Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
Decision Support Systems 42 (2006) 674–689
Decision Support Systems
www.elsevier.com/locate/dsw

A new approach to classification based on association rule mining

Guoqing Chen [*], Hongyan Liu, Lan Yu, Qiang Wei, Xing Zhang

Department of Management Science and Engineering, School of Economics and Management, Tsinghua University, Beijing 100084, China

Received 19 February 2004; received in revised form 9 March 2005; accepted 9 March 2005
Available online 25 July 2005

Abstract

Classification is one of the key issues in the fields of decision sciences and knowledge discovery. This paper presents a new approach for constructing a classifier, based on an extended association rule mining technique in the context of classification. The characteristic of this approach is threefold: first, applying the information gain measure to the generation of candidate itemsets; second, integrating the process of frequent itemsets generation with the process of rule generation; third, incorporating strategies for avoiding rule redundancy and conflicts into the mining process. The corresponding mining algorithm proposed, namely GARC (Gain based Association Rule Classification), produces a classifier with satisfactory classification accuracy, compared with other classifiers (e.g., C4.5, CBA, SVM, NN). Moreover, in terms of association rule based classification, GARC could filter out many candidate itemsets in the generation process, resulting in a much smaller set of rules than that of CBA.
© 2005 Elsevier B.V. All rights reserved.

Keywords: Data mining; Association rule; Classification; Information gain

1. Introduction

Classification is one of the key issues in the field of decision sciences, a field which plays an important role in supporting business and scientific decision-making. In recent years, it has also been one of the focal points in data mining and knowledge discovery. Classification is finding a classifier that results from training datasets with predetermined targets, fine-tuning it with test datasets, and using it to classify other datasets of interest. There exists various ways of constructing classifiers in the form of, for example, rules, decision trees, Bayesian networks, support vectors machine, etc. [12,14–16,21,24,26,29–31]. Decision trees classifiers, such as Quinlan's C4.5/5.0 classifier and its extensions [30], have received considerable attention due to its speed and understandability. Moreover, a number of efforts have been put forward to focus on the various aspects of improvements [5,9,25,33]. Another type of classification technique that has attracted an increasing number of attempts in recent years is finding classification rules based on association rule mining techniques, e.g., Refs. [4,20–23,29].

A classification rule is of the form $X \Rightarrow C$, where X is a set of data items, and C is a class (label) and a predetermined target. With such a rule, a transaction or data record t in a given database could be classified into class C if t contains X. Apparently, a classifica-

* Corresponding author. Tel.: +86 10 62772940; fax: +86 10 62785876.
E-mail address: chengq@em.tsinghua.edu.cn (G. Chen).

0167-9236/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.dss.2005.03.005

tion rule could be regarded as an association rule of a special kind.

Roughly speaking, an association rule is a relationship between data items. Two measures, namely the Degree of Support (D_{supp}) and the Degree of Confidence (D_{conf}), are used to define a rule. For example, a rule like "Milk⇒Diaper with $D_{\text{supp}}=20\%$, $D_{\text{conf}}=80\%$" means that "20% of the customers bought both Milk and Diaper" and that "80% of the customers who bought Milk also bought Diaper". That is, D_{supp} corresponds to statistical significance, while D_{conf} is a measure of the rule's strength [3].

Formally, let $\boldsymbol{I}=\{I_i,\ i=1,\ldots,s\}$ be a set of items. A transaction database T is a set of transactions, where each transaction t is a set of items such that $t\subseteq\boldsymbol{I}$. An association rule is of the form $X\Rightarrow Y$, where $X\subset\boldsymbol{I}$, $Y\subset\boldsymbol{I}$ are called itemsets, and $X\cap Y=\emptyset$. A transaction t is called to contain X, if $X\subseteq t$. Let $D_{\text{supp}}(X)$ be the fraction of transactions that contain X in a database T, $D_{\text{supp}}(X)=\|X\|/|T|$. The degree of support and degree of confidence for a rule $X\Rightarrow Y$ are defined as follows:

$$D_{\text{supp}}(X\Rightarrow Y)=\|X\cup Y\|/|T|$$

$$D_{\text{conf}}(X\Rightarrow Y)=\|X\cup Y\|/\|X\|$$

where X and Y are itemsets with $X\cap Y=\emptyset$, T is the set of all the transactions contained in the database concerned, $\|X\|$ is the number of the transactions in T that contain X, $\|X\cup Y\|$ is the number of the transactions in T that contain X and Y, and $|T|$ is the number of the transactions in T. In other words, $D_{\text{supp}}(X\Rightarrow Y)$ is the percentage of transactions containing both X and Y in the whole dataset, while $D_{\text{conf}}(X\Rightarrow Y)$ is the ratio of the number of transactions that contain X and Y over the number of transactions that contain X. They are used to evaluate a rule against given thresholds, minimal support α and minimal confidence β, respectively. In particular, if D_{supp} of an itemset X is no less than α (i.e., $D_{\text{supp}}(X)\geq\alpha$), then X is called a frequent itemset, otherwise called an excluded itemset. There have been many efforts proposed to discover association rules in various ways [1,2,8,11–13,17,28,32,34,37], among which the Apriori algorithm by Agrawal and Srikant [1] is usually deemed as a classical algorithm.

In the classification based on association rules mining, a well-known method, namely the CBA method proposed by Liu et al. [21] and its modifications [20,23], uses an Apriori-type association rule mining approach [1] to generate classification rules, which usually generates all the frequent itemsets, followed by the rule generation process. Subsequently, filters may be applied to the rules so as to eliminate non-interesting ones such as conflicts and so on. In other words, basically, CBA directly employs the Apriori-type approach for a particular kind of association rule, namely classification rules in forms of $X\Rightarrow C$. Thus, its efficiency heavily relies on the process of generating frequent itemsets. Like conventional association rules, classification rules are generated based on all the frequent itemsets generated. Then these rules are sorted according to a filtering measure, if desired.

While classifiers in forms of rules are often appealing for use and explanation by decision makers, directly applying the Apriori-type approach may however result in a large number of itemsets and then of rules, which would further increase the effort for understanding the rules as well as for resolving rule redundancy and conflicts. Therefore, it is considered desirable if some strategies such as itemset reduction and redundancy/conflict resolutions could be incorporated into the process of frequent itemsets generation, such that fewer itemsets need to be generated and therefore with fewer resultant rules. Apparently, a smaller set of classification rules is often preferable than a larger set at the same level of accuracy in terms of rule understandability.

Moreover, in the process of frequent itemsets generation, the Apriori-type method usually considers all the combinations of items in candidate itemsets. With massive datasets, the number of these combinations is generally very large. In fact, different items in these combinations may play different roles in measuring the degrees of support and degrees of confidence. Therefore, it is deemed desirable if only a part of the items (e.g., those "informative" ones) in candidate itemsets need to be considered in generating frequent itemsets.

This paper addresses some of the above-mentioned issues and presents a new approach for constructing a classifier, based on an extended association rule mining technique in the context of classification. Section 2 describes the issues of concern along with the notion of information gain to be used in the mining process to reduce the number of candidate itemsets, as well as with the notions and certain related properties of rule redundancy and conflicts. In Section 3, the new mining algorithm called GARC (Gain based Association Rule Classification) is presented, which combines the processes of frequent itemsets generation and rule generation, where the measures for redundancy and conflicts avoidance and information gain are incorporated. Finally, results and respective analyses of data experiments are provided in Section 4.

2. Classification rules

2.1. Basic notions

As mentioned previously, the classification rules mining problem can be regarded as a special case of the association rules mining problem [21,22,27]. The task of classification is to find a set of rules so as to identify the classes of undetermined transactions. In classification, a classifier is usually built based upon a dataset that is divided into two groups: one is for training, and the other for testing, each consisting of data items and class labels. In terms of association rules, these class labels are special cases of items. For the sake of clarity, we hereafter refer to them separately, otherwise indicated where necessary.

Let T be the dataset with each transaction composed of a number of distinctive items in the set of all items $\boldsymbol{I}$ and a class label in $G=\{C_1, C_2, \ldots, C_g\}$, X be a subset of $\boldsymbol{I}$ (i.e., $X \subseteq \boldsymbol{I}$), and C_k be a class label in $G(k=1, 2,\ldots,g)$. Notably, in classification-oriented association rule mining, only those rules each with one single class label as its consequent need to be considered; therefore in this paper, each itemset (such as XC_k) is used to represent a rule (such as $X \Rightarrow C_k$) identically. In other words, itemset XC_k corresponds to rule $X \Rightarrow C_k$, with $D_{\text{supp}}(XC_k)=\|XC_k\|/|T|$, and $D_{\text{conf}}(XC_k)=\|XC_k\|/\|X\|$. A transaction t in T is called to contain X if $t \supseteq X$. XC_k is called to be a p-itemset, if X contains p items. If $D_{\text{supp}}(XC_k) \geq \alpha$, then XC_k is called a frequent itemset. Furthermore, if $D_{\text{conf}}(XC_k) \geq \beta$, then XC_k is called a qualified itemset, and can be used to produce a classification rule such as $X \Rightarrow C_k$. For a rule $X \Rightarrow C_k$, sometimes X is referred to as the antecedent of the rule and C_k as the consequent of the rule. Moreover, for the sake of convenience, two parameters namely *lcount* and *wcount* are sometimes used to denote $\|X\|$ and $\|XC_k\|$, as the number of transactions containing X and the number of transactions containing XC_k, respectively. Thus, mining classification rules is used to discover such qualified association rules as $X \Rightarrow C_k$, for $k=1, 2,\ldots,g$.

2.2. Information gain

Information gain is one of the measures used to select best split attributes in decision tree classifiers [7,35]. In this paper, it could also be used as a measure to reduce the number of itemsets. In the process of frequent itemsets generation, instead of considering all the combinations of items in candidate itemsets in the Apriori-type method, information gain measure will be used to select the best attribute. In this way, only those items containing the best item with maximum information gain need to be selected to further generate candidate itemsets.

Suppose an attribute A has n distinct values that partition the training dataset T into subsets T_1, T_2, ..., T_n. For a dataset $S \subseteq T$, freq(C_k, S) represents the number of transactions in S that belong to class C_k. Then info(S) is defined as follows to measure the average amount of information needed to identify the class of a transaction in S:

$$\text{info}(S) = -\sum_{k=1}^{g} \frac{\text{freq}(C_k, S)}{|S|} \times \log_2\left(\frac{\text{freq}(C_k, S)}{|S|}\right)$$

where $|S|$ is the number of transactions in S and g is the number of classes.

After the dataset T is partitioned in accordance with n values of attribute A, the expected information requirement could be defined as:

$$\text{info}_A(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \times \text{info}(T_i)$$

The information gained by partitioning T according to attribute A is defined as:

$$\text{gain}(A) = \text{info}(T) - \text{info}_A(T)$$

Among all attributes in dataset T, the best split attribute is the one that maximizes the information gain.

2.3. Rule redundancy and conflicts

Though classification rules can be discovered using Apriori-type association rule mining techniques directly, the whole set of classification rules (i.e., rules satisfying α and β) might be poor in quality. First, the number of classification rules may be too large to easily construct classifiers. More seriously, from the viewpoint of classification, there may exist conflicting rules (e.g., $X \Rightarrow C_i$ and $X \Rightarrow C_j$) and redundant rules (e.g., $X \Rightarrow C_i$ and $XY \Rightarrow C_i$, with $D_{\text{conf}}(X \Rightarrow C_i) \geq D_{\text{conf}}(XY \Rightarrow C_i)$). The conflicting rules will lead to identifying a transaction into two classes, while the redundancy will result in a rule like $XY \Rightarrow C_i$ that is semantically meaningless for classification (given $X \Rightarrow C_i$).

Definition 2.1. Rule r is called to precede rule r' if either $D_{\text{conf}}(r) > D_{\text{conf}}(r')$, or $D_{\text{conf}}(r) = D_{\text{conf}}(r')$ and $D_{\text{supp}}(r) > D_{\text{supp}}(r')$.

Definition 2.2. Let Ψ be a set of discovered classification rules. Then rule $Z \Rightarrow C_i$ in Ψ is called redundant if there already exists a rule $X \Rightarrow C_i$ in Ψ such that $Z \supset X$. Moreover, for $i \neq j$, rules $Z \Rightarrow C_j$ and $X \Rightarrow C_i$ in Ψ are called conflicting if there already exists a rule $X \Rightarrow C_i$ in Ψ such that either $Z = X$, or $Z \supset X$ and $Z \Rightarrow C_j$ does not precede $X \Rightarrow C_i$.

Apparently, coping with such rule redundancy and conflicts is desirable because otherwise (1) a transaction containing X may be classified into two classes (e.g., C_i and C_j), (2) a rule (e.g., $XY \Rightarrow C_i$) may not be regarded useful (i.e., redundant) for identifying a transaction due to the existence of another rule (e.g., $X \Rightarrow C_i$), and (3) a transaction containing XY may be classified into two classes (e.g., C_i and C_j); or $XY \Rightarrow C_j$ is not significant enough to be used, compared with $X \Rightarrow C_i$. It is worth mentioning, however, that these notions of redundancy and conflict are particularly relevant for classification and may not be of concern for association rules in general.

Usually, given a (nonempty) set Ψ of discovered classification rules, i.e., $\Psi = \{r | r$ is a classification rule, $D_{\text{supp}}(r) \geq \alpha$ and $D_{\text{conf}}(r) \geq \beta\}$, filters can be built to deal with the redundancy and conflicts. It can be seen that for any nonempty Ψ there exists a corresponding nonempty set Ψ_{c} of rules with such redundancy and conflicts removed. Ψ_{c} is referred to as a compact set of Ψ. A constructive way to obtain Ψ_{c} is a repetitive resolution procedure as follows:

First set $\Psi_{\text{c}} = \Psi$, then repeat the following steps until no further changes are made for Ψ_{c}.

(i) check each rule $XY \Rightarrow C_i$ in Ψ_{c}, if there exists a rule $X \Rightarrow C_i$ in Ψ_{c}, then delete $XY \Rightarrow C_i$. That is, $\Psi_{\text{c}} = \Psi_{\text{c}} - \{XY \Rightarrow C_i\}$.

(ii) check each rule $X \Rightarrow C_i$ in Ψ_{c}, if there exists a rule $X \Rightarrow C_j$ in Ψ_{c} that does not precede $X \Rightarrow C_i$, then delete $X \Rightarrow C_j$. That is, $\Psi_{\text{c}} = \Psi_{\text{c}} - \{X \Rightarrow C_j\}$.

(iii) check each rule $XY \Rightarrow C_j$ in Ψ_{c}, if there exists a rule $X \Rightarrow C_i$ in Ψ_{c} that precedes it, then delete $XY \Rightarrow C_j$. That is, $\Psi_{\text{c}} = \Psi_{\text{c}} - \{XY \Rightarrow C_j\}$.

Obviously, Ψ_{c} is nonempty if Ψ is nonempty. Moreover, Ψ_{c} is not unique, depending on the order in which the above three steps are performed. Notably, in this paper, our primary attention is not paid to developing a separate filter to derive Ψ_{c} from Ψ, but to exploring certain ways to avoid rule conflicts and redundancy, which could be incorporated in the integrated mining process.

2.4. Strategies in avoiding rule conflicts and redundancy

When the strategies are incorporated into the process of generating Ψ, the rules in Ψ will be free of redundancy and conflict. More importantly, these strategies will help identify excluded (i.e., not frequent) itemsets inside the process of candidate itemsets generation, resulting in fewer itemsets to be generated.

With regard to the redundancy stated in Definition 2.2, the strategy that could be applied is that, if $X \Rightarrow C_i$ holds, then any candidate itemset containing XC_i need not to be produced, because any such itemsets as XYC_i (i.e., $Z = XY$) would not be regarded semantically necessary from the perspective of classification. In other words, if $X \Rightarrow C_i$ holds, it means that any transaction containing X will be classified into class C_i, including any transaction containing XY. Note that the number of candidate itemsets to generate is reduced.

Next, consider rule conflicts for $Z = X$ in Definition 2.2. The following theorem indicates that it can simply be avoided if the pre-specified minimal confidence β is set to be over 0.5, which is regarded reasonable in many real applications.

Theorem 2.1. *If $\beta > 50\%$, then rules $X \Rightarrow C_i$ and $X \Rightarrow C_j$ will not hold in T simultaneously.*

Proof. Without loss of generality, assume that $X \Rightarrow C_i$ holds in T with $D_{\text{conf}}(X \Rightarrow C_i) \geq \beta > 50\%$. For $g \geq 2$, since $\| X \| = \| XC_i \| + \sum_{j=1, j \neq i}^{g} \| XC_j \|$, and $\frac{\| X \|}{\| X \|} = \frac{\| XC_i \|}{\| X \|} + \sum_{j=1, j \neq i}^{g} \frac{\| XC_j \|}{\| X \|}$, then

$$\sum_{\substack{j=1 \\ j \neq i}}^{g} \frac{\| XC_j \|}{\| X \|} = 1 - D_{\text{conf}}(X \Rightarrow C_i).$$

From $\frac{\| XC_j \|}{\| X \|} \geq 0$, for j=1, 2,…,$g$, we have $\frac{\| XC_j \|}{\| X \|} \leq 1 - Dconf(X \rightarrow C_i) < 50\% < \beta$, which means that $X \Rightarrow C_j$ does not hold in T. □

In other words, if both $X \Rightarrow C_i$ and $X \Rightarrow C_j$ hold simultaneously, both of them must involve two mutually disjoint sets of transactions (denoted as T_{XC_i} and T_{XC_j}) containing XC_i and XC_j respectively. Otherwise, if a transaction t is involved in generating both rules, we will have $XC_iC_j \subseteq t$, which is a contradiction to the structure of t, for t contains only a single class label of G$=\{C_1, C_2, \ldots, C_g\}$. That is, $T_{XC_i} \cap T_{XC_j} = \emptyset$. Since these two sets are the subsets of T_X (where T_X is the set of transactions containing X), we have $|T_{XC_i}|/|T_X| + |T_{XC_j}|/|T_X| \leq |T_X|/|T_X| = 1$. Semantically, the following relationship exists: $\|X_{C_i}\|/\|X\| + \|X_{C_j}\|/\|X\| \leq \|X\|/\|X\| = 1$, which means that $D_{\text{conf}}(X \Rightarrow C_i) + D_{\text{conf}}(X \Rightarrow C_j) \leq 1$. Apparently, however, this is a contradiction to the supposition that $D_{\text{conf}}(X \Rightarrow C_j) \leq \beta > 0.5$ and $D_{\text{conf}}(X \Rightarrow C_i) \geq \beta > 0.5$. In brief, the strategy to set β to be over 0.5 will prevent the rule conflict from happening.

In addition, the following theorem can be used to further reduce the number of candidate itemsets. This also corresponds to rule conflicts stated in Definition 2.2. It will be proved in Theorem 2.2 that, if $X \Rightarrow C_i$ holds in T and if $D_{\text{conf}}(X \Rightarrow C_i) > 1 - \alpha$ or $D_{\text{supp}}(X) < 2\alpha$, then $XY \Rightarrow C_j$ does not hold in T.

Theorem 2.2. *Suppose rule $X \Rightarrow C_i$ holds in T, (1) if $1 - D_{conf}(X \Rightarrow C_i) < \alpha$, then any itemset like XYC_j is an excluded itemset; (2) if $||X|| < 2|T|\alpha$, then any itemset like XYC_j is an excluded itemset; where $Y \cap X = \emptyset$, $Y \cap C_k = \emptyset$, k= 1, 2,…, g, and $g \geq 2$.*

Proof. (1) Since $\|X\| = \| XC_i \| + \sum_{j=1, j \neq i}^{g} \| XC_j \|$, then $\frac{\| XC_j \|}{\| X \|} \leq 1 - D_{\text{conf}}(X \Rightarrow C_i)$. □

Further, $\frac{\| XYC_j \|}{\| X \|} \leq \frac{\| XC_j \|}{\| X \|} \leq 1 - D_{\text{conf}}(X \Rightarrow C_i)$, *thus* $\frac{\| XYC_j \|}{| T |} \leq \frac{\| XYC_j \|}{\| X \|} \leq 1 - D_{\text{conf}}(X \Rightarrow C_i)$.

Then since $1 - D_{\text{conf}}(X \Rightarrow Ci) < \alpha$, then $\frac{\| XYC_j \|}{| T |} < \alpha$, which means $D_{\text{supp}}(XYC_j) < \alpha$. That is, XYC_j is an excluded itemset.

(2) Since $\|X\| < 2|T|\alpha$, then $\frac{\| X \| - \| XC_i \|}{|T|} < \frac{2|T|\alpha - \| XC_i \|}{|T|}$.

From $\frac{\| XC_i \|}{|T|} \geq \alpha$, we have $\|XC_i\| \geq |T|\alpha$.

Then $\frac{\| X \| - \| XC_i \|}{| T |} < \frac{2|T|\alpha - |T|\alpha}{| T |} = \alpha$.

Since $\frac{\| XYC_j \|}{|T|} \leq \frac{\| X \| - \| XC_i \|}{|T|}$, Dsupp$(XYC_j) < \alpha$.

That is, XYC_j is an excluded itemset.

Thus, if rule $X \Rightarrow C_i$ holds in T, and the conditions of Theorem 2.2 are satisfied, then the rule conflicts can be avoided, because hereby any itemset like XYC_j is an excluded itemset. Accordingly, this strategy may be applied to the mining process, in which the itemset, XYC_j, does not need to be considered further in generating larger candidate itemsets.

Example 1. Given a dataset as shown in Table 1, with $\beta = 0.8$ and $\alpha = 0.21$. After the first scan of the dataset, rule "overcast⇒play ($D_{\text{conf}} = 1$, $D_{\text{supp}} = 0.29$)" can be obtained. Before executing the second scan, it has been already known that any larger candidate itemset such as {overcast, Y, don't play} is an excluded itemset, because $1 - 1 < \alpha$ according to Theorem 2.2, where $Y \subseteq$ {Temperature, Humidity, Windy}.

Table 1
Training dataset

TID	Outlook	Temperature	Humidity	Windy	Class
1	Sunny	Mild	Normal	True	Play
2	Sunny	Hot	High	True	Don't play
3	Sunny	Hot	High	False	Don't play
4	Sunny	Mild	High	False	Don't play
5	Sunny	Cool	Normal	False	Play
6	Overcast	Mild	High	True	Play
7	Overcast	Hot	High	False	Play
8	Overcast	Cool	Normal	True	Play
9	Overcast	Hot	Normal	False	Play
10	Rain	Mild	High	True	Don't play
11	Rain	Cool	Normal	True	Don't play
12	Rain	Mild	High	False	Play
13	Rain	Cool	High	False	Play
14	Rain	Mild	High	False	Play
15	Overcast	Cool	High	False	Don't play

If another transaction as follows is added to Table 1:

15	Overcast	Cool	high	False	Don't Play

then D_{conf} (overcast⇒play)=0.8, so we have $1-D_{\text{conf}}$ (overcast⇒play)$=1-0.8=0.2<0.21$. Likewise, {overcast, *Y*, don't play} is an excluded itemset. Suppose that the class label of transaction 9 in Table 1 is changed to be *Don't play*, with $\beta=0.7$ and $\alpha=0.15$. Then from ||overcast||$=4<2|T|\alpha=2\times 14\times 0.15=4.2$ (Theorem 2.2), itemsets {overcast, play} and {overcast, don't play} can be excluded from the itemsets used to generate larger candidate itemsets.

3. Discovering classification rules

In this section, an algorithm called GARC (Gain based Association Rule Classification) will be presented, which could discover the compact set of classification rules. Though the general idea is in the spirit of association rule mining, it differs from conventional CBA techniques that directly apply the Apriori-type association rule mining procedures. The main characteristic of the proposed algorithm is threefold. First, it combines the conventional itemset generation and rule generation processes, and makes use of the information maintained for both rule itemsets and excluded itemsets. Second, the information gain measure is incorporated so as to only generate the itemsets including the best-split attribute value, which leads to a reduction of candidate itemsets. Third, certain strategies are applied into the mining process such that conflicting/redundant rules are avoided as well as the number of candidate itemsets generated is reduced. As a result, the resultant compact set is more condensed and understandable (in terms of fewer rules), and in the mean time, as revealed by data experiments in the next sections, the classification accuracy turns out to be satisfactory.

3.1. GARC: gain based association rule classification

Generally speaking, one transaction in T with s items can generate around 2^s candidate itemsets. To cope with this, the information gain measure is first used here to reduce the number of candidate itemsets. That is, only those candidate itemsets including the best split attribute value will be generated. Concretely, after the first scan of the database, all of 1-itemsets can be obtained and saved in a variable named *Cand*. According to the *lcount* and *wcount* values of each candidate itemset, information gain for each attribute A, which could be used to partition the database T into n datasets, may be calculated as follows:

$$\begin{aligned}\text{info}_A(T) &= \sum_{i=1}^{n}\frac{|T_i|}{|T|}\times\text{info}(T_i) = -\sum_{i=1}^{n}\frac{|T_i|}{|T|}\\ &\quad\times\left(\sum_{k=1}^{g}\frac{\text{freq}(C_j,T_i)}{|T_i|}\times\log_2\frac{\text{freq}(C_j,T_i)}{|T_i|}\right)\\ &= -\sum_{i=1}^{n}D_{\text{supp}}(A=v_i)\\ &\quad\times\left(\sum_{k=1}^{g}Dc_{\text{conf}}(i_k)\times\log_2 D_{\text{conf}}(i_k)\right)\end{aligned}$$

where T_i corresponds to the dataset whose attribute A's value equals v_i, g is the number of classes, i_k represents itemset $\{v_i, C_k\}$. As a result, a best split attribute (called *bestattr*) can be selected after the first scan of the database. Then during the next scan of the database, only those itemsets containing this best split attribute specified by *bestattr* will be generated. The following example helps illustrate the idea.

Example 2. Let us consider Table 1 again. After the first scan of the dataset in Table 1, among the four attributes, attribute *outlook* is selected as the best split attribute. Then during the second scan of the database, tuple 1 can produce the following three candidate itemsets: {sunny, mild, play}, {sunny, normal, play}, and {sunny, true, play}. Note that {mild, normal, play}, {mild, true, play}, and {normal, true, play} will not be generated.

In addition to information gain, certain conflicts/redundancy avoidance strategies are used to improve the quality of the rule set as well as to reduce the number of candidate itemsets, which is detailed in the next subsection, along with how excluded itemsets are dealt with.

3.2. Algorithmic details

As stated previously, an itemset *XC* is interchangeably referred to as a rule *X*⇒*C*. A qualified itemset

Table 2
Algorithm GARC

```
Algorithm GARC:
1.  $rule=\{r|r$ is an 1-itemset, $D_{supp}(r)\geq\alpha$ and $D_{conf}(r)\geq\beta\}$;
    //initiating the set of qualified itemsets//
2.  $excluded=\{e|e$ is an 1-itemset, and $D_{supp}(e)<\alpha\}$; //initiating
    the set of excluded itemsets//
3.  $bestattr=gain$;
4.  if $(\beta\leq 0.5)$ and $(\forall r{:}X\Rightarrow C_i\in rule,\ \exists r'{:}X\Rightarrow C_j\in rule$
    such that $r'$ does not precede $r$) then
5.      $rule=rule-\{X\Rightarrow C_j\}$;          //deleting conflicting rules//
6.  for $k$ from 2 to $m$ do //$m$ is the number of ancetedent
    attributes//
7.      $empty(cand)$; // emptying $cand$, the set of candidate
        itemsets//
8.      if $coverall(rule)$
9.          break;
10.     for each transaction $t$ in $T$ do
11.         $C_t=CandidateGen(t,\ bestattr,\ k)$;
12.         for each $c\in C_t$do
13.             $maintCand(rule,\ excluded,\ c,\ cand)$
14.         end for;
15.     end for;
16.     $R=\{r|r\in cand,\ D_{supp}(r)\geq\alpha$ and $D_{conf}(r)\geq\beta\}$;
        //the set of qualified $k$-itemsets//
17.     if $(\beta\leq 0.5)$ and $(\forall r{:}X\underset{i}{\Rightarrow}C\ \in R,\ \exists r'{:}X\Rightarrow C_j\in R$
        such that $r'$ does not precede $r$) then
18.         $R=R-\{X\Rightarrow C_j\}$;          //deleting conflicting rules//
19.     if $(\forall r{:}X\Rightarrow C_i\in R,\ \exists r'{:}XY\Rightarrow C_j\in R$ such that $Y\neq\emptyset$,
        $X\cap Y=\emptyset$, and $r'$ does not precede $r$) then
20.         $R=R-\{XY\Rightarrow C_j\}$;          //deleting conflicting rules//
21.     $rule=rule\cup R$;
22.     $E=\{e|e\in cand,\ D_{supp}(e)<\alpha\}$;
23.     $excluded=excluded\cup E$;
24. end for;
25. $sort(rule)$;
```

XC corresponds to a qualified rule $X\Rightarrow C$. In addition, for XC, X is referred to the antecedent of XC, denoted by antecedent(XC)=X. The main algorithm is shown in Table 2.

Lines 1–3 perform the first scan of the database. It produces all the 1-itemsets from which qualified 1-itemsets are generated. By the method described above, the best split attribute is selected by *gain*, which employs the information gain measure and helps reduce the number of candidate itemsets (Table 3). Lines 6–24 perform the consecutive scans of the database. *coverall(rule)* tests whether rules already contain all of transactions in the training dataset, and if true, the main iteration breaks. During each scan, for a certain transaction *t*, *CandidateGen(t, bestattr, k)* generates all *k*-itemsets with each containing the *bestattr*. Working on these itemsets in *Ct, maintCand(rule, excluded, c, cand)* then generates and maintains candidate itemsets according to qualified itemsets and excluded itemsets. Note that the itemsets returned by *maintCand* will be redundancy-free, and will not produce any conflicting rules if the conditions of Theorem 2.2 are satisfied. In the mean time, this will lead to generating fewer itemsets inside the process. Moreover, since rule conflicts with regard to Theorem 2.1 will be avoided if its condition ($\beta>0.5$) is satisfied, lines 4–5 and 17–20 further remove conflicting rules (r') when the conditions of Theorems 2.1 and 2.2 do not hold. The advantage of incorporating the conflict resolution strategy at the stages inside the *k*-itemset generation process (rather than after the process as a separate filter) is to further reduce the number of candidate itemsets generated (for $k\geq 1$). Finally, all rules are sorted, when the main procedure is terminated.

Note that each of the rules included in the classifier built by the above algorithm satisfies the pre-specified minimal support and minimal confidence thresholds (i.e., (α and β). Moreover, the rules that cannot be predicated to be excluded itemsets according to Theorem 2.2 (line 6 in Table 4) will be added to the set of candidate itemsets and further counted. If *c* contains a qualified itemset or an excluded itemset, the class attribute in *c* is substituted by a fixed mark *q* that is different from all class labels of *G* (i.e., *G* is the set of all class labels) in the database for the purpose of counting *lcount* while not affecting *wcount* (Tables 4 and 5). In addition,

Table 3
Sub-algorithm *gain*

```
gain
begin
    for each attribute $A_i\in\{A_1,A_2\ldots A_m\}$ do
        compute info($A_i$) using $D_{supp}$ and $D_{conf}$ values of all 1-itemsets
    end for
    $bestattr=A_1$; $mininfo=info(A_1)$;
    for each attribute $A_i\in\{A_1,A_2\ldots A_m\}$ do
        if $mininfo>info(A_i)$ then
            $mininfo=info(A_i)$
            $bestattr=A_i$
        end if
    end for
    return $bestattr$
end
```

Table 4
Sub-algorithm *maintCand*

```
maintCand(rule, excluded, c, cand)
1. begin
2.   if $\neg\exists r \in$ rule, $(c \supset r)$ and $\neg\exists e \in excluded$, $(c \supset e)$ then
3.     if $\neg\exists r \in$ rule, $(c \supset ancetedent(r))$ then
4.       addToCand(c, cand); //adding c into cand//
5.     else
6.       if $\exists r \in rule$, $(c \supset ancetedent(r))$ and
         $((1-D_{conf}(r) \geq \alpha)$ and $(D_{supp}(X) \geq 2\alpha)$ then
7.         addToCand(c, cand);
8.       else //when c is an excluded itemset//
9.         $excluded = excluded \cup E$;
10.      end if;
11.    end if;
12.  else
13.    if $\exists e \in excluded$, $(c \supset e)$ or $\exists r \in rule$, $(c \supset r)$ then
14.      $c' = ancetedent(e) \cup \{q\}$; //q is a fixed mark different
         from any class in G//
15.      addToCand(c', cand); //adding c' into cand//
16.    end if;
17.  end if;
18. end;
```

the rules that are redundant according to Definition 2.2 will not be included (generated) in *cand* (line 13 in Table 4, and line 14 in Table 5).

Finally, the algorithm will terminate in a finite number of m passes at most, where m is the number of attributes. Notably, the set of resultant classification rules is a compact set. Moreover, if the discovered rule set without using the redundancy/conflict resolution strategies is not empty (e.g., if the set of qualified 1-itemsets is not empty), the compact set will not be empty either.

4. Experimental results

This section shows an empirical performance evaluation of algorithm GARC, along with some comparisons with other algorithms. The experiments consist of five parts. The first part is to compare GARC with C4.5-type [30], CBA [21], NN [10], and SVM [37] classifiers on accuracy. The second part of the experiments is to test how the pruning strategies affect the efficiency and further examines the execution time of GARC. The third part discusses the impact of minimal support and minimal confidence thresholds on GARC outcomes. In the fourth part, the use of information gain for rule reduction is examined. The last part compares GARC with the CBA classifier in terms of the number of rules produced. The experiments were conducted in the environment with Windows 2000 Server, Intel Pentium 4 1.5 GHz, 512 MB RAM and Visual C++. It should be mentioned that, all the following experiments are tested based on datasets from a commonly used benchmarking database in the field, namely the UCI Machine Learning Repository [27], including the 26 datasets that CBA method selected. In total, 30 datasets are used.

The basic information of the datasets is listed in Table 6.

Since some data are continuous and Apriori-type methods mainly focus on discrete data, the entropy based discretization method is applied in order to deal with continuous attributes for the experiments. More concretely, a recursive entropy minimization heuristic is used for discretization and combined with the Minimum Description Length criterion to control the number of intervals produced over a continuous space [15].

4.1. Accuracy

Accuracy is one of the basic performance measures for classification algorithms. For a classifier,

Table 5
Sub-algorithm *addToCand*

```
addToCand(c, cand);
1. begin
2.   find=0; count=1;
3.     for each candidate itemset $c_i$ in cand do
4.     if $c = c_i$ then
5.       $c_i$.wcount= $c_i$.wcount+1;
6.       $c_i$.lcount= $c_i$.lcount+1;
7.       find = 1;
8.     else
9.       if ancetedent(c) = ancetedent($c_i$) then
10.      $c_i$.lcount = $c_i$.lcount+1;
11.      count = count + $c_i$.lcount;
12.    end if;
13.  end for;
14.  if find = 0 and (consequent of c is not equal to q) then
     //when c is not redundant//
15.    c is included in cand with c.lcount=count and
       c.wcount=1;
16.  end if;
17. end;
```

Table 6
Basic information of the 30 UCI datasets

	Dataset	Attributes	Number of attributes	Null value (Y/N)	Number of training data	Number of testing data
1	Anneal	Discrete, continuous	38	Y	598	300
2	Australian	Discrete, continuous	14	N	460	230
3	Auto	Discrete, continuous	26	Y	136	69
4	Breast	Continuous	10	Y	466	233
5	Cleve	Discrete, continuous	13	N	202	101
6	Crx	Discrete, continuous	15	N	490	200
7	Diabetes	Continuous	8	N	512	256
8	German	Discrete, continuous	20	N	666	33
9	Glass	Continuous	9	N	142	72
10	Heart	Continuous	13	N	180	90
11	Hepatitis	Discrete, continuous	19	Y	103	52
12	Horse	Discrete, continuous	22	Y	300	68
13	Hypothyroid	Discrete, continuous	29	Y	2514	1258
14	Ionosphere	Continuous	34	Y	234	117
15	Iris	Continuous	4	N	100	50
16	Labor	Discrete, continuous	16	Y	40	17
17	Led7	Discrete	7	N	200	3000
18	Lymph	Discrete	18	N	98	50
19	Pima	Continuous	8	N	512	256
20	Sick	Discrete, continuous	29	Y	2800	972
21	Sonar	Continuous	60	N	138	70
22	Tic-tac-toe	Discrete	9	N	638	320
23	Vehicle	Continuous	18	N	564	282
24	Waveform	Continuous	21	N	300	1000
25	Wine	Continuous	13	N	118	60
26	Zoo	Discrete	16	N	67	34
27	Balance	Continuous	4	N	416	209
28	Lenses	Discrete	4	N	16	8
29	Monk2	Discrete	6	N	169	432
30	Vote	Discrete	16	N	300	135

its classification accuracy is the ratio of the number of cases truly predicted by the classifier over the total number of cases in the test dataset, e.g.,

$$\text{Accuracy} = \frac{\text{num(test\_predicted} = \text{true)}}{\text{num\_totaltest}} \times 100\%.$$

In this experiment, we compared GARC with C4.5 rule, CBA, NN and SVM classifiers based on the 30 UCI datasets. We obtained the accuracy results as shown in Tables 7, 8. It will be discussed in later subsections on how the settings of the thresholds are considered.

The results shown in Tables 7 and 8 indicate that the classification accuracy of GARC is satisfactory. On average, the accuracy of GARC seemed to be higher than that of the C4.5 rule and similar to that of CBA. Moreover the GARC classifier appeared to be more stable than CBA and C4.5 rule classifiers in terms of standard deviations of accuracy. These findings could be further justified by statistical significance tests. Moreover, Table 8 shows that the accuracy of GARC is lower than that of NN or SVM, and that the standard deviation of GARC is lower than that of NN and SVM. However, it is important to note that GARC, NN, SVM are not significantly different in accuracy.

Thus, we could test the significance of the accuracy mean difference for any two algorithms by approximately constructing a confidence interval at a given confidence level [6,18]. The testing results revealed that on average the accuracy of GARC was

Table 7
Algorithms' accuracy on C4.5, CBA and GARC

	Datasets	C4.5%	CBA%	GARC%
1	Anneal	88.70	98.00	89.30
2	Australian	87.00	86.96	87.39
3	Auto	62.70	72.46	71.32
4	Breast	95.70	96.57	94.85
5	Cleve	77.20	81.19	80.13
6	Crx	83.00	83.50	82.50
7	Diabetes	69.10	74.22	71.03
8	German	73.40	76.35	75.20
9	Glass	62.50	65.28	68.06
10	Heart	83.30	83.33	80.57
11	Hepatitis	80.80	76.92	86.69
12	Horse	85.30	80.88	75.00
13	Hypothyroid	99.20	98.20	94.79
14	Ionosphere	88.00	93.16	90.64
15	Iris	92.00	94.00	94.01
16	Labor	82.40	88.24	82.35
17	Led7	67.50	57.67	56.53
18	Lymph	70.00	84.00	77.56
19	Pima	76.60	76.17	73.83
20	Sick	99.00	96.50	93.83
21	Sonar	74.30	64.29	74.30
22	Tic-tac-toe	82.20	99.06	100.00
23	Vehicle	67.70	70.21	61.89
24	Waveform	70.40	75.66	71.15
25	Wine	85.00	86.67	83.46
26	Zoo	85.30	79.41	82.35
27	Balance	77.50	72.73	71.29
28	Lenses	62.50	62.50	75.32
29	Monk2	65.00	67.13	65.74
30	Vote	97.00	95.56	89.67
	Mean	79.68	81.23	80.03
	Derivation	1.23	1.40	1.15
	Standard deviation	11.09	11.84	10.72

GARC is running with default setting of $\alpha=0.01$, $\beta=0.7$.

not significantly different from that of CBA, C4.5, NN or SVM. These have been shown in Table 9.

In addition, two C4.5 extensions, namely C4.5 tree and C4.5 tree pruning [30], were tested on the same 30 datasets by means of confidence intervals, revealing that the accuracy of GARC was not significantly different from that of either C4.5 tree or C4.5 tree pruning at 95% confidence level. Furthermore, our test on the 30 datasets is, however, not supportive to the statement in Ref. [21] that the accuracy of the C4.5 rule is higher than that of either C4.5 tree or C4.5 tree pruning.

In summary, GARC is satisfactory in terms of accuracy, compared with CBA, C4.5-type, NN and SVM. Worthwhile to mention is that, compared with non-rule-based classifiers (e.g., NN and SVM), GARC produces a classifier in the form of explicit rules, which are often appealing for use and explanation to decision makers.

Table 8
Algorithms' accuracy on SVM, NN and GARC

	Datasets	SVM%	NN%	GARC%
1	Anneal	100.00	98.67	96.67
2	Australian	86.96	90.00	87.39
3	Auto	72.46	63.77	71.00
4	Breast	96.14	94.85	95.71
5	Cleve	82.18	81.19	81.19
6	Crx	82.50	85.00	85.50
7	Diabetes	73.83	78.13	74.61
8	German	71.86	75.15	76.35
9	Glass	79.17	73.61	68.06
10	Heart	88.88	87.78	88.00
11	Hepatitis	84.62	78.85	86.54
12	Horse	86.76	80.88	88.24
13	Hypothyroid	100.00	98.01	94.79
14	Ionosphere	96.58	94.87	94.85
15	Iris	94.00	96.00	96.00
16	Labor	100.00	94.12	82.35
17	Led7	68.97	67.73	67.47
18	Lymph	82.00	86.00	80.00
19	Pima	79.69	78.52	76.17
20	Sick	96.71	97.02	93.83
21	Sonar	88.57	78.57	74.30
22	Tic-tac-toe	99.38	97.50	100.00
23	Vehicle	79.08	78.37	67.36
24	Waveform	80.85	81.06	71.55
25	Wine	98.33	95.00	86.67
26	Zoo	88.24	88.24	85.29
27	Balance	99.04	92.34	73.21
28	Lenses	62.50	75.00	87.50
29	Monk2	84.72	100.00	74.54
30	Vote	97.78	99.26	96.30
	Mean	86.73	86.18	83.38
	Deviation	1.11	1.03	1.00
	Standard deviation	10.52	10.13	10.01

SVM classifier use LIBSVM software package available online at http://www.csie.ntu.edu.tw/~cjlin/libsvm Ref. [36]. For SVM, the parameters will largely affect the results. With default settings by LIBSVM, the average accuracy is 79.84. After parameters selection with 10-fold cross validation, SVM is running on the best situation.
NN classifier is using WEKA software package [35]. A 3-level Back-Propagation Model has been constructed, with the number of neurons set to 2, 4 and 8. The results are not sensitive to the number of neurons. NN is also running on the best situation. GARC is running on the best situation with corresponding α and β.

Table 9
Confidence intervals on the mean difference for accuracy of classifiers

	Confidence level %	Interval%	Significance
GARC–CBA	95	[−6.92,4.51]	No
	90	[−6.00,3.59]	No
GARC–C4.5 rule	95	[−5.17,5.87]	No
	90	[−4.28,4.98]	No
GARC–NN	95	[−8.71,1.74]	No
	90	[−7.87,0.90]	No
GARC–SVM	95	[−9.35,1.30]	No
	90	[−8.49,0.44]	No

4.2. GARC with pruning strategies

This subsection examines GARC's pruning strategy effectiveness in terms of accuracy, understandability (e.g., the number of rules generated) and computational efficiency (e.g., the number of candidate itemsets generated, execution time, etc.). By a pruning strategy we mean the strategy discussed in Section 2.4 and incorporated in the mining process. Since we are to study the GARC performance with and without the strategy incorporation, the dataset used needs to be expansible and adjustable in size and complexity. Apparently, the previously used 30 datasets can hardly serve this purpose. Hence, as proposed in Ref. [3] for similar experiments, a synthetic database is employed. Each transaction in the database has 9 attributes shown in Table 10. There are ten classification functions available to produce data distributions with varied complexities. IBM Research Center developed a series of classification functions of increasing complexity that used the above attributes to classify people into different groups [19]. Four of them are selected, which include the low-complexity (function 2), mid-complexity (function 5 and 8) and the most complex function 10. Specifically, function 10 is one of the hardest to characterize and could result in the highest classification errors (Table 11). The Data generator source is from Ref. [19].

Since GARC works with categorical attributes, the non-categorical attributes were discretized first. We used a simple equal-width method for discretization. The interval width and the number of intervals are shown in Table 12.

The performances of GARC with and without those (pruning) strategies proposed in Section 2 are shown in Fig. 1. The findings indicated that GARC with the strategies was superior to that without the strategies in three respects, namely, computational efficiency (fewer candidate itemsets and shorter execution time as shown in Fig. 1a,b), understandability (fewer rules

Table 10
Attributes of the synthetic database

Attribute	Value
Salary	Uniformly distributed from 20 000 to 150 000
Commission	If salary ≥ 75 000, commission = 0 else uniformly distributed from 10 000 to 75 000
Age	Uniformly distributed from 20 to 80
Ed_level	Uniformly chosen from 0 to 4
Car	Make of the car, uniformly chosen from 1 to 20
Zipcode	Uniformly chosen from 9 available zipcodes
Housevalue	Uniformly distributed from 0.5*k*100 000 to 1.5*k*100 000, where $0 \le k \le 9$ and depends on the zipcodes
Years owned	Uniformly distributed from 1 to 30
Loan	Uniformly distributed from 0 to 500 000

Table 11
Functions' definitions

Function 2:
Class A: ((age<40) ∧ (50k≤salary≤100k))∨
((40≤age<60) ∧ (75k≤salary≤125k))∨
((age60) ∧ (25k≤salary≤75k)).
Function 5:
Class A: ((age<40) ∧
(((50k≤salary≤100k)) ? (100k≤loan≤300k) :
(200k≤loan≤400k))))∨
((40≤age<60)∧
(((75k≤salary≤125k)) ? (200k≤loan≤400k) :∨
(300k≤loan≤500k))))∨
((age≥60)∧
(((25k≤salary≤75k)) ? (300k≤loan≤500k) :
(100k≤loan≤300k))))
Function 8:
disposable = (0.67 × (salary + commission) – 5000 × elevel –20k)
Class A: disposable >0
Function 10
hyears < 20⇒ equity=0
hyears ≥ 20⇒ equity=0.1 hvalue (hyeares 20)
disposable = (0.67 (salary+commission) 5000 × elevel + 0.2 equity –10k)
Class A: disposable >0

* A ? B : C stands for a logic expression meaning that if A is TRUE then B, else C.

Table 12
Discretization of the attribute values

Attribute	Interval width	No. of intervals
Salary	25,000	6
Commission	10,000	7
Age	10	6
Ed_level	–	5
Car	–	20
Zipcode	–	9
Housevalue	100,000	14
Years owned	3	10
Loan	50,000	10

as shown in Fig. 1c), and accuracy (similar rates as shown in Fig. 1d).

Further, Fig. 2 illustrates the execution time of GARC (e.g., for function 10) with the number of training samples increasing from 100,000 to 500,000, showing a near-linear computational complexity in time. This was also done for CBA and resulted in almost the same outcome.

4.3. Settings of minimal support and minimal confidence

As mentioned in previous subsections, the experiments were conducted with a setting of $\alpha=0.01$ and $\beta=0.7$ for min-support and min-confidence. In this section, we will discuss further the impact of such settings on the accuracy of GARC. With each of the same datasets, we could obtain the best setting of α and β in yielding the highest accuracy. Obviously, the best setting for one dataset is generally different from that for another dataset. To determine a single setting to be used in comparison for 30 datasets, we chose $\alpha=0.01$ and $\beta=0.7$, as they appeared quite often (e.g., about 19 times out of 30 for $\alpha=0.01$ and 14/30 for $\beta=0.7$). Table 13 shows the details.

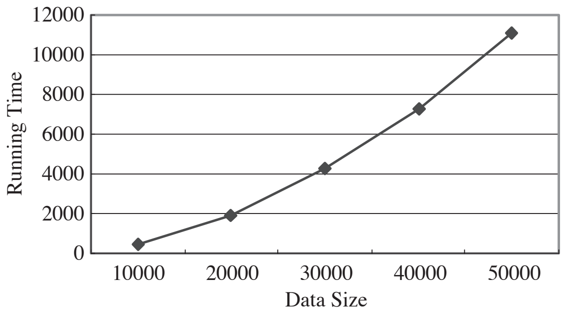

Fig. 2. Running time vs. data size.

Moreover, our experiments showed that as min-confidence increases, the accuracy would increase first then decrease. This may be because when min-confidence is too low, many useless rules will be generated, which will disturb the classifier. On the other hand, when min-confidence is too high, many actually meaningful rules will not be discovered, which will lead many transactions to being classified into the default class, resulting in lower accuracy. Generally, it could be found that the best performance of accuracy is around the situation where $\alpha=0.01$ and $\beta=0.7$.

4.4. Impact of the information gain measure in GARC

As discussed previously, GARC uses information gain to retrieve the best split attribute. In this section, some experimental results are given to show how this measure affects the accuracy and efficiency of GARC.

	With Strategies	**Without Strategies**
Function 2	21	228
Function 5	21	336
Function 8	87	144
Function 10	29	249

(a) Number of Candidate Itemsets

	With Strategies	**Without Strategies**
Function 2	13.24	179.10
Function 5	21	190
Function 8	14.74	16.25
Function 10	11.21	182.39

(b) Execution Time (sec.)

	With Strategies	**Without Strategies**
Function 2	5179	22231
Function 5	10302	26951
Function 8	11056	12000
Function 10	8435	28059

(c) Number of Rules

	With Strategies	**Without Strategies**
Function 2	0.84	0.83
Function 5	0.81	0.85
Function 8	0.90	0.90
Function 10	0.84	0.86

(d) Accuracy

Fig. 1. Performances of GARC with and without strategies.

Table 13
Settings of α and β vs. accuracy

	Highest accuracy			Highest accuracy at $\alpha=0.01$		Accuracy at $\alpha=0.01$, $\beta=0.7$ (%)
	%	α	β	%	β	
Anneal	96.67	0.01	0.95	96.67	0.95	89.33
Australian	87.39	0.01	0.7	87.39	0.7	87.39
Auto	71	0.01	0.7	71	0.7	71.07
Balance	73.21	0.01	0.85	73.21	0.85	71.29
Breast	95.71	0.01	0.95	95.71	0.95	94.85
Cleve	81.19	0.01	0.9	81.19	0.9	80.2
Crx	85.5	0.05	0.9	84.5	0.9	82.5
Diabetes	74.61	0.01	0.85	74.61	0.85	71.48
German	76.35	0.01	0.75	76.35	0.75	76.05
Glass	68.06	0.01	0.7	68.06	0.7	68.06
Heart	88	0.01	0.5	88	0.5	81.11
Hepatitis	86.54	0.01	0.7	86.54	0.7	86.54
Horse	88.24	0.01	0.85	88.24	0.85	75
Hypo	94.79	0.01	0.7	94.79	0.7	94.79
Ionosphere	94.87	0.01	0.95	94.87	0.95	91.45
Iris	96	0.01	1	96	1	94
Labor	82.35	0.01	0.7	82.35	0.7	82.35
Led7	67.47	0.02	0.5	66.33	0.5	57
Lenses	87.5	0.07	0.8	75	0.7	75
Lymph	80	0.02	0.8	78	0.8	78
Monk2	74.54	0.01	0.95	74.54	0.95	67.13
Pima	76.17	0.01	0.85	76.17	0.85	73.83
Sick	93.83	0.01	0.7	93.83	0.7	93.83
Sonar	74.3	0.01	0.7	74.3	0.7	74.3
Tic-tac-toe	100	0.01	0.7	100	0.7	100
Vehicle	67.36	0.01	0.7	67.36	0.7	67.36
Vote	96.3	0.01	0.95	96.3	0.95	89.67
Waveform	71.55	0.02	0.7	69.51	0.9	71
Wine	86.67	0.09	0.7	83.33	0.7	83.33
Zoo	85.29	0.05	0.95	82.35	0.7	82.35

Table 14
Accuracy by using information gain and not using information gain

Datasets	Information gain incorporated	Information gain not incorporated
Anneal	89.33	90
Australian	87.39	87.39
Auto	71.07	65.22
Balance	71.29	72.25
Breast	94.85	94.85
Cleve	80.2	68.32
Crx	82.5	81.5
Diabetes	71.48	67.97
German	76.05	72.55
Glass	68.06	66.67
Heart	81.11	81.11
Hepatitis	86.54	86.54
Horse	75	75
Hypo	94.79	94.79
Ionosphere	91.45	91.45
Iris	94	94
Labor	82.35	82.35
Led7	57	55.8
Lenses	75	75
Lymph	78	78
Monk2	67.13	67.13
Pima	73.83	73.83
Sick	93.83	93.83
Sonar	74.3	67.14
Tic-tac-toe	100	100
Vehicle	67.36	63.83
Vote	89.67	84.44
Waveform	71	71.98
Wine	83.33	83.33
Zoo	82.35	82.35
Mean	80.34	78.95
Standard deviation	10.35	11.29

The results revealed that the average accuracy with information gain was slightly higher than that without information gain. Statistically, the accuracy with information gain is significantly similar to that without information gain at both 95% and 90% confidence levels. These are shown in Tables 14 and 15.

In addition, considering the average number of rules and running times, the results revealed that information gain would lead to much fewer rules and less computational time remarkably. This is largely due to the fact that the number of candidate itemsets has been considerably reduced through introducing information gain in the mining process. On average, the number of rules with information gain was only around 39% of that without the gain, and the execution time with information gain was only around 3.2% of that without information gain, according to the experiment (shownin Table 16).

4.5. GARC and CBA

Though GARC and CBA are all based on association rule mining, they are different: CBA is basically

Table 15
Confidence intervals on the mean difference for accuracy

Information gain–No information gain	Confidence level%	Interval%	Significance
Accuracy	95	[−4.09, 6.87]	No
	90	[−3.21, 5.99]	No

of Apriori-type, whereas GARC is not. The main difference is that GARC combines rule generation with respective frequent itemset generation, making use of excluded itemsets in generating candidate itemsets. In addition, GARC uses information gain to reduce the number of candidate itemsets, which could then reduce the number of rules. Moreover, the number of rules could also be reduced using pruning/resolution strategies such that certain conflicting and redundant rules could be avoided, whereas the CBA algorithm itself will generate more rules. Table 17 tabulates the comparative results based on the 30 benchmarking datasets that were used previously. Clearly, GARC generated much fewer rules than CBA, providing better understandability (at similar levels of accuracy). This can easily be verified by statistical significance tests. On average, the number of rules generated by GARC was only around 4.3% of that by CBA, according to the experiment.

Table 16
Number of rules and running time by using information gain (IG) and not using information gain (NIG)

	Number of rules		Running time (s)	
	IG	NIG	IG	NIG
Anneal	72	85	21.422	7269.953
Australian	17	17	0.125	0.0160
Auto	650	2156	3785.860	116 901.047
Balance	4	10	0.000000001	0.063
Breast	21	25	20.5	237.328
Cleve	23	37	253.564	1578.468
Crx	21	64	2.031	61.812
Diabetes	11	13	25.985	112.078
German	78	188	559.766	21 953.438
Glass	17	21	2.062	12.687
Heart	12	12	0.000000001	0.00000001
Hepatitis	23	23	0.063	0.010
Horse	26	26	0.125	0.016
Hypo	48	48	0.265	0.094
Ionosphere	67	67	0.360	0.160
Iris	7	10	0.000000001	0.000000001
Labor	15	42	0.093	0.266
Led7	33	51	0.234	1.766
Lenses	12	13	0.000000001	0.000000001
Lymph	17	17	0.296	0.100
Monk2	2	2	0.000000001	0.000000001
Pima	6	6	6.016	33.594
Sick	56	56	0.281	0.172
Sonar	16	33	2.765	549.750
Tic-tac-toe	26	26	0.063	0.010
Vehicle	112	543	173.688	4852.828
Vote	32	96	0.937	49.969
Waveform	25	168	0.250	30.078
Wine	16	16	0.093	0.010
Zoo	90	151	2.390	60.313
Mean	51.83	134.07	161.97	5123.53
IG/NIG	39%		3.2%	

Table 17
Number of rules generated by GARC and CBA

	GARC	CBA
Anneal	72	533
Australian	17	1518
Auto	650	4505
Balance	4	147
Breast	21	21
Cleve	23	478
Crx	21	2686
Diabetes	11	40
German	78	1501
Glass	17	32
Heart	12	166
Hepatitis	23	700
Horse	26	988
Hypo	48	1557
Ionosphere	67	2891
Iris	7	14
Labor	15	52
Led7	33	533
Lenses	12	12
Lymph	17	2172
Monk2	2	397
Pima	6	21
Sick	56	1659
Sonar	16	883
Tic-tac-toe	26	200
Vehicle	112	3043
Vote	32	953
Waveform	25	3851
Wine	16	738
Zoo	90	2869
Mean	51.83	1205.33
GARC/CBA	4.3%	

5. Conclusions

Classification is one of the important issues in decision science and knowledge discovery. This paper has presented a new approach to discovering classification rules based on the concept of association rules. In doing so, the corresponding algorithm proposed, namely GARC, has integrated the generation of itemsets and rules, and incorporated information gain and certain conflicts/redundancy resolution strategies into the mining process. Finally, a compact set could be derived. Compared with other classifiers (e.g., CBA, C4.5-type, NN and SVM classifiers), the new approach could achieve a similar level of accuracy. Moreover, the experimental results have shown the advantages of GARC over CBA in terms of number of rules, and over SVM/NN in terms of explicit rules for use and explanation by decision makers. Future studies are centering on explorations of other optimization strategies so as to further improve the mining efficiency.

Acknowledgements

This work was partly supported by the National Natural Science Foundation of China (79925001/70231010), and China's MOE Funds for Doctoral Programs (20020003095).

References

[1] R. Agrawal, R. Srikant, Fast algorithm for mining association rules, Proceeding of the 20th VLDB conference, Morgan Kaufmann, Santiago, Chile, 1994, pp. 487–499.

[2] R. Agrawal, T. Imielinski, A. Swami, Database mining: a performance perspective, IEEE Transaction on Knowledge and Data Engineering 5 (1993) 914–925.

[3] R. Agrawal, T. Imielinski, A. Swami, Mining association rules between sets of items in large databases, Proceeding of 1993 ACM-SIGMOD International Conference on Management of Data, ACM Press, Washington, D.C., 1993, pp. 207–216.

[4] K. Ali, K. Manganaris, R. Srikant, Partial classification using association rules, Proceeding of the Third International Conference on Knowledge Discovery and Data Mining, The AAAI Press, Newport Beach, California, 1997, pp. 115–118.

[5] K. Alsabti, S. Ranka, V. Singh, CLOUDS: a decision tree classifier for large datasets, in: R. Agrawal, P. Stolorz, G. Piatetsky-Shapiro (Eds.), Proceeding of the Fourth Int. Conference on Knowledge Discovery and Data Mining, AAAI Press, Newport Beach, California, 1998, pp. 2–8 (New York, New York).

[6] D. Bertsimas, R.M. Freund, Data, Model, and Decisions: the Fundamentals of Management Science, South-Western College Publishing, 2000.

[7] L. Breiman, Classification and Regression trees, Wadsworth, Belmont, 1984.

[8] S. Brin, R. Motwani, J. Ullman, Dynamic itemset counting and implication rules for market basket data, Proceeding of 1997 ACM-SIGMOD International Conference on Management of Data, ACM Press, Tucson, Arizona, 1997, pp. 255–264.

[9] J. Catlett, Megainduction: Machine Learning on Very Large Databases. PhD thesis, University of Sydney, 1991.

[10] C.C. Chang, C.J. Lin, LIBSVM: a library for support vector machines, 2001.

[11] G.Q. Chen, Q. Wei, Fuzzy association rules and the extended mining algorithms, Information Sciences 147 (2002) 201–228.

[12] G.Q. Chen, Q. Wei, E. Kerre, Fuzzy data mining: discovery of fuzzy generalized association rules, in: G. Bordogna, G. Pasi (Eds.), Recent Research Issues on the Management of Fuzziness in Databases, Physica-Verlag (Springer), 1999.

[13] G.Q. Chen, Q. Wei, D. Liu, G. Wets, Simple association rules (SAR) and the SAR-based rule discovery, Computers and Industrial Engineering 43 (2002) 721–733.

[14] R. Duda, P. Hart, Pattern Classification and Scene Analysis, John Wiley and Sons, 1973.

[15] U.M. Fayyad, K.B. Irani, Multi-interval discretization of continuous-valued attributes for classification learning, Proceedings of the 13th International Joint Conference on Artificial Intelligence, 1993, pp. 1022–1027.

[16] N. Friedman, D. Geiger, M. Goldszmidt, Bayesian network classifier, Machine Learning 29 (1997) 131–163.

[17] T. Fukuda, Y. Morimoto, S. Morishita, Data mining using two-dimensional optimized association rules: scheme, algorithms, and visualization, Proc. of the 1996 ACM-SIGMOD Int'l Conf. on the Management of Data, 1996, pp. 12–13.

[18] Harshbarger,Thad R., Introductory Statistics: A Decision Map, Second edition. The City College, City University of New York, P 376, Macmillan Publishing Co., Inc. New York, Collier Macmillan Publishers, London, 1977.

[19] http://www.almaden.ibm.com/software/quest/Resources/datasets/syndata.html#classSynData.

[20] W. Li, J. Han, J. Pei, CMAR: accurate and efficient classification based on multiple class-association rules, ICDM 2001, IEEE Computer Society, San Jose, California, 2001, pp. 369–376.

[21] B. Liu, W. Hsu, Y. Ma, Integrity classification and association rule mining, in: R. Agrawal, P. Stolorz, G. Piatetsky-Shapiro (Eds.), Proceeding of the Fourth Int. Conference on Knowledge Discovery and Data Mining, AAAI Press, New York, New York, 1998, pp. 80–86.

[22] H.Y. Liu, J. Chen, G. Chen, Mining insightful classification rules directly and efficiently, Proceeding of the 1999 IEEE International Conference on Systems Man and Cybernetics, IEEE Computer Society, Tokyo, 1999, pp. 911–916.

[23] B. Liu, Y. Ma, C. Wong, Classification using association rules: weaknesses and enhancements, in: Vipin Kumar, et al., (Eds.), Data Mining for Scientific and Engineering Applications, 2001, p. 591.

[24] H. Lu, H. Liu, Decision tables: Scalable classification exploring RDBMS capabilities, Proceeding of the 26th International Conference on Very Large Databases, Morgan Kaufmann, Cairo, Egypt, 2000, pp. 373–384.

[25] M. Mehta, R. Agrawal, J. Rissanen, SLIQ: A fast scalable classifier for data mining, Proceeding of the Fifth Interna-

tional Conference on Extending Database Technology, Springer, Avignon, France, 1996, pp. 18–32.

[26] D. Meretakis, B. Wüthrich, Extending naïve Bayes classifiers using long itemsets, Proceedings of 5th International Conference on Knowledge Discovery and Data Mining, San Diego, California, August 1999, 1999.

[27] C.J. Merz, P. Murphy, UCI Repository of Machine Learning Databases, 1996 (http://www.cs.uci.edu/~mlearn/MLRepository.html).

[28] A. Mueller, Fast Sequential and Parallel Algorithms for Association Rule Mining: A Comparison, 1995 (CS-TR-3515).

[29] G. Piatetsky-Shapiro, U. Fayyad, P. Smyth, From data mining to knowledge discovery, in: G. Piatetsky-Shapiro, U. Fayyad, P. Smyth (Eds.), An Overview. Advances in Knowledge Discovery and Data Mining, AAAI/MIT press, 1996, pp. 1–35.

[30] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, 1993.

[31] J. Roberto, J. Bayardo, Brute-force mining of high-confidence classification rules, Proceeding of the Third International Conference on Knowledge Discovery and Data Mining, AAAI Press, Newport Beach, California, 1997.

[32] J. Roberto, Bayardo Jr., R. Agrawal, D. Gunopulos, Constraint-based rule mining in large dense databases, Proc. of the 15th International Conference on Data Engineering, 1999, pp. 188–197.

[33] J. Shafer, R. Agrawal, M. Mehta, SPRINT: A scalable parallel classifier for data mining, Proceeding of the 22nd VLDB Conference, Morgan Kaufmann, India, 1996, pp. 544–555.

[34] R. Srikant, Q. Vu, R. Agrawal, Mining association rules with item constraints, Proc. of the 3rd Int'l Conference on Knowledge Discovery in Databases and Data Mining, AAAI Press, Newport Beach, California, USA, 1997.

[35] S. Weiss, C. Kulikowski, Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems, Morgan Kaufman, 1991.

[36] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann, San Francisco, USA, 2000.

[37] M.J. Zaki, S. Parthasarathy, M. Ogihara, W. Li, New algorithms for fast discovery of association rules, American Association for Artificial Intelligence (1999).

Guoqing Chen received his PhD from the Catholic University of Leuven (K.U. Leuven, Belgium) and now is Professor of Information Systems at School of Economics and Management, Tsinghua University (Beijing, China). His research interests include Information Systems Management, Business Intelligence and Decision Support, and Soft Computing. Dr. Chen is member of ACM (SIGMOD and SIGKDD) and AIS and has wide publications internationally including a monographic book on data modeling published by Kluwer Academic Publishers (Boston, 1998).

Hongyan Liu has received her PhD in Management Science from Tsinghua University, China. She is an associate professor in the Management Science and Engineering Department at Tsinghua University. Her current research interests include database and information system, data warehouse, knowledge discovery in database and bioinformatics.

Lan Yu has received his bachelor's degree in Management Information Systems from the School of Economics and Management, Tsinghua University. He is a PhD candidate student in Management Science and Engineering Department at SEM. His current research interests focus on supervised learning (e.g., classification, reinforcement learning) and mainly apply them in finance and traffic domain.

Qiang Wei has received his PhD in Management Science from School of Economics and Management in 2003. He is an assistant professor in the Department of Management Science and Engineering, School of Economics and Management, Tsinghua University, China. His current research interests include knowledge discovery, data mining techniques, management information systems, system simulations. He has been a lead author for papers that have appeared in *Journal of Information Sciences*, *International Journal of Intelligent Systems*, and *Journal of Computer and Industrial Engineering*.

Xing Zhang has received his bachelor's degree in Management Information Systems from the School of Economics and Management, Tsinghua University. He is a PhD candidate student in Management Science and Engineering Department at SEM. His research interests focus on classification, reinforcement learning, and data mining techniques.