose and prove three propositions that guarantee using the minimum confidence threshold value, for computing a set of CARs, that avoids ambiguity at the classification stage. In Subsection 3.2, we describe the process for mining the CARs jointly with the new pruning strategy. In Section 3.3, a new way for ordering the CARs as well as a new coverage strategy for reducing the number of non-covered unseen-transactions are proposed. Finally, using all the above improvements the CAR-IC classifier is introduced.

3.1. Determining the confidence threshold

Commonly, CAR based classifiers use support and confidence thresholds for mining the set of CARs. However, these thresholds must be carefully defined by the user; there is not a guideline that helps the user to choose these threshold values. If we use too small threshold values (close to 0) then we have to deal with a large number of CARs, while if we use too big threshold values (close to 1) then many interesting CARs could be omitted. In this paper, unlike previous works reported in the literature, we propose to use the minimum confidence threshold that avoids ambiguity at the classification stage. We will say that two CARs are ambiguous if they have the same antecedent implying different classes.

In order to determine the minimum confidence threshold that avoids ambiguity we introduce three propositions. The Proposition 1 guarantees that the sum of the confidence values of all CARs having identical antecedent is 1. Later, the Proposition 2 shows that only one CAR can have a confidence value greater than 0.5 and the Proposition 3 guarantees that, for any dataset, 0.5 is the minimum threshold value that avoids CAR ambiguity at the classification stage.

Proposition 1. *Let X be an itemset and $C = \{c_1, c_2, \ldots, c_m\}$ be the set of predefined classes, the following equation is fulfilled:*

$$\sum_{i=1}^{m} Conf(X \Rightarrow c_i) = 1$$

Proof. From the definition of CAR confidence (Eq. (3)) we have:

$$\sum_{i=1}^{m} Conf(X \Rightarrow c_i) = \sum_{i=1}^{m} \frac{Sup(X \Rightarrow c_i)}{Sup(X)} = \frac{\sum_{i=1}^{m} Sup(X \Rightarrow c_i)}{Sup(X)} \quad (4)$$

From Eqs. (1) and (2), we have:

$$\sum_{i=1}^{m} Sup(X \Rightarrow c_i) = \sum_{i=1}^{m} Sup(X \cup \{c_i\}) = \sum_{i=1}^{m} \frac{|D_{X \cup \{c_i\}}|}{|D|} = \frac{\sum_{i=1}^{m} |D_{X \cup \{c_i\}}|}{|D|} \quad (5)$$

Since each transaction has one and only one class then $\sum_{i=1}^{m} |D_{X \cup \{c_i\}}| = |D_X|$ and

$$\sum_{i=1}^{m} Sup(X \Rightarrow c_i) = \frac{|D_X|}{|D|} = Sup(X) \quad (6)$$

Later, substituting (6) in (4)

$$\sum_{i=1}^{m} Conf(X \Rightarrow c_i) = \frac{Sup(X)}{Sup(X)} = 1 \qquad \square$$

Proposition 2. *Let X be an itemset and $C = \{c_1, c_2, \ldots, c_m\}$ be the set of predefined classes, at most one CAR $X \Rightarrow c_k$ ($c_k \in C$) has a confidence value greater than 0.5.*

Proof. From Eq. (3), we have that $Conf(X \Rightarrow c)$ takes values between 0 and 1 because *Conf* evaluates the probability of finding c in transactions that also contain X. Besides, from Proposition 1 we have that the sum of the confidence values of all CARs with equal antecedent is 1. Therefore, we can not have more than one CAR with equal antecedent and confidence value greater than 0.5 because in that case the sum of their confidence values would be greater than 1. $\square$

Based on Proposition 2, if we set the confidence threshold to 0.5, for each itemset X we can obtain at most one CAR having X as antecedent, with a confidence greater than 0.5. In this way, ambiguity at the classification stage is avoided. However, Proposition 2 does not guarantee that for all datasets 0.5 is the minimum confidence threshold that avoids ambiguity. Therefore, we will prove the following proposition:

Proposition 3. *Let X be an itemset and $C = \{c_1, c_2, \ldots, c_m\}$ be the set of predefined classes, independently of the dataset, 0.5 is the minimum confidence threshold value that avoids ambiguity in CARs having X as antecedent.*

Proof. In order to prove this proposition, we just have to find a dataset containing an itemset X where any confidence value smaller than 0.5 does not avoid ambiguity in CARs having to X as antecedent. Suppose a dataset D having only two classes c_1 and c_2 such that $|D_{\{c_1\}}| = |D_{\{c_2\}}|$ and containing the itemset X in all its transactions (see Table 1). In the dataset D, for all confidence threshold $\alpha < 0.5$ we have two CARs $X \Rightarrow c_1$ and $X \Rightarrow c_2$ having confidence values greater than α (see Eq. (7)).

$$Conf(X \Rightarrow c_1) = Conf(X \Rightarrow c_2) = \frac{n}{2 * n} = 0.5 > \alpha \qquad \square \quad (7)$$

Taking into account the previous analysis, we propose to use a confidence threshold value equal to 0.5 for computing CARs, which, independently of the dataset, is the minimum value that avoids CAR ambiguity at classification stage.

3.2. Mining the set of CARs

Similar to frequent itemset (FI) mining, the CARs mining task consumes too many time due to its combinatorial explosion. Usually, CAR computing implies traversing the CAR space (see Fig. 1). The recent approaches use either breadth first search (BFS) or depth first search (DFS). In a BFS strategy all CARs of size $k-1$ are computed before computing CARs of size k, and in a DFS strategy a recursive process allows to traverse the CAR space by adding items to the current CAR's antecedent until the CAR cannot grow anymore.

In order to generate the set of CARs, we propose a modification of the FI mining algorithm CA (Hernández et al., 2010), called CAR-CA. The CA algorithm, according to the experiments shown by their authors, outperforms other efficient algorithms for mining frequent itemsets, like Apriori (Agrawal & Srikant, 1994) (used in training phase of CBA (Liu et al., 1998)), Fp-growth (Han, Pei, &

Table 1
Dataset D used in the proof of Proposition 3.

Transaction	Itemset	Class
t_1	X	c_1
t_2	X	c_1
...	...	...
t_n	X	c_1
t_{n+1}	X	c_2
t_{n+2}	X	c_2
...	...	...
t_{2n}	X	c_2

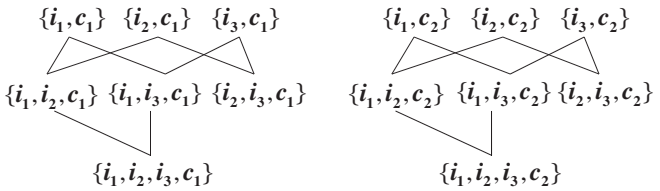

Fig. 1. CAR space for three items and two classes.

Yin, 2000) (used in training phase of CMAR (Li et al., 2001)) and TFP (Coenen, Leng, & Ahmed, 2004) (used in training phase of TFPC (Coenen et al., 2005)).

In Zaki et al. (1997), for mining FI, 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. Later, they use a DFS strategy for mining all FI. In this paper, we propose to divide the CAR space into equivalence classes defined by a new equivalence relation:

Definition 1. Let U be the CAR space, we will say that $r_1 \in U$ is related with $r_2 \in U$ ($r_1\ R\ r_2$) iff:

- r_1 and r_2 have the same size (k).
- r_1 and r_2 have the same consequent (the same class).
- The antecedents of r_1 and r_2, which have $k-1$ tems, have a common $k-2$-length prefix.

The above relation is clearly an equivalence relation because it is transitive, symmetric and reflexive. Equivalence relations define equivalence classes. All equivalence classes defined in a set Q, using an equivalence relation R, define a partition in Q, therefore, by structuring the CAR space in equivalence classes using the proposed equivalence relation we can calculate, separately, the rules belonging to each class. In Fig. 2, we show the CAR space partitioned in equivalence classes using the proposed equivalence relation. Based on this analysis, for mining the CAR space, we propose to combine the BFS and DFS strategies processing the equivalence classes in DFS form, and for each one, computing the k-CARs in BFS form.

Similar to CA (Hernández et al., 2010), in order to compute the support values fast, we take advantage of bit-to-bit operations by representing the dataset as an m x n binary matrix, where m is the number of transactions and n is the number of items including class items. 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) is compressed and represented as an integer array I_j, as follows:

$$I_j = \{W_{1,j}, W_{2,j}, \ldots, W_{q,j}\}, \quad q = \lceil m/32 \rceil \tag{8}$$

where each integer $W_{i,j}$ of the array I_j represents 32 transactions (in a 32 bit architecture). We iteratively generate a list L_{EC_k} representing the equivalence classes containing CARs of size k, each equivalence class EC_k in L_{EC_k} has the format:

$$\langle c, AntPref_{k-2}, IA_{AntPref_{k-2}}, AntSuff \rangle \tag{9}$$

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 $1 \leqslant id \leqslant q$, this array is built by intersecting (using AND operations) the integer arrays of items belonging 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 confidence. The procedure for obtaining IA is as follows: Let i and j be two items, then:

$$IA_{\{i\}\cup\{j\}} = \{(W_{k,i}\ \&\ W_{k,j}, k) | (W_{k,i}\ \&\ W_{k,j}) > 0, \quad k \in [1,q]\} \tag{10}$$

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

$$IA_{X\cup\{j\}} = \{(b\ \&\ W_{k,j}, k) | (b,k) \in IA_X, (b\ \&\ W_{k,j}) > 0, \quad k \in [1,q]\} \tag{11}$$

In order to compute the support of an itemset X with an integer array IA_X, the expression (12) is used:

$$Sup(X) = \sum_{(b,k)\in IA_X} BitCount(b) \tag{12}$$

where $BitCount(b)$ is a function that computes the Hamming weight of b.

In the literature, regardless of the strategy used for mining the CARs, during the mining stage each CAR satisfying the support and confidence thresholds is extended by adding a new item. The extended CAR, called candidate CAR, is searched in the transaction

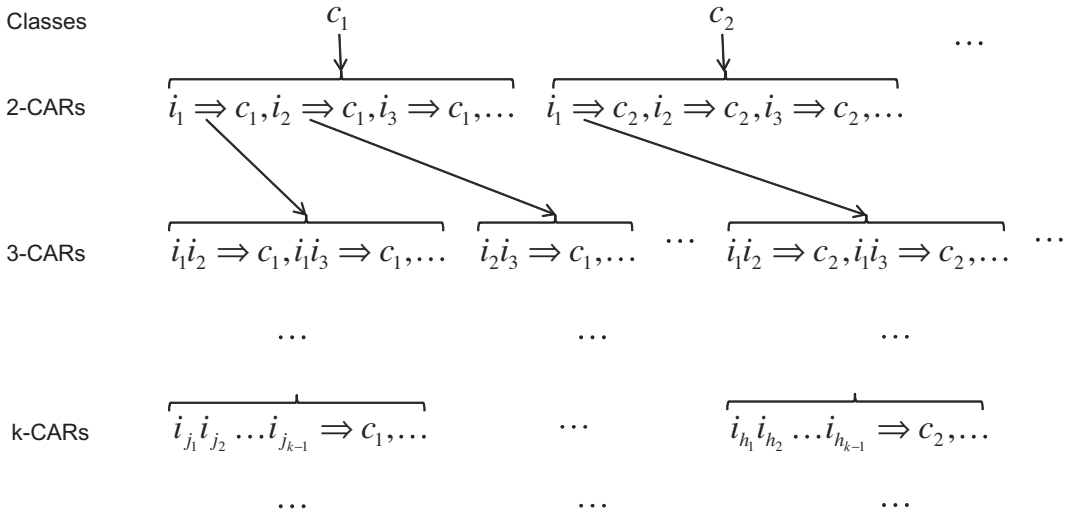

Fig. 2. CAR space structured in equivalence classes using the new equivalence relation.

dataset to compute its support and confidence values. Recent algorithms for mining the set of CARs (Coenen et al., 2005; Wang et al., 2007a, 2007b, 2008) prune the CAR space each time a CAR satisfying the support and confidence thresholds is found, and it results in general (small) rules with high confidence. In our proposal, instead of pruning the CAR space when a CAR satisfies the support and confidence thresholds, we propose the following pruning strategy:

- If a candidate CAR $X \Rightarrow c$ does not satisfy either the support threshold or the confidence threshold we do not extend the CAR anymore avoiding to explore this part of the CAR space in vain, *i.e.*, we do not generate candidate CARs from CARs wich do not satisfy either the support threshold or the confidence threshold.
- Otherwise, if a candidate CAR $X \Rightarrow c$ satisfies the confidence threshold we keep extending it while $Conf(X \cup \{i\} \Rightarrow c)$ is greater than or equal to $Conf(X \Rightarrow c)$, where i is a new item added to the rule's antecedent, thus we allow to obtain more specific (large) rules with high confidence.

3.3. Ordering and classifying

Once the set of CARs has been generated, the CAR list is sorted. As we mentioned earlier, for classifying unseen transactions, we will use specific (large) rules with high confidence. 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 confidence (the highest values first). If two rules have the same size and confidence, then we follow the order in which they were generated.

The intuition behind this ordering is that more specific rules should be preferred rather than more general rules because specific rules involve more items of an unseen transaction than general rules. If there is a tie in size, those rules with high confidence should be preferred rather than rules with low confidence. For example, in Table 2(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 2(b), we show the same three CARs but sorted by our ordering strategy, more specific rules first and for rules of the same size, highest confidence first. Therefore, given the transaction $\{i_1, i_2, i_3, i_4, i_5, i_6\}$, our classifier would assign the class c_3 because the rules $\{i_4 i_5 i_6\} \Rightarrow c_3$ and $\{i_1\ i_2\ i_3\} \Rightarrow c_2$ have the same size but the former has greater confidence.

Table 2
Example of CAR ordering strategies.

#	CAR	Confidence
(*a*) *More general rules first*		
1	$\{i_1\} \Rightarrow c_1$	0.65
2	$\{i_4 i_5 i_6\} \Rightarrow c_3$	0.65
3	$\{i_1 i_2 i_3\} \Rightarrow c_2$	0.60
(*b*) *More specific rules first*		
1	$\{i_4 i_5 i_6\} \Rightarrow c_3$	0.65
2	$\{i_1 i_2 i_3\} \Rightarrow c_2$	0.60
3	$\{i_1\} \Rightarrow c_1$	0.65

Table 3
Example of a set of interesting CARs.

CAR	Confidence
$\{i_1\} \Rightarrow c$	0.52
$\{i_1, i_2\} \Rightarrow c$	0.52
$\{i_1, i_2, i_3\} \Rightarrow c$	0.54
$\{i_1, i_2, i_3, i_4\} \Rightarrow c$	0.56
$\{i_5\} \Rightarrow c$	0.51
$\{i_5, i_6\} \Rightarrow c$	0.53
$\{i_5, i_6, i_7\} \Rightarrow c$	0.53

As it was mentioned in Section 1, in previous works, the authors considered that a CAR r covers a transaction t if the antecedent of the rule r is a subset of t. Consider the set of interesting CARs in Table 3, if you want to classify the transactions $\{i_2, i_3\}$ or $\{i_2, i_3, i_4\}$ then the majority class will be assigned (or the classifier refuses to classify) because, using exact coverage, these transactions are not covered by any CAR of Table 3 although $\{i_2, i_3\}$ and $\{i_2, i_3, i_4\}$ appear in the antecedents of some rules.

In order to reduce the number of non-covered unseen-transactions for those cases where the exact coverage does not work, we propose the following coverage definition:

Definition 2. A CAR $X \Rightarrow c$, with $|X| = n \geqslant 2$, inexactly covers a transaction t if there is an itemset $X' \subset t$ such that $X' \subset X$ and $|X'| = n - 1$.

Using this inexact coverage definition, the transactions $\{i_2, i_3\}$ and $\{i_2, i_3, i_4\}$ of our example would be covered by the CARs $\{i_1, i_2, i_3\} \Rightarrow c$ and $\{i_1, i_2, i_3, i_4\} \Rightarrow c$ respectively. This covering definition allows reducing the number of non-covered unseen-transactions.

For classifying unseen transactions, we decided to follow the "Best K rules" satisfaction mechanism, because, as it was explained in Section 2, the "Best rule" mechanism could suffer biased classification or overfitting since the classification is based on only one rule; and the "All rules" mechanism could take into account rules with low ranking, which affects the accuracy of the classifier.

Algorithm 1: Training phase.

Input: training dataset *db*
Outpur: the classifier (ordered set of CARs)
1 *Answer* = ∅
2 *CARs* = *Generating_CARs*(*db*)
3 *Answer* = *Ordering_CARs*(*CARs*)
4 **return** *Answer*

Using the improvements introduced in this section, we build a new classifier based on class association rules (named CAR-IC) as follows:

(a) In the training phase CAR-IC computes, from the training dataset, the set of CARs using the CAR-CA algorithm applying the proposed pruning strategy jointly with the confidence threshold value suggested by Proposition 3. After, CAR-IC sorts the set of CARs in a descending order according to the size of the CARs and if there is a tie, the CARs are sorted in a descending order according to their confidence. if the tie persists then CAR-IC orders the CARs according to the order they were generated.

Algorithm 2: Classification phase.

Input: set of sorted CARs, unseen transaction *t*
Output: the assigned class
1 *Answer* = ∅
2 *BestK* = *Select_BestK*(*t*)
3 *Answer* = *Classify*(*BestK*)
4 **return** *Answer*

(b) In the classification phase, CAR-IC classifies an unseen transaction t applying the "Best K rules" satisfaction mechanism (we used $K = 5$ as in the other evaluated classifiers). We first use the CARs that exactly cover t; if no CAR covers exactly t then CAR-IC uses those CARs that inexactly cover t. The class having the highest average confidence is assigned to t. If there is a tie, one of the tied classes is randomly assigned. If no rule covers t (exactly or inexactly), unlike other classifiers which assign the majority class hiding the real behavior of the classifier, CAR-IC refuses to classify t. However, in our experiments such abstentions are counted as errors. Although some real applications do not support abstentions, we prefer to include them in the experiments as errors for avoiding CAR-IC be favored from casual hits using the majority class.

Algorithms 1 and 2 show the pseudo code of the training and classification phases of the CAR-IC classifier, respectively.

4. Experimental results

In this section, we present the results of our experimental comparison between CAR-IC and the main classifiers based on CARs reported in the literature: CBA (Liu et al., 1998), CMAR (Li et al., 2001), CPAR (Yin & Han, 2003), TFPC (Coenen et al., 2005), HARMONY (Wang & Karypis, 2006) and DDPMine (Cheng, Yan, Han, & Yu, 2008).

The codes of CBA, CMAR, CPAR and TFPC were downloaded from the Frans Coenen's homepage (http://www.csc.liv.ac.uk/~frans), the code of DDPMine was provided by their authors and for HARMONY, we used the accuracy values reported in Wang and Karypis (2006). The accuracy of a classifier depends on the number of transactions correctly classified and it is computed as 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 and we report the average over the ten folds. The same folds were used for all classifiers. All the 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 (Coenen et al., 2005; Li et al., 2001; Liu et al., 1998; Yin & Han, 2003), we used several datasets, specifically 20 datasets also reported in Li et al. (2001), Liu et al. (1998), Wang and Karypis (2006), Yin and Han (2003). The chosen datasets (see characteristic in Table 4) were originally taken from the UCI Machine Learning Repository (Asuncion & Newman, 2007), and their numerical attributes were discretized by Frans Coenen using the LUCS-KDD (Coenen, 2003) discretized/normalized ARM and CARM Data Library. The discretization technique used in LUCS-KDD is different from those used in Coenen et al. (2005), Li et al. (2001), Liu et al. (1998), Yin and Han (2003); this is the reason because our results reported in Table 6 are different from other results previously reported, even though the results were obtained using the same classifier and the same dataset (Wang & Karypis, 2005).

For CBA, CMAR, CPAR and TFPC classifiers we used the confidence threshold set to 0.5 and the support threshold set to 0.01, as their authors suggested. No matter this suggestion, we tested these classifiers with other threshold values (see Table 5), and we confirmed the best results are obtained using 0.01 as support threshold and 0.5 as confidence threshold; as Coenen reported in Coenen et al. (2005). For CAR-IC we used 0.5 as confidence threshold, based on the analysis presented in Section 3.1. Notice that the bold values is Table 5 as well as in Tables 6 and 8 indicate the best value of each row.

In Table 6, the results show that CAR-IC yields an average accuracy higher than all other evaluated classifiers, having a difference of 0.88% with respect to the classifier in the second place. Table 7 shows the position obtained, from 1 to 6, by each classifier according to its average accuracy (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 average accuracy however it had a good average ranking; this is because, although CBA had low accuracy for some datasets (*e.g.* letRecog, ionosphere and mushroom), it reached the first, second or third place in 11 of the 20 datasets. On the other hand, CMAR had a good average accuracy but a poor average ranking. The proposed classifier, CAR-IC, was the best in average accuracy as well as in average ranking, in this last case tied with DDPMine, which was the second best in average accuracy.

As we can see in Table 4, the used datasets have between 2 and 26 classes. In order to show the accuracy of CAR-IC on the different datasets, we grouped them according to the number of classes, in three groups: (1) datasets with 2 or 3 classes; (2) datasets with 4, 5, 6 or 7 classes and (3) datasets with more than 7 classes. In Table 8 we report the average accuracy of CAR-IC and DDPMine

Table 4
Dataset characteristics.

Dataset	# Instances	# Items	# Classes
Adult	48842	97	2
Anneal	898	73	6
Breast	699	20	2
Connect4	67557	129	4
Dermatology	366	49	6
E-coli	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 CBA, CMAR, CPAR and TFPC classifiers over the tested datasets, for different support and confidence values.

Classifier	% Support/ % Confidence								
	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	DDPMine	CAR-IC
Adult	**84.21**	79.72	77.24	80.79	81.90	82.82	82.61
Anneal	94.65	89.09	**94.99**	88.28	91.51	90.86	92.73
Breast	**94.09**	88.84	92.95	89.98	92.42	86.53	90.03
Bonnect4	66.67	64.83	65.15	65.83	**68.05**	67.80	56.02
Dermatology	80.00	**82.92**	80.08	76.30	62.22	63.42	80.16
E-coli	**83.17**	77.01	80.59	58.53	63.60	64.25	82.06
Flare	84.23	83.30	64.75	84.30	75.02	77.10	**85.98**
Glass	68.30	**74.37**	64.10	64.09	49.80	53.61	68.95
Heart	**57.33**	55.36	55.03	51.42	56.46	57.19	54.35
Hepatitis	57.83	81.16	74.34	81.16	83.16	82.29	**84.62**
HorseColic	79.24	80.06	81.57	79.06	**82.53**	81.07	82.47
Ionosphere	31.64	89.61	89.76	86.05	92.03	**93.25**	86.10
Iris	94.00	92.33	94.70	95.33	93.32	94.03	**96.67**
Led7	66.56	72.31	71.38	68.71	**74.56**	73.98	73.02
LetRecog	28.64	26.25	28.13	27.57	**76.81**	76.12	73.14
Mushroom	46.73	**100.00**	98.52	99.03	99.94	**100.00**	98.54
PageBlocks	90.94	87.98	92.54	89.98	91.60	**93.24**	92.26
PenDigits	87.39	82.48	80.39	81.73	96.23	**97.87**	81.93
Pima	75.03	72.85	74.82	74.36	72.34	75.22	**76.01**
Waveform	77.58	72.22	70.66	66.74	80.46	**83.83**	74.39
Average	72.41	77.63	76.58	75.46	79.20	79.72	**80.60**

Table 7
Ranking position based on accuracy.

Dataset	CBA	CMAR	CPAR	TFPC	HARMONY	DDPMine	CAR-IC
Adult	1	6	7	5	4	2	3
Anneal	2	6	1	7	4	5	3
Breast	1	6	2	5	3	7	4
Connect4	3	6	5	4	1	2	7
Dermatology	4	1	3	5	7	6	2
E-coli	1	4	3	7	6	5	2
Flare	3	4	7	2	6	5	1
Glass	3	1	4	5	7	6	2
Heart	1	4	5	7	3	2	6
Hepatitis	6	4	5	4	2	3	1
HorseColic	6	5	3	7	1	4	2
Ionosphere	7	4	3	6	2	1	5
Iris	5	7	3	2	6	4	1
Led7	7	4	5	6	1	2	3
LetRecog	4	7	5	6	1	2	3
Mushroom	6	1	5	3	2	1	4
PageBlocks	5	7	2	6	4	1	3
PenDigits	3	4	7	6	2	1	5
Pima	3	6	4	5	7	2	1
Waveform	3	5	6	7	2	1	4
Average	3.70	4.60	4.25	5.25	3.55	3.10	3.10

Table 8
Average accuracy of DDPMine and CAR-IC over datasets grouped by number of classes.

# classes	DDPMine	CAR-IC
2 or 3	**86.56**	85.72
4,5,6 or 7	71.02	**74.08**
more than 7	77.86	**79.23**

(the second best) classifiers over these three groups of datasets, in this experiment we can see that CAR-IC works well independently of the number of classes in the datasets.

In Table 9, we show the percent of abstentions and the accuracy of CAR-IC using exact and inexact coverage. When we used inexact coverage, CAR-IC had, in average, 1.35% less abstentions than using exact coverage and its average accuracy was improved in 1.02%. Notice that, using exact coverage, CAR-IC obtains an accuracy of 79.58 (see Table 9), which is greater than the accuracy of the other classifiers with the exception of DDPMine, showing that the proposed pruning and ordering strategies together, even without using inexact coverage, result in a good classifier. Additionally, in the last column of Table 9, we show the accuracy obtained if the majority class is assigned when no rule covers an unseen transaction after checking exact and inexact coverage, as the other classifiers do. In these experiments, we can clearly see that assigning the majority class affects the results (in this case increasing the accuracy).

Finally, in order to determine if the results shown in Table 6 are statistically significant, we performed a pairwise comparison between CAR-IC and the other tested classifiers. Each cell in Table 10 contains the number of datasets where our classifier significantly Win/Lose to each other classifier. We detected ties using a one-tailed T-Test (Demšar, 2006; Dietterich, 1998) with

Table 9
Percentage of abstentions (%Abst.) and accuracy (Acc.) of CAR-IC using exact (EC) and inexact (IC) coverage. The last column (default class) shows the accuracy if the default class is assigned when no CAR covers an unseen transaction.

Dataset	%Abst. (EC)	Acc.	%Abst. (IC)	Acc.	default class
Adult	0.85	82.11	0.33	82.61	84.22
Anneal	1.11	91.80	0.12	92.73	92.97
Breast	7.63	84.42	1.59	90.03	91.12
Connect4	0.83	56.02	0.80	56.02	56.99
Dermatology	5.46	78.43	2.73	80.16	80.51
E-coli	1.65	82.06	1.32	82.06	82.59
Flare	1.04	85.98	1.04	85.98	86.57
Glass	3.63	68.12	2.08	68.95	69.53
Heart	5.50	53.21	4.03	54.35	54.46
Hepatitis	1.43	84.56	0.72	84.62	85.92
HorseColic	2.42	82.47	1.81	82.47	83.54
Ionosphere	6.65	84.07	3.80	86.10	86.83
Iris	1.48	96.06	0.74	96.67	97.37
Led7	2.29	72.71	1.42	73.02	73.45
LetRecog	1.45	73.14	1.28	73.14	73.61
Mushroom	0.25	98.54	0.22	98.54	98.95
PageBlocks	1.08	91.82	0.53	92.26	92.58
PenDigits	6.40	77.86	2.14	81.93	82.38
Pima	4.63	75.23	3.62	76.01	76.77
Waveform	1.96	73.06	0.51	74.39	75.33
Average	2.89	79.58	1.54	80.60	81.28

Table 10
Pairwise comparison between CAR-IC and the other classifiers. Each cell shows the number of times CAR-IC Win/Lose with respect to the corresponding classifier over the 20 selected datasets.

	CBA	CMAR	CPAR	TFPC	HARMONY	DDPMine	CAR-IC
CBA		11/7	8/5	10/5	8/8	7/9	6/12
CMAR	7/11		8/8	8/5	5/12	5/12	4/14
CPAR	5/8	8/8		10/3	6/11	5/11	4/14
TFPC	5/10	5/8	3/10		5/14	4/13	1/16
HARMONY	8/8	12/5	11/6	14/5		2/7	5/10
DDPMine	9/7	12/5	11/5	13/4	7/2		5/9
CAR-IC	12/6	14/4	14/4	16/1	10/5	9/5	

significance level of 0.05. The results in the pairwise comparison reveal that the CAR-IC classifier beats in accuracy all other evaluated classifiers over most of the tested datasets.

5. Conclusions

In this paper, a new accurate classifier based on CARs was proposed. This classifier, called CAR-IC, introduces a new pruning strategy for obtaining specific rules with as high as possible confidence value, but avoiding ambiguity at the classification stage. Besides, CAR-IC proposes a new way for ordering the set of CARs based on the size of the CARs and their confidence value. Finally, CAR-IC also introduces a new coverage strategy for reducing the number of non-covered unseen-transactions.

Based on our experiments, we can conclude that the proposed improvements allow to improve the accuracy obtained by other CAR based classifiers (CBA, CMAR, CPAR, TFPC, HARMONY and DDPMine). Additionally, we can conclude that assigning the majority class, when no rule covers an unseen transactions, as all CAR based classifiers do, hides the real behavior of the classifier.

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 are necessary for problems where the objects can belong to more than one class. For this, we will study the use of confidence thresholds smaller than 0.5 allowing to obtain more than one rule with the same antecedent.

References

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. In *Proceedings of 20th international conference on very large data bases* (pp. 487–499).

Asuncion, A., & Newman, D. (2007). UCI machine learning repository. URL <http://www.ics.uci.edu/~mlearn/MLRepository.html>

Bae, J. K., & Kim, J. (2010). Integration of heterogeneous models to predict consumer behavior. *Expert Systems with Applications, 37*, 1821–1826.

Berzal, F., Cubero, J. C., Sánchez, D., & Serrano, J. M. (2004). ART: A hybrid classification model. *Machine Learning, 54*(1), 67–92.

Buddeewong, S., & Kreesuradej, W. (2005). A new association rule-based text classifier algorithm. In *Proceedings of the 17th IEEE international conference on tools with artificial intelligence* (pp. 684–685). Washington, DC, USA: IEEE Computer Society.

Ceci, M. (2006). Spatial associative classification: Propositional vs structural approach. *Intelligent Information Systems*, 191–213.

Cesario, E., Folino, F., Locane, A., Manco, G., & Ortale, R. (2008). Boosting text segmentation via progressive classification. *Knowledge and Information Systems, 15*(3), 285–320.

Cheng, H., Yan, X., Han, J., & Yu, P. S. (2008). Direct discriminative pattern mining for effective classification. In *Proceedings of the 2008 IEEE 24th international conference on data engineering* (pp. 169–178). Washington, DC, USA: IEEE Computer Society.

Clark, P., Boswell, R. (1991). Rule induction with CN2: Some recent improvments. In *Proceedings of european working session on learning* (pp. 151–163).

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

Coenen, F., Leng, P., & Ahmed, S. (2004). Data structures for association rule mining: T-trees and P-trees. *IEEE Transactions on Knowledge and Data Engineering, 16*, 774–778.

Coenen, V., Leng, P., & Zhang, L. (2005). Threshold tuning for improved classification association rule mining. In *Lecture notes in artificial intelligence: Advances in knowledge discovery and data mining* Vol. 3518, (pp. 216–225).

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research, 7*, 1–30.

Dietterich, T. G. (1998). Aproximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation, 10*(7), 1895–1923.

Dong, G., & Li, J. (2005). Mining border descriptions of emerging patterns from dataset pairs. *Knowledge and Information Systems, 8*(2), 178–202.

Dua, S., Singh, H., & Thompson, H. W. (2009). Associative classification of mammograms using weighted rules. *Expert Systems with Applications, 36*(5), 9250–9259.

Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without a candidate generation. In *Proceedings of the ACM SIGMOD international conference on management of data* (pp. 1–12).

Hernández, R., Carrasco, J.A., Martínez, J.F., & Hernández, J. (2010). Classifying using specific rules with high confidence. In *Proceedings of the 9th mexican international conference on artificial intelligence, Special session: Advances in artificial intelligence and applications* (pp. 75–80). IEEE Computer Society.

Karabatak, M., & Ince, M. C. (2009). An expert system for detection of breast cancer based on association rules and neural network. *Expert Systtems with Applications, 36*, 3465–3469.

Lavraĉ, N., Flach, P., & Zupan, B. (1999). Rule evaluation measures: A unifying view. In *Proceedings of the 9th international workshop on inductive logic programming* (pp. 174–185). Springer-Verlag.

Li, W., Han, J., & Pei, J., 2001. CMAR: accurate and efficient classification based on multiple class-association rules. In *Proceedings of the IEEE international conference on data mining* (pp. 369–376).

Liu, B., Hsu, W., & Ma, Y. (1998). Integrating classification and association rule mining. In *Proceedings of the 4th international conference on knowledge discovery and data mining* (pp. 80–86).

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

Quinlan, J. R. (1993). C4.5: programs for machine learning. San Francisco, CA, USA.

Thabtah, F., Cowling, P., & Peng, Y. (2005). MCAR: multi-class classification based on association rule. In *Proceedings of the 3rd ACS/IEEE international conference on computer systems and applications*. (pp. 33+).

Wang, J., & Karypis, G. (2005). HARMONY: Efficiently mining the best rules for classification. In *Proceedings of SDM* (pp. 205–216).

Wang, J., & Karypis, G. (2006). On mining instance-centric classification rules. *IEEE Transactions on Knowledge and Data Engineering, 18*(11), 1497–1511.

Wang, W., Wang, Y.J., R. Ba n.-A., Cui, Z., & Coenen, F. (2009). Application of classification association rule mining for mammalian mesenchymal stem cell differentiation. In: *Proceedings of the 9th industrial conference on advances in data mining. applications and theoretical aspects* (pp. 51–61). Berlin, H eidelberg: Springer-Verlag.

Wang, Y.J., Xin, Q., & Coenen, F. (2007a). A novel rule ordering approach in classification association rule mining. In *Proceedings of the 5th international conference on machine learning and data mining in pattern recognition* (pp. 339–348).

Wang, Y.J., Xin, Q., & Coenen, F. (2007b). A Novel Rule Weighting Approach in Classification Association Rule Mining. In *International conference on data mining workshops*, 0 (pp. 271–276).

Wang, Y. J., Xin, Q., & Coenen, F. (2008). Hybrid rule ordering in classification association rule mining. *Transactions on MLDM, 1*(1), 1–15.

Yin, X., & Han, J. (2003). CPAR: Classification based on predictive association rules. In *Proceddings of the SIAM international conference on data mining*.

Yoon, Y., & Lee, G. G. (2005). Practical application of associative classifier for document classification. In G. G. Lee, A. Yamada, H. Meng, & S. H. Myaeng (Eds.), *Proceedings of the second asia information retrieval symposium. Lecture notes in computer science* (Vol. 3689, pp. 467–478). Springer.

Zaki, M., Parthasarathy, S., Ogihara, M., & Li, W. (1997). New algorithms for fast discovery of association rules. In *3rd international conference on knowledge discovery and data mining* (pp. 283–286).