Optimal and Robust Rule Set Generation

Jiuyong Li (MPhil)

School of Computing and Information Technology
Faculty of Engineering and Information Technology
Griffith University
Brisbane, Australia

A thesis submitted in fulfillment of
the requirements of the degree of Doctor of Philosophy

25 February 2002

To my family for their unfailing support.

Abstract

The rapidly growing volume and complexity of modern databases makes the need for technologies to describe and summarise the information they contain increasingly important. Data mining is a process of extracting implicit, previously unknown and potentially useful patterns and relationships from data, and is widely used in industry and business applications.

Rules characterise relationships among patterns in databases, and rule mining is one of the central tasks in data mining. There are fundamentally two categories of rules, namely association rules and classification rules. Traditionally, association rules are connected with transaction databases for market basket problems and classification rules are associated with relational databases for predictions. In this thesis, we will mainly focus on the use of association rules for predictions.

An optimal rule set is a rule set that satisfies given optimality criteria. In this thesis we study two types of optimal rule sets, the informative association rule set and the optimal class association rule set, where the informative association rule set is used for market basket predictions and the class association rule set is used for the classification. A robust classification rule set is a rule set that is capable of providing more correct predictions than a traditional classification rule set on incomplete test data.

Mining transaction databases for association rules usually generates a large number of rules, most of which are unnecessary when used for subsequent prediction. We define a rule set for a given transaction database that is significantly smaller than an association rule set but makes the same predictions as the complete association rule set. We call this rule set the informative rule set. The informative rule set is

not constrained to particular target items; and it is smaller than the non-redundant association rule set. We characterise the relationships between the informative rule set and the non-redundant association rule set. We present an algorithm to directly generate the informative rule set without generating all frequent itemsets first, and that accesses databases less often than other direct methods. We show experimentally that the informative rule set is much smaller than both the association rule set and the non-redundant association rule set for a given database, and that it can be generated more efficiently. In addition, we discuss a new unsupervised discretization method to deal with numerical attributes in general association rule mining without target specification. Based on the analysis of the strengths and weaknesses of two commonly used unsupervised numerical attribute discretization methods, we present an adaptive numerical attribute merging algorithm that is shown better than both methods in general association rule mining.

Relational databases are usually denser than transaction databases, so mining on them for class association rules, which is a set of association rules whose consequences are classes, may be difficult due to the combinatorial explosion. Based on the analysis of the prediction mechanism, we define an optimal class association rule set to be a subset of the complete class association rule set containing all potentially predictive rules. Using this rule set instead of the complete class association rule set we can avoid redundant computation that would otherwise be required for mining predictive association rules and hence improve the efficiency of the mining process significantly. We present an efficient algorithm for mining optimal class association rule sets using upward closure properties to prune weak rules before they are actually generated. We show theoretically the efficiency of the proposed algorithm will be greater than Apriori on dense databases, and confirm experimentally that it generates an optimal class association rule set, which is very much smaller than a complete class association rule set, in significantly less time than generating the complete class association rule set by Apriori.

Traditional classification rule sets perform badly on test data that are not as com-

plete as the training data. We study the problem of discovering more robust rule sets, i.e. we say a rule is more robust than another rule set if it is able to make more accurate predictions on test data with missing attribute values. We reveal a hierarchy of k-optimal rule sets where a k-optimal rule set with a large k is more robust, and they are more robust than a traditional classification rule set. We introduce two methods to find k-optimal rule sets, i.e. an optimal association rule mining approach and a heuristic approximate approach. We show experimentally that a k-optimal rule set generated from the optimal association rule mining approach performs better than that from the heuristic approximate approach and both rule sets perform significantly better than a typical classification rule set (C4.5Rules) on incomplete test data.

Finally, we summarise the work discussed in this thesis, and suggest some future research directions.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my sincerely thanks to Professor Rodney Topor, Professor Hong Shen, Professor Paul Pritchard and the School of CIT for financially supporting my PhD studies. I would like to thank Professor Hong Shen and Professor Rodney Topor (both are principal supervisors) for their excellent supervision and Aaron Harwood for his beneficial discussions. I also wish to thank Professor Geoffrey Webb for his supervision prior to my PhD studies. I would not have been able to complete this thesis without the support from my family. Thank you Youhong and Xi and I will spend more time with you now. Thank all who helped me in the last three and a half years of research and studies.

Statement of originality

This work has not previously been submitted for a degree or diploma in any university. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

Notation

Upper case letters, e.g. X, Y, Z	Sets (a set of items or a set of attribute-value pairs)
Lower case letters, e.g. c, z	Elements in a set (items or attribute value pairs)
$\|X\|$	Cardinality of set X. (i.e. the number of elements)
$XY = X \cup Y$	Union of sets X and Y
$Xz = X \cup \{z\}$	Union of sets X and $\{z\}$
$X \backslash Y$	X less Y (the elements of X that are not in Y)
$\mathcal{P}(X)$	Set of all subsets of X
$\mathcal{P}^l(X)$	Set of all subsets of X with cardinality of l
$\Rightarrow$	Implication
$\forall$	For all
$\exists$	There exists
$\nexists$	There exists no
$\binom{m}{n} = \frac{m!}{n!(m-n)!}$	Binomial coefficient
σ and ψ	The minimum support and confidence
κ and μ	The minimum accuracy and local support

Chapter 1

Introduction

The rapidly growing volume and complexity of modern databases makes the need for technologies to describe and summarise the information they contain increasingly important. Usually, knowledge in a database consists of patterns and relationships. Knowledge discovery from databases (KDD) [31] is a process of extracting implicit, previously unknown and potentially useful knowledge from data. Considering this new, implicit and user-interesting knowledge is often buried under a large amount of trivial, obvious and user-uninteresting observations, data mining is a suitable term for such process. Data mining [43, 41] is a multi-disciplinary research field, and many research fields have their contributions, such as database, machine learning, statistics, and artificial intelligence. It includes several sub-fields such as rule generation, classification and clustering, probabilistic modelling and visualization.

The rule is one of the most expressive and human readable representations for knowledge, and hence rule mining is one of the central tasks in data mining. There are fundamentally two categories of rules: association rules and classification rules. Correspond to these two rule types, two groups of rule generation techniques, association rule mining and classification rule mining, are developed from database and machine learning communities respectively. The relationships between these two types of rule mining techniques are shown in Figure 1.1, where three paths between databases to goals represent three research directions. The work reported in this thesis focuses mainly on

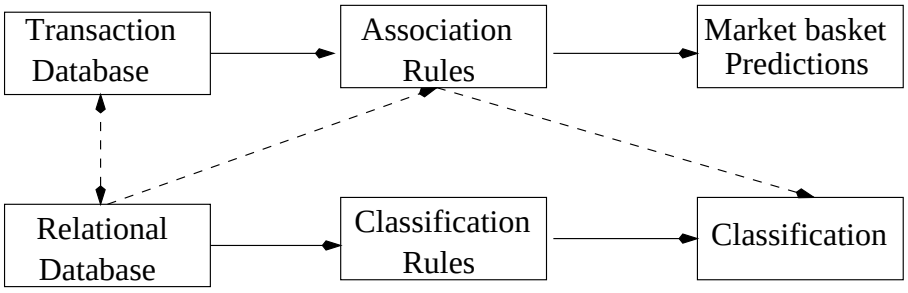


Figure 1.1: Two types of rule mining techniques and their relationships

the top two paths, namely mining transaction databases for predictive association rules and mining relational databases[1] for classification rules by association rule mining. In this chapter, we will summarise some significant existing work closely related to the work in this thesis, and reveal the connections between the work in the thesis and other existing work.

1.1 Association rule mining

1.1.1 General description

Consider an abstraction of a market basket problem, each customer's checkout items form a transaction, and a collection of transactions constitute a transaction database. A set of items is called an itemset. Given two disjoint itemsets X and Y, we say $X \Rightarrow Y$ is an association rule if

1. the fraction of transaction containing both X and Y is at least $s\%$ from a database (support), and

2. at least $c\%$ transactions containing X include Y as well (confidence),

where s and c are the minimum support and the minimum confidence respectively.

The primary goal of association rule mining [2] is to find all rules satisfying the minimum support and the minimum confidence requirements.

Association rules are widely accepted for the following reasons:

[1]The term relational database is used as the definition in this thesis

- Support and confidence criteria are simple and meaningful. Support and confidence are very easy to compute. Although they are simple, they convey important information for significance and accuracy of a rule. Further, most complex interestingness criteria can be derived from them [11].

- The support constraint enables efficient algorithms. All supersets of an infrequent itemset are infrequent. This upward closure property of infrequent itemsets enable efficient algorithms for association rule mining, such as Apriori [4, 3].

1.1.2 Algorithms for mining association rules

Generally, an association rule mining algorithm involves the following two stages:

1. Generating all itemsets whose support is above the minimum support (frequent itemsets), and

2. Forming all association rules from the frequent itemsets.

Most work has been focused on the first step, since it has been believed that the first stage process accounts for most cost of the whole procedure for association rule mining. In fact, the second stage process is also very expensive. One reason that the first stage was favored is that the upward closure property of infrequent itemsets enables efficient algorithms. Based on the property, Aprawal et al presented Apriori [4, 3]. Apriori is a fundamental algorithm in association rule mining, so we give a brief description as follows.

Apriori is a level-wise algorithm, and it generates $(l+1)$-level frequent candidates, which are potential frequent itemsets with $(l+1)$ items, from l-level frequent itemsets.

Apriori Generator:
// Combining
for each pair of itemsets $\{i_1, i_2, \ldots, i_{l-1}, i_p\}$ and $\{i_1, i_2, \ldots, i_{l-1}, i_q\}$
in all l-level frequent itemsets
insert itemset $\{i_1, i_2, \ldots, i_{l-1}, i_p, i_q\}$ in the $(l+1)$-candidate set

```
// Pruning
for each itemset in the (l + 1)-candidate set
    if any of its l-sub itemset is not frequent
        remove the set from the (l + 1)-candidate set
```

The generator involves two steps, combining and pruning. Since a large proportion of item combinations are infrequent, combining step avoids generating many unqualified candidates. In the pruning step, a $(l+1)$-level candidate is pruned if any of its l-sub itemset is not in l-level frequent itemset. This is a direct application of the upward closure property of infrequent patterns. After all candidates are generated, one pass of accessing the database is necessary to verify frequent $(l+1)$-itemsets. So, Apriori needs to access a database as many times as the length of the longest frequent itemset plus one. It is clear that the storage structure for frequent itemsets is crucial in counting the frequency of itemset occurrences. A hash-based storage has been shown to be efficient [4, 84]. In recent reports [9, 13, 42], the prefix tree (or set-enumeration tree) [96] has been commonly used as storage structure, since it is very efficient in deposit and retrieval and is simple to operate.

Apriori has been the most important algorithm in association rule mining, and some variant algorithms [49, 50, 74, 84, 94] are presented subsequently.

Other different algorithms [114, 99, 42, 98] also claimed improvement over Apriori. They all trade memory consumption for speed.

MaxClique [114] and Miner [98] generate all candidates based on 2-frequent itemsets. This candidate generating method saves time for accessing a database, but the number of candidates is significantly larger than that of Apriori, and the increment of candidates expands greatly when the length of transactions increases and the minimum support decreases. When the candidates cannot reside in main memory, they may not be faster than Apriori.

MaxClique [114] and VIPER [99] use vertical layout of transactions. It is clear that if the list of transactions supporting an itemset is stored along with the itemset,

the process of counting will be more efficient. However, the memory usage increases greatly.

FP-growth algorithm [42] projects a database into a dense frequent prefix tree, and hence the subsequent accessing cost is reduced. The frequent prefix tree could be very large, and may not be stored in main memory. In this case, its efficiency may not as good as Apriori.

More detailed discussions on relationships between speed and memory usage are presented in Chapter 2.

There are some algorithms that focus on finding the longest frequent itemsets [65, 13], since from the longest frequent itemsets all frequent itemsets can be obtained (but without precise support information).

Princer-search [65] combines both top-down and bottom-up search methods in searching the maximal frequent patterns. However, when the average length of the maximal frequent patterns is much smaller than the average transaction length (which is true for most databases) it will not be more efficient than Apriori because in this case the top-down search dose not help much.

Max-Miner [13] uses a depth first method to dig the maximal (in length) frequent patterns efficiently. However, this method is not suitable for forming association rules because the maximal frequent itemsets do not contains all necessary information for forming rules.

Zaki [111] considered the problem in a new way. He revealed sets of closed itemsets where all itemsets in one closed itemset family have the same support. So it is not necessary to count frequencies of all frequent itemsets. This novel idea reduces the number of counted frequent itemsets. However, the algorithm for generating frequent closed itemsets is a depth-first algorithm [113], and hence it needs to access a database as many times as the number of frequent closed itemsets. It may not be more efficient than Apriori when a database cannot reside in main memory.

Generally, in designing association rule mining algorithms, we have to consider three factors: speed, memory usage and IO cost (the number of times for accessing a

database). One big problem for association rule mining is combinatorial explosion. As a result, there may not be enough main memory to trade for speed. In addition, a goal for association rule mining is to explore very large databases. Hence we cannot expect all databases to reside in main memory. Consequently, Apriori is very competitive because of its reasonable memory usage and IO cost.

In the following, we will have a look at the second stage process, namely forming rules.

Forming rules is straightforward, but by no means a simple procedure. Consider a set of frequent itemsets $\mathcal{F}$ containing frequent itemsets with the length of l. The number of all possible rules approaches $|\mathcal{F}|2^l$. Usually, the number of frequent itemsets is very large, and the cost for forming rules is large as well.

An algorithm conducting the above two-stage process is called an indirect association rule mining algorithm. A direct association rule mining algorithm incorporates the two stages into one to form rules at the same time when generating frequent itemsets. A direct algorithm could be more efficient than an indirect algorithm because it may not require all frequent itemsets. Here, we consider some novel direct mining algorithms. The constraint rule miner [12] is very efficient when the consequence is fixed, and it only accesses a database as many times as the number of items in the antecedent of the longest rule plus one. However, it may not be efficient when the number of targets is large, and neither are other optimality miners [11]. An OPUS-based miner [108] is very efficient when the goal is to find a number of highest support rules as shown in [115]. However, it needs to read a database as many times as the number of different antecedents among all generated rules. It is clear that it may be inefficient when a database cannot reside in main memory.

In this thesis, we propose two direct algorithms, one is for transaction databases and the other for relational databases. They differ from the existing direct algorithms in that they generate rules with respect to all possible consequences by conducting a limited number of interactions with a database.

1.1.3 Other issues in association rule mining

There are many variations of frequent itemset finding, for example, sequential patterns [6], frequent episodes, [73], and generalized association rules [101]. Other related issues are updating frequent itemsets [10] and rules [103], online mining [45], and parallel and distributed mining [112, 110]. In the following, we only discuss the two fundamental issues that are closed related to our research.

Interestingness criteria and interesting rule sets

The traditional interestingness criteria for association rules are support and confidence [4]. The popularity of the association rule mining is largely due to these simple and meaningful criteria, but most criticisms are incurred from them as well. A number of new interestingness criteria have been presented, among which are gain [36], chi-square [18], interest [19] (or lift [108]), conviction [19], surprisingness [32], unexpectedness [26], R-interesting (indicates the "greater-than-expected-value") [5, 102], actionability [100], and many more [46, 51, 105]. The general discussions on the interestingness of a rule is reported in [86, 71, 52]. Indeed it is hard to have a general single metric to quantify "interestingness" of a rule for all users, since all users may have their own criterion. If an interesting criterion is only used as a filter to select interesting rules after all rules are found, then any choice will not affect the efficiency of the rule mining procedure much. It is more challenging and worthwhile to explore interestingness criteria that produce more efficient algorithms for mining interesting rule sets. There have been some such proposals, which are discussed in the following.

The algorithm for generating correlated association rule set [18] uses the upward closure property of chi-squared statistic (in the binomial case) to reduce the subsequent rule forming cost. The rule set can be generated more efficiently and meaningfully than a traditional association rule set. However, both complexity and inaccuracy of the chi-square increase when the number of cells in a contingent table grows. Similar ideas also appeared in [25] by using a simplified chi-squared criterion, but the algorithm only generates rules between two items.

The constraint rule set [12], which is a rule set generated by imposing a confidence improvement requirement for those long rules, avoids generating many candidates leading to rules gaining no improvement over their simple form rules, and hence is efficient. However, the algorithm assumes that the consequence is fixed, and hence is unsuitable for application without constraint targets or with a large number of targets.

The same problem occurs in optimality rule set miners [11]. Here we have a few words on SC-optimality rule set because of its exceptionally small size. An SC-optimality rule set includes rules that are maximal in support or confidence. However, these rules may be easily observed and not actually of interest to a user. In addition, those rules may cover only a small proportion of a database, and hence lack comprehensive explanative and predictive ability.

A non-redundant association rule set [111] excludes the derivable rules that will be defined in Chapter 2, and hence is smaller than an association rule set. However, it may not be generated more efficiently. Firstly, the frequent closed itemsets are generated by a depth-first algorithm. As we stated before, it may be inefficient when the database cannot reside in main memory. Secondly, the number of frequent itemsets used to generate the non-redundant rule set is not necessarily less than the number of general frequent itemsets. A closed itemset is a maximum itemset, while a non-redundant rule is a simple rule related to short itemsets. So, all short itemsets in a closed family have to be extended when generating non-redundant rules from a frequent closed itemsets. As a result, rule generation efficiency may not be improved.

Our work has significant contributions on the topic. We proposed the minimum predictive rule sets for both transaction databases and relational databases. They are interesting because they are small in size and make the same predictions as association rule sets do. They can be generated more efficiently because two upward closure properties can prune many unqualified rules before they are generated.

Quantitative association rule mining

A database may contain numerical attributes, and a general association rule mining algorithm cannot be applied before a numerical attribute is discretized. Note that in the traditional association rule mining there are no pre-specified targets, hence all supervised discretization methods [28] are unsuitable. Two main unsupervised discretization methods are equal-width discretization and equal-depth discretization. The former divides the whole range of a continuous attribute into N intervals of equal value distance and the latter separates the whole range of continuous attribute into N intervals so that there are $1/N$ of the total instances in each interval.

The equal-depth discretization method is preferred in quantitative association rule mining [102]. A problem of using the equal-depth method is the difficulty in determining how many intervals are suitable in splitting an attribute. A small interval size possibly results in loss of interesting rules, and a large interval size on the other hand tends to reduce the accuracy of rules. To deal with this problem, a measure of K-partial completeness is defined to decide how many intervals is suitable for a continuous attribute [102]. However, this method may not work very well on highly skewed data.

An adaptive method [59] proposed by us considers both instance densities and value distances at the same time, and need not pre-specify the number of intervals. It achieves improvement over both equal-width and equal-depth methods.

More precisely, a new form of association rule [78], the distance-based association rule, is proposed for interval data, but this new technique is sensitive to input thresholds.

There are many other proposals for quantitative association rule mining. However, they use presumed targets or templates [37, 55, 106, 29]. When there are pre-specified targets, a supervised discretization method is a better choice. This is because all supervised discretization methods are better than unsupervised discretization methods [28, 53]. As far as supervised discretization methods are concerned, most research has been conducted in the machine learning community. We outline them briefly here so that we do not address the same topic in Section 2.

Information gain [87] was used as a measure of binary splitting numerical attributes in constructing decision trees. The cut-point is the one with the highest information gain. Further, this procedure is extended for multi-value splitting [20], in which subsets of the previous binary splitting are partitioned recursively until a stopping criterion is reached. Usually, the stopping point is decided by the minimum number of instances in one interval, the maximum number of intervals or the minimum information gain. In addition, a MDL (Minimum Description Length principle) based criterion was used [30] to decide the stopping point. There are many different proposals in supervised discretization, and summaries and comparisons can be found [28, 21, 47]. However, there is no conclusion as to which method performs consistently better than others.

1.2 Classification rule mining

1.2.1 General description

Consider a relational database, each column represents a domain, called an attribute, and contains a finite number of values. Each row is called a record, which is a tuple of attribute values or simply a set of attribute and attribute-value pairs. A database is called a training database if there is a special attribute, called the class attribute. Otherwise it is called a test database. Classification is a process to categorize records in a test database to known classes, and is a form of prediction. A record in a database is also referred to as an instance. Consider a pattern as a set of attribute-value pairs, a classification rule is an accurate implication from a pattern to a class. A rule is capable of making a prediction on a record if its antecedent is a subset of the record. A rule based classifier is a sequence of classification rules and a default class. A rule set based classifier works in the following way: for a test record, use the highest precedence rule to match the record to see if it can make prediction. If so, then the class of the record is predicted to be the consequence of the rule. If not, try the next highest precedence rule. If no rule can make a prediction on the record, the prediction is the default class.

1.2.2 Two types of traditional classification rule mining algorithms

Classification rule mining algorithms have been mainly developed in machine learning community. They are used to find a simple rule set to best fit training data.

Before discussing algorithms for classification rule mining, we introduce the covering algorithm [77], which is the basis for most classification rule mining algorithms.

Covering algorithm:
Input: a training database D
Output: a rule set R

set $R = \emptyset$
while ($D \neq \emptyset$)
    find a rule from D heuristically
    remove all records identified by the rule from D
    add the rule to R
return R

It is clear that all rules in the found rule set have little overlapping coverage because those covered records are removed once a rule is found. Or, a record in the database generally supports only one rule in the rule set. As a result, a rule set from the covering algorithm is a set of rules where each rule covers disjoint set of records (or nearly).

Classification rule mining algorithms are generally categorized into two groups, simultaneously covered set algorithms and sequential covered set algorithms [80]. The simultaneously covered set algorithms, such as ID3 and C4.5 [87, 89], first of all select a "best" attribute by a heuristic criterion, and then partition the database into a set of disjoint sub data sets by distinct attribute values of the attribute. They next recursively partition each sub data set in the same way. This procedure stops when the partitioned data set is pure (each data set only contains instances of one class.). All the selected attributes and their distinct values form a decision tree, where each path from the root

to a leaf is a rule. In practice, a presented rule set is completed by post-pruning the raw rule set derived from a decision tree.

The sequential covered set algorithms, such as AQ15 [76] and CN2 [22, 23] search for a "best" rule that accounts for a part of training data. They separate these explained instances, and then recursively search for another rule in the remaining data set until no training instance remains. In the above procedure, the "best" rule is chosen by a heuristic criterion.

There are many rule mining algorithms under the covering algorithm. They are distinguished by their heuristic criteria and search methods. Some commonly used heuristic criteria are purity [83, 109], information gain [87], information gain ratio [89], cross entropy (J-measure [39] and m-estimate[23]), Laplace estimate [22]. Search methods include hill climbing as in [89], and beaming searching [22, 23]. Both top-down and bottom-up searching methods are adopted in classification rule mining algorithms. More discussion on classification rule mining algorithm can be referred from [80, 38]

Most classification rule mining algorithms have some problems. Firstly, most of them are heuristic, so they may miss some global optimal rules. Secondly, the found rule sets tailor the training data, so they may suffer from the over-fitting problem. Thirdly, they usually access training data many times so that they may not suit for mining rule sets from large databases. In this thesis, we will further argue that a rule set from a heuristic algorithm is not as robust as a rule set from our association rule mining technique.

1.2.3 Other issues in classification rule mining

Classification is a broad research field and includes many research topics, such as, decision tree [17, 89], neural networks [91, 90], instance based classifier [7], genetic learning [48], Bayesian classification [35], etc. In this thesis, we only concentrate on classification rules, or prepositional rules. First order rules [88, 85] containing variables in their antecedents are not our concern. In the following, we consider two problems that are closely related to our research.

Robust rule generation and predictions

In real world databases, there may exist missing data, or missing attribute values. A robust system should be able to withstand missing values in a database. Let us consider this problem in two cases, missing attribute values in training databases and missing attribute values in test databases.

For the first case, there have been some methods proposed. When partial values in an attribute are missing, the most straightforward solution is to replace the missing values with the most frequent value in the attribute. More precisely, frequencies of attribute values are counted against every class, and a missing value in a record is replaced by a most frequent value in the same class. This method is used in [79]. Instead of replacing a missing attribute value by one certain value, an attribute value can be assigned by multiple values with estimated probabilities based on the observed frequencies of different values. This method is used in [89].

For the second case that the missing attribute values occur in the test data, to the best of our knowledge there is no previous study. We say a rule set is more robust than another rule set if it can make more accurate predictions on a test database with missing attribute values than the other rule set. In this thesis, we study the properties of robust rule sets and present algorithms to generate robust rule sets.

Systematic searching algorithms in classification rule mining

A heuristic searching method finds only one of a set of possible rule sets that fit the training database, but the found rule set may be not the best one. Even though we may generate a set of rule sets by sampling the training database [16, 34], there still is no guarantee for finding the optimal one.

This shortage of heuristic searching has been noticed and some systematic searching algorithms have been developed to search for the best rules. Based on a fixed search tree, or an enumeration tree [92], all possible solutions in a problem space can be searched once. This structure guarantees the finding of the best solution, and is the basis for some systematic searching algorithms such as [24, 93, 95, 97]. However, they

may suffer combinatorial explosion for databases with many attributes. The feature subset selection algorithm (FSS) [81] reorganize the searching tree dynamically, so that the states that lead to non best solution can be maximally pruned. OPUS [107] is an enhancement of FSS by incorporation with A* algorithm [44]. It is implemented with a machine learning pruning mechanism. Although OPUS is very efficient in its pruning mechanism, it searches for only one best rule at one time. So it is still very expensive for finding an optimal rule set. When the searching goal is satisfactory rather than optimal, OPUS can be adapted for association rule mining as in [108]. However, this modified OPUS efficiently produces the highest support rule set that does not suit for predictions. When the modified OPUS is used for generating all association rules, it may not perform as well as Apriori as shown in [115]. In addition, the modified OPUS needs to read a database as many times as the number of different antecedents in the found rule set, so it may not work efficiently when a database cannot reside in main memory.

In this thesis, we present an efficient algorithm to mine the optimal class association rule set. An optimal rule set is significantly smaller than an association rule set, but includes all potential predictive rules. This algorithm is more efficient than Apriori and accesses a database less often than Apriori.

1.3 Relationships between association rules and classification rules

Apparently, classification rules and association rules are distinct in the following aspects.

- Classification rules are generated from a relational database for solving classification problems, whereas association rules are originally generated from a transaction database for market basket problems.

- Classification rules always have pre-specified consequences (classes) that never appear in the antecedent of a classification rule, whereas association rules usually

have no pre-specified consequences and the consequence of an association rule may be in the antecedent of another rule.

- The goal of classification rule mining is to obtain a set of simple and accurate rules, whereas the goal of association rule mining is to find all rules satisfying some thresholds.

- A classification rule mining algorithm usually uses a heuristic criterion and cannot guarantee finding an optimal rule set, whereas an association rule mining algorithm use a systematic search method which can produce an optimal rule set.

Despite the above differences, they are closely related. For example, a relational database can be mapped to a transaction database when numerical attributes are discretized. A classification rule set is a subset of an association rule set with some specified consequences when the minimum support is low enough. Previous classification rule mining algorithms already have an implicit minimum accuracy requirement. An minimum support can greatly reduce risks of the over-fitting problem suffered by a classification rule set. More details discussions are presented in Chapter 4.

Consequently, it is possible to use association rule mining techniques to solve classification rule mining problems. Bing Liu et al [66, 70] have realized this potential, and revealed that the classifiers built on an association rule set are more accurate than those traditional ones and more work in this direction appeared in [64]. Other studies [27, 75] also obtained improvement over traditional classifiers. In this thesis, we further claim that a rule set from association rule mining technique is more robust than a rule set from a traditional classification rule mining algorithm. We reveal the relationships between association rule sets and classification rule sets.

In the procedure of applying association rule mining techniques to classification rule mining, some progresses have been made.

Firstly, a relational database is usually denser than a transaction database, hence the risk of combinatorial explosion increases when an association rule mining algorithm is applied to a relational database. The technique of mining the optimal class associ-

ation rule set [60, 62], which will be discussed in Chapter 3, significantly reduces the number of rules in a dense relational database for prediction purposes, and makes the rule mining procedure much more efficient than Apriori.

Secondly, the distribution of classes is uneven in most relational databases, so a universal minimum support cannot work well. The concept of multiple minimum supports [67] is presented to solve this problem. Simultaneously, our proposal [58] for the local support solves the same problem.

1.4 Main contributions of this thesis

Main contributions in this thesis are in the following four aspects:

In traditional association rule mining on transaction databases, a significant problem is that there are many generated rules, which are not of interest to users. We may select all interesting rules from all generated association rules, but this is clearly inefficient since we use much time to generate and then remove those unwanted rules. We observed that when using association rules for market basket prediction, only partial rules are useful. Hence, we define the informative association rule set to be the smallest subset of an association rule set to provide the same predictions as the association rule set by the confidence priority. We also characterise the relationships between the informative association rule set and the non-redundant association rule set. We further present a direct algorithm, which needs not generate frequent itemsets first like most association rule mining algorithms did, to produce the informative association rule set. Unlike other direct association rule mining algorithms, the proposed algorithm accesses a database less often for generating rules on all possible items. We show experimentally that the informative association rule set is significantly smaller than both the association rule set and the non-redundant association rule set and the proposed direct algorithm is very efficient in generating the informative association rule set. We also show that this efficiency improvement is due to less database accesses and small number of rule candidates rather than using large memory consumption as a tradeoff as did in other efficient algorithms. Details of this work are presented in Chapter 2.

In quantitative association rule mining, there are many previous proposals. However, most of them use supervised discretization methods that make use of pre-specified target information, and they may be inapplicable in traditional association rule mining where no targets are pre-specified. We analyse the strengths and weaknesses of two most commonly used unsupervised discretization methods, and present an adaptive generalized method that is better than both methods in quantitative association rule mining. Details of this work are also presented in Chapter 2 because this work is related to mining association rules without class information.

In mining relational databases for classification rules, the generation of all class association rules may be difficult even from a small relational database because a relational database is much denser than a transaction database. In addition, a generated large class association rule set reduces the subsequent post pruning efficiency. We define the optimal class association rule set that includes all potentially predictive class association rules. This optimal class association rule set is significantly smaller than the complete class association rule set for a database. We present an efficient algorithm to generate the optimal class association rule set, which is demonstrated theoretically and experimentally to be significantly more efficient than Apriori in a dense relational database. Details of this work are presented in Chapter 3.

In the research on generating classification rule sets from association rule mining approach, some previous work shows that more accurate classifiers can be produced. In this thesis, we further show that a classification rule set generated from association rule mining approach will be more robust than traditional classification rule sets. That is, it can provide more accurate predictions than a traditional classification rule set when a test database is not as complete as the training database. We study a new problem of mining robust classification rule sets, and present a robust rule set generation method which produces rule sets that are more robust than those produced from an extension of C4.5Rules. We reveal the relationships between a traditional classification rule set and the optimal classification rule set through a hierarchy of k-optimal rule sets with increasing robustness. We experimentally show that a k-optimal rule set performs

significantly better than a traditional classification rule set when there are missing values in a test database. Details of this work are presented in Chapter 4.

Finally, we present some future research problems identified in the process of the research in this thesis in Chapter 5.

Chapter 2

The informative association rule set

Mining transaction databases for association rules usually generates a large number of rules, most of which are unnecessary when used for subsequent prediction. In this chapter we define a rule set for a given transaction database that is much smaller than the association rule set but makes the same predictions as the association rule set by the confidence priority. We call this subset the informative rule set. The informative rule set is not constrained to particular target items; and it is smaller than the non-redundant association rule set. We characterise relationships between the informative rule set and the non-redundant association rule set. We present an algorithm to directly generate the informative rule set without generating all frequent itemsets first, which accesses the database less often than other direct methods. We show experimentally that the informative rule set is much smaller than both the association rule set and the non-redundant association rule set for a given database, and that it can be generated more efficiently. Furthermore, we discuss a new unsupervised discretization method to deal with numerical attributes in general association rule mining without target specification. Based on the analysis of the strengths and weaknesses of two commonly used unsupervised numerical attribute discretization methods, we present an adaptive numerical attribute merging algorithm that is shown experientially better than both

methods in general quantitative association rule mining.

Partial work in this chapter has been published in Proceedings of 2001 IEEE International Conference on Data Mining (ICDM 2001) [61] and Proceedings of the 5th International Computer Science Conference (ICSC'99) [59] as full papers.

2.1 Introduction

2.1.1 Association rules and market basket predictions

The rapidly growing volume and complexity of modern databases makes the need for technologies to describe and summarise the information they contain increasingly important. The general term to describe this process is data mining. Association rule mining is the process of generating associations or, more specifically, association rules, in transaction databases. Association rule mining is an important subfield of data mining and has wide application in many fields. Two key problems with association rule mining are the high cost of generating association rules and the large number of rules that are normally generated. Much work has been done to address the first problem. Methods for reducing the number of rules generated depend on the application, because a rule may be useful in one application but not another.

In this chapter, we are particularly concerned with generating rules for prediction. For example, given a set of association rules that describe the shopping behavior of the customers in a store over time, and some purchases made by a particular customer, we wish to predict what other purchases will be made by that customer.

The association rule set [2] can be used for prediction if the high cost of finding and applying the rule set is not a concern. The constrained and optimality association sets [12, 11] can not be used for this prediction because their rules do not have all possible items to be consequences. The non-redundant association rule set [111] can be used, but can be large as well.

We propose the use of a particular rule set, called the informative (association) rule

set, that is smaller than the association rule set and that makes the same predictions based on confidence priority.

We compare the informative rule set with constrained and optimality association rule sets, and characterise relationships between the informative association rule set and non-redundant association rule set.

The general method of generating association rules by first generating frequent itemsets can be unnecessarily expensive, as many frequent itemsets do not lead to useful association rules. We present a direct method for generating the informative rule set that does not involve generating the frequent itemsets first. Unlike other algorithms that generate rules directly, our method does not constrain the consequences of generated rules as in [11, 12] and accesses the database less often than other unconstrained methods [108].

We show experimentally, using standard synthetic data, that the informative rule set is much smaller than both the association rule set and the non-redundant rule set, and that it can be generated more efficiently.

A database may contain numerical attributes, and a general association rule mining algorithm cannot apply before a numerical attribute is discretized. In traditional association rule mining, the consequences are usually unknown before a rule set is generated. Hence, most supervised numerical discretization algorithms are not applicable. The two widely used unsupervised numerical attribute discretization algorithms are equal width and equal depth. In this chapter, we analyze strengths and weaknesses of both methods, and present an adaptive numerical attribute merging algorithm that is shown experientially better than the two methods in generating association rules.

2.1.2 Related work

Association rule mining was first studied in [2] by Agrawal et al. Most research work has been on how to mine frequent itemsets efficiently. Apriori [4] is a widely accepted approach, and there have been many enhancements to it [49, 50, 74, 84, 94]. In addition, other approaches have been proposed [42, 99, 114], mainly by using more memory to

save time. For example, the algorithm presented in [42] organizes a database into a condensed structure to avoid repeated database accesses, and algorithms in [99, 114] use the vertical layout of databases to save counting time.

Some direct algorithms for generating association rules without generating frequent itemsets first have previously been proposed [12, 11, 108]. Algorithms presented in [12, 11] focused only on one fixed consequence and hence is not efficient for mining all association rules. The algorithm presented in [108] needs to scan a database as many times as the number of all possible antecedents of rules. As a result, it may not be efficient when a database cannot be retained in the memory.

There are also two types of algorithms to simplify the association rule set, direct and indirect. Most indirect algorithms simplify the set by post-pruning and reorganization, as in [104, 68, 82], which can obtain an association rule set as simple as a user would like but does not improve efficiency of the rule mining process. There are some attempts to simplify the association rule set directly. The algorithm for mining constraint rule sets is one such attempt [12]. It produces a small rule set and improves mining efficiency since it prunes unwanted rules in the processing of rule mining. However, a constraint rule set contains only rules with some specific items as consequences, as do the optimality rule sets [11]. They are not suitable for association prediction where all items may be consequences. The most significant work in this direction is to mine the non-redundant rule set because it simplifies the association rule set and retains the information intact [111]. However, the non-redundant rule set is still too large for prediction.

Quantitative association rule mining is an important research issue in association mining. Although there are many proposals [37, 55, 106, 29], they use presumed targets or templets. In the traditional association rule mining, we do not know the consequences of rules before a set of rules is generated, and hence all supervised discretization methods [28] are unsuitable. There are some proposal using unsupervised discretization methods [102, 78]. The algorithm presented in [102] bears a difficulty in deciding the number of intervals in splitting an attribute for skewed databases. The distance-based association rule [78] generating algorithm is too sensitive to the input thresholds.

2.1.3 Contributions

Main contributions of the work in this chapter are listed as follows:

We define the informative rule set for a given transaction database, which is the smallest rule set providing the same prediction as the association rule set by the confidence priority. We characterise the relationships between the informative association rule set and the non-redundant association rule set.

We present a direct algorithm to generate the informative rule set efficiently. The algorithm generates rules at the same time when generating frequent itemsets. Unlike other direct association rule mining algorithms, the proposed algorithm accesses the database less often for generating rules on all possible items. Instead of using more memory as a tradeoff for efficiency as some other efficient association rule mining algorithms did, the proposed algorithm utilizes less memory than Apriori.

We present an adaptive merging algorithm for numerical attributes that is better than two most commonly used unsupervised discretization algorithms in quantitative association rule mining.

2.2 The informative association rule set

2.2.1 Association rules and related definitions

Let $I = \{1, 2, \ldots, m\}$ be a set of *items*, and $T \subseteq I$ be a *transaction* containing a set of items. An *itemset* is defined to be a set of items, and a k-itemset is an itemset containing k items. A database D is a collection of transactions. The *support* of an itemset (e.g. X) is the ratio of the number of transactions containing the itemset to the number of all transactions in a database, denoted by $sup(X)$. Given two itemsets X and Y where $X \cap Y = \emptyset$, an association rule is defined to be $X \Rightarrow Y$ where $sup(X \cup Y)$ and $sup(X \cup Y)/sup(X)$ are not less than user specified thresholds respectively. $sup(X \cup Y)/sup(X)$ is called the *confidence* of the rule, denoted by $conf(X \Rightarrow Y)$. The two thresholds are called the *minimum support* and the *minimum confidence* respectively. For convenience, we abbreviate $X \cup Y$ by XY and use the terms rule and association

rule interchangeably in the rest of this chapter.

Suppose that every transaction is given a unique identifier. A set of identifiers is called a *tidset*. Let mapping $t(X)$ be the set of identifiers of transactions containing the itemset X. It is clear that $sup(X) = |t(X)|/|D|$. In the following, we list some basic relationships between itemsets and tidsets.

1. $X \subseteq Y \Rightarrow t(X) \supseteq t(Y)$,
2. $t(X) \subseteq t(Y) \Rightarrow t(XZ) \subseteq t(YZ)$ for any Z, and
3. $t(XY) = t(X) \cap t(Y)$.

We say that rule $X \Rightarrow Y$ is *more general* than rule $X' \Rightarrow Y$ if $X \subset X'$, and we denote this by $X \Rightarrow Y \subset X' \Rightarrow Y$. conversely, $X' \Rightarrow Y$ is *more specific* than $X \Rightarrow Y$. We define the *covered set* of a rule to be the tidset of its antecedent. We say that rule $X \Rightarrow Y$ *identifies* transaction T if $XY \subset T$. We use Xz to represent $X \cup \{z\}$.

To simplify the presentation, we define $sup(X, Z) = sup(X) - sup(XZ)$, $t(X, Z) = t(X) \backslash t(XZ)$, and $cov(X, Z) = cov(X) \backslash cov(XZ)$.

2.2.2 The informative association rule set

Let us consider how a user uses the set of association rules to make predictions. Given an input itemset and an association rule set. Initiate the prediction set to be an empty set. Select a matched rule with the highest confidence from the rule set, and then put the consequence of the rule into prediction set. We say that a rule matches a transaction if its antecedent is a subset of the transaction. To avoid repeatedly predicting on the same item(s), remove those rules whose consequences are in the prediction set. Repeat selecting the next highest confidence matched rule from the remaining rules in the rule set until the user is satisfied or there is not rule to select. The justification for choosing the confidence priority model will be presented in the discussion section.

We have noticed that some rules in the association rule set will never be selected in the above prediction procedure, so we will remove those rules from the association rule set and form a new rule set. This new rule set will predict exactly the same as

the association rule set, the same set of prediction items in the same generated order. Here, we consider the order because a user may stop selection at any time, and we will guarantee to obtain the same prediction items in this case. In addition, the sequence reflects the priority among items in the prediction itemset.

Formally, given an association rule set R and an itemset P, we say that the *predictions* for P from R is a sequence of items Q. The sequence of Q is generated by using the rules in R in descending order of confidence. For each rule r that matches P (i.e., for each rule whose antecedent is a subset of P), each consequent of r is added to Q. After adding a consequence to Q, all rules whose consequences are in Q are removed from R.

To exclude those rules that never been used in the prediction, we present the following definition.

Definition 2.1 Let R_A be an association rule set and R_A^1 the set of single-target rules in R_A. A set R_I is informative over R_A if (1) $R_I \subset R_A^1$; (2) $\forall r \in R_I \nexists r' \in R_I$ such that $r' \subset r$ and $conf(r') \geq conf(r)$; and (3) $\forall r'' \in R_A^1 - R_I$, $\exists r \in R_I$ such that $r'' \supset r$ and $conf(r'') \leq conf(r)$.

The following result follows immediately.

Lemma 2.1 *There exists a unique informative rule set for any given rule set.*

Proof Suppose that we have two informative rule sets R_1 and R_2 for the complete rule set R. If two informative rule sets are not identical, we must have a rule r such that $r \in R_1 \wedge r \notin R_2$. Since r is not in R_2, there must be a rule $r' \in R_2$ such that $r' \subset r$ and $conf(r') \geq conf(r)$. Clearly, R_1 cannot be informative whether it includes or excludes r' by the definition, which is a contradiction.

Consequently, there exists a unique informative rule set for a complete rule set. □

We give two examples to illustrate this definition.

Example 2.1 Consider the following small transaction database: $\{1 : \{a, b, c\}, 2 : \{a, b, c\}, 3 : \{a, b, c\}, 4 : \{a, b, d\}, 5 : \{a, c, d\}, 6 : \{b, c, d\}\}$. Suppose the minimum sup-

port is 0.5 and the minimum confidence is 0.5. There are 12 association rules (that exceed the support and confidence thresholds). They are $\{a \Rightarrow b(0.67, 0.8), a \Rightarrow c(0.67, 0.8), b \Rightarrow c(0.67, 0.8), b \Rightarrow a(0.67, 0.8), c \Rightarrow a(0.67, 0.8), c \Rightarrow b(0.67, 0.8), ab \Rightarrow c(0.50, 0.75), ac \Rightarrow b(0.50, 0.75), bc \Rightarrow a(0.50, 0.75), a \Rightarrow bc(0.50, 0.60), b \Rightarrow ac(0.50, 0.60), c \Rightarrow ab(0.50, 0.60)\}$, where the numbers in parentheses are the support and confidence respectively. Every transaction identified by the rule $ab \Rightarrow c$ is also identified by rule $a \Rightarrow c$ or $b \Rightarrow c$ with higher confidence. So $ab \Rightarrow c$ can be omitted from the informative rule set without losing predictive capability. Rule $a \Rightarrow b$ and $a \Rightarrow c$ provide predictions b and c with higher confidence than rule $a \Rightarrow bc$, so rule $a \Rightarrow bc$ can be omitted from the informative rule set. Other rules can be omitted similarly, leaving the informative rule set containing the 6 rules $\{a \Rightarrow b(0.67, 0.8), a \Rightarrow c(0.67, 0.8), b \Rightarrow c(0.67, 0.8), b \Rightarrow a(0.67, 0.8), c \Rightarrow a(0.67, 0.8), c \Rightarrow b(0.67, 0.8)\}$.

Example 2.2 Consider the rule set $\{a \Rightarrow b(0.25, 1.0), a \Rightarrow c(0.2, 0.7), ab \Rightarrow c(0.2, 0.7), b \Rightarrow d(0.3, 1.0), a \Rightarrow d(0.25, 1.0)\}$. Rule $ab \Rightarrow c$ may be omitted from the informative rule set as the more general rule $a \Rightarrow c$ has equal confidence. Rule $a \Rightarrow d$, must be in the informative rule set even though it can be derived by transitivity from rules $a \Rightarrow b$ and $b \Rightarrow d$. Otherwise, if it were omitted, item d could not be predicted from the itemset $\{a\}$, as the definition of prediction does not provide for reasoning by transitivity.

Now we present the main property of the informative rule set.

Theorem 2.1 *Let R_A be an association rule set. Then the informative rule set R_I over R_A is the smallest subset of R_A such that, for any itemset P, the prediction sequence for P from R_I equals the prediction sequence for P from R_A.*

Proof We will prove this theorem from two steps. Firstly, a rule omitted by R_I does not affect prediction from R_A for any P. Secondly, a rule set omitted one rule from R_I cannot present the same prediction sequences as R_A for any P.

Firstly, we will prove that a rule omitted by R_I do not affect prediction from R_A for any P.

Consider a single-target rule r' omitted by R_I, there must be another rule r in both R_I and R_A such that the $r \subset r'$ and $conf(r) \geq conf(r')$. When r' matches P, r does. If both rules have the same confidence, omitting r' does not affect prediction from R_A. If $conf(r) > conf(r')$, r' must be automatically omitted from R_A after r is selected and the consequence of r is in the prediction sequence. So, omitting r' does not affect prediction from R_A.

Consider a multiple-target rule in R_A, e.g. $A \Rightarrow bc$, there must be two rules $A' \Rightarrow b$ and $A' \Rightarrow c$ in both R_I and R_A for $A' \subseteq A$ such that $conf(A' \Rightarrow b) \geq conf(A \Rightarrow bc)$ and $conf(A' \Rightarrow c) \geq conf(A \Rightarrow c)$. When rule $A \Rightarrow bc$ matches P, $A' \Rightarrow b$ and $A' \Rightarrow c$ do. It is clear that if $conf(A' \Rightarrow b) = conf(A' \Rightarrow c) = conf(A \Rightarrow bc)$, then omitting $A \Rightarrow bc$ does not affect prediction from R_A. If $conf(A' \Rightarrow b) > conf(A \Rightarrow bc)$ and $conf(A' \Rightarrow c) > conf(A \Rightarrow bc)$, rule $A \Rightarrow bc$ must be automatically omitted from R_A after $A' \Rightarrow b$ and $A' \Rightarrow c$ are selected and item b and c are in the prediction sequence. Similarly, we can prove that omitting $A \Rightarrow bc$ from R_A does not affect prediction when $conf(A' \Rightarrow b) > conf(A' \Rightarrow c) = conf(A \Rightarrow bc)$ or $conf(A' \Rightarrow c) > conf(A' \Rightarrow b) = conf(A \Rightarrow bc)$. So omitting $A \Rightarrow bc$ from R_A does affect prediction. Similarly, we can conclude that a multiple-target rule in R_A does not affect its prediction sequence.

Thus a rule omitted by R_I does not affect prediction from R_A.

Secondly, we will prove the minimum property. Suppose we omit one rule $X \Rightarrow c$ from the R_I. Let $P = X$, there must be a position for c in the prediction sequence from R_A determined by $X \Rightarrow c$ because there is not other rule $X' \Rightarrow c$ such that $X' \subset X$ and $conf(X' \Rightarrow c) \geq conf(X \Rightarrow c)$. When $X \Rightarrow c$ is omitted from R_I, there may be two possible results for the prediction sequence from R_I. One is that item c does not occur in the sequence. The other is that item c is in the sequence but its position is determined by another rule $X' \Rightarrow c$ for $X' \subset X$ with smaller confidence than $X \Rightarrow c$. As a result, the two prediction sequences would not be the same.

Hence, the informative rule set is the smallest subset of R_A that provides the same predictions for any itemset P.

Consequently, the theorem is proved. □

Finally, we describe a property that characterises some rules to be omitted from the informative rule set.

We can divide the tidset of an itemset X into two parts on an itemset (consequence), $t(X) = t(XZ) \cup t(X, Z)$. If the second part is an empty set, then the rule $X \Rightarrow Z$ has 100% confidence. Usually, the smaller is $|t(X, Z)|$, the higher is the confidence of the rule. Hence, $|t(X, Z)|$ is very important in determining the confidence of a rule.

Lemma 2.2 *If $t(X, Z) \subseteq t(Y, Z)$, then rule $XY \Rightarrow Z$ does not belong to the informative rule set.*

Proof Let us consider two rules, $XY \Rightarrow Z$ and $X \Rightarrow Z$.

We know that $conf(XY \Rightarrow Z) = s_1/(s_1 + r_1)$, where $s_1 = |t(XYZ)|$ and $r_1 = |t(XY, Z)|$, and $conf(X \Rightarrow Z) = s_2/(s_2 + r_2)$, where $s_2 = |t(XZ)|$ and $r_2 = |t(X, Z)|$.

$r_1 = |t(XY, Z)| = |t(X, Z) \cap t(Y, Z)| = |t(X, Z)| = r_2$.

$s_1 = |t(XYZ)| \leq |t(XZ)| = s_2$.

As a result, $conf(XY \Rightarrow Z) \leq conf(X \Rightarrow Z)$. Hence rule $XY \Rightarrow Z$ must be omitted by the informative rule set. □

This is an important property for the informative rule set, since it enables us to predict rules that cannot be in the informative rule set in the early stage of association rule mining. We will discuss this in detail in section 4.

2.3 Comparison with the non-redundant association rule set

It is clear that the informative rule set is different from constraint [12] and optimality [11] rule sets, because they do not have all possible items to be consequences and subsequently cannot make predictions the same as the association rule set. The non-redundant rule set proposed by Zaki [111] can make the same prediction as the association rule set, but it is larger than the informative rule set. We will discuss its relationship with the informative rule set in the following.

To facilitate our discussion, we first restate non-redundant rules in a way that is easy to compare with our informative rule set.

Generally, we say that a rule is derivable if its confidence and support can be derived from other more general rules. More specifically, rule $X \Rightarrow Y$ is derivable if there is a set of rules R in which all rules are more general than rule $X \Rightarrow Y$, such that rule $X \Rightarrow Y$ and its support and confidence can be obtained from R. For example, rule $ab \Rightarrow c(0.2, 0.7)$ can be derived from two rules $a \Rightarrow b(0.25, 1.0)$ and $a \Rightarrow c(0.2, 0.7)$. The numbers in parentheses are supports and confidences.

We give one type of derivable rules as follows.

Lemma 2.3 *If $t(X) \subseteq t(Y)$, then for any itemset Z rule $XY \Rightarrow Z$ and $Z \Rightarrow XY$ are derivable.*

Proof Since $t(X) \subseteq t(Y)$, rule $X \Rightarrow Y$ is a 100% confidence rule and $sup(XZ) = sup(XYZ)$. As a result, $sup(XY \Rightarrow Z) = sup(X \Rightarrow Z)$ and $conf(XY \Rightarrow Z) = conf(X \Rightarrow Z)$. Consequently, rule $XY \Rightarrow Z$ can be derived from rules $X \Rightarrow Z$ and $X \Rightarrow Y$.

Similarly, rule $Z \Rightarrow XY$ can be derived from rules $Z \Rightarrow X$ and $X \Rightarrow Y$ and its confidence and support are the same as those of rule $Z \Rightarrow X$.

Consequently, $XY \Rightarrow Z$ and $Z \Rightarrow XY$ are derivable. □

It follows that

Lemma 2.4 *Every rule that is redundant by [111] (Theorem 5 and Theorem 6) is derivable.*

Proof Detailed in 2.10 Appendix. □

Hence, a non-redundant rule set excludes all derivable association rules given in Lemma 2.3. By comparison, the informative rule set excludes at least all derivable rules given in lemma 2.3.

Firstly, all derivable rules given in Lemma 2.3 are omitted by the informative rule set. Since the confidence of rule $XY \Rightarrow Z$ is not greater than that of a more general rule $X \Rightarrow Z$, it is omitted by the informative rule set. It is clear that rule $Z \Rightarrow XY$ is omitted as well.

Secondly, the informative rule set excludes more than those derivable rules. For example, given a small transaction set: $\{\{1 : X, c_1\}, \{2 : X, c_1\}, \{3 : Y, c_1\}, \{4 : Y, c_1\}, \{5 : X, Y, c_1\}, \{6 : X, Y, c_1\}, \{7 : X, Y, c_1\}, \{8 : X, Y, c_1\}, \{9 : X, Y, c_1\}, \{10 : X, Y, c_2\}\}$, which has in total 10 transactions. We have the following five rules: $X \Rightarrow c_1(conf = 0.88)$, $Y \Rightarrow c_1(conf = 0.88)$, $XY \Rightarrow c_1(conf = 0.83)$, $X \Rightarrow Y(conf = 0.75)$, and $Y \Rightarrow X(conf = 0.75)$. Rule $XY \Rightarrow c_1(conf = 0.83)$ is omitted by the informative rule set, but not by the non-redundant rule set.

In fact, all derivable rules have something to do with 100% confidence rules, and these rules are not very common in a rule set generated from a transaction database. So, a non-redundant rule set cannot exclude many rules from an association rule set generated from transaction databases.

There is another type of derivable rule, the transitivity rule. For example, if both $a \Rightarrow b$ and $b \Rightarrow c$ are 100% confidence rules, then $a \Rightarrow c$ must be a 100% confidence rule and its support is the same as $a \Rightarrow b$. Hence, $a \Rightarrow c$ is derivable. Further, rule $c \Rightarrow a$ is derivable. This is because its confidence equals to $\text{conf}(c \Rightarrow b) \times \text{conf}(b \Rightarrow a)$ and its support is the same as that of $b \Rightarrow a$.

The informative rule set does not exclude these transitive rules while the non-redundant rule set excludes them. However these transitive rules are rare since two consecutive 100% confidence rules are involved. In a rule set generated from a transaction database, there are few transitive rules, so their effect on the size of a rule set can be ignored. For example, in our experiments, there is no such transitive rule generated. Hence, an informative rule set is a subset of a non-redundant association rule set.

2.4 Upward closure properties

Most efficient association rule mining algorithms use the upward closure property of infrequent itemsets: if an itemset is infrequent, so are all its supersets. Hence, many infrequent itemsets are prevented from being generated in association rule mining, and this is the essence of Apriori. If we have similar properties of the rules omitted by the informative rule set, then we can prevent generation of many rules omitted by the informative rule set. As a result, algorithm based on the properties will be more efficient.

First of all, we discuss a property that will facilitate the following discussions. It is convenient to compare support of itemsets in order to find subset relationships among their tidsets. This is because we always have support information when mining association rules. We have a relationship for this purpose.

Lemma 2.5 *$t(X) \subseteq t(Y)$ if and only if $sup(X) = sup(XY)$.*

Proof We firstly prove the forward relationship.

Since $t(X) \subseteq t(Y)$, $sup(XY) = |t(XY)|/|D| = |t(X) \cap t(Y)|/|D| = |t(X)|/|D| = sup(X)$.

We then prove the backward relationship.

Since $sup(X) = sup(XY)$, we have that $|t(X)| = |t(X) \cap t(Y)|$. Hence, the only possibility is $t(X) \subseteq t(Y)$.

In summary, $t(X) \subseteq t(Y)$ if and only if $sup(X) = sup(XY)$. □

We present two upward closure properties for mining the informative rule set, which are shown as the following two lemmas. It is clear that they are easy to use in algorithm design.

Lemma 2.6 *If $sup(X) = sup(XY)$, then for any Z, rule $XY \Rightarrow Z$ and all more specific rules do not occur in the informative rule set.*

Proof Since $sup(X) = sup(XY)$, we have $t(X) \subseteq t(Y)$. As a result, $XY \Rightarrow Z$ is derivable by Lemma 2.3, and hence is omitted by the informative rule set.

Furthermore, $t(XX') = t(XX'Y)$ holds for any X'. We have $sup(XX') = sup(XX'Y)$. Similarly, rule $XX'Y \Rightarrow Z$ is omitted by the informative rule set.

Consequently, rule $XY \Rightarrow Z$ and all other more specific rules are omitted by the informative rule set. □

It is clear that this lemma is for those derivable rules defined by Lemma 2.3.

Lemma 2.7 *If $sup(X, Z) = sup(XY, Z)$, then rule $XY \Rightarrow Z$ and all more specific rules do not occur in the informative rule set.*

Proof Since $sup(X, Z) = sup(XY, Z)$, we have $t(X, Z) \subseteq t(Y, Z)$. As a result, $XY \Rightarrow Z$ is omitted by the informative rule set by Lemma 2.2.

Furthermore, $t(XX', Z) = t(XX'Y, Z)$ holds for any X'. We have $sup(XX', Z) = sup(XX'Y, Z)$. Similarly, rule $XX'Y \Rightarrow Z$ is omitted by the informative rule set

Consequently, rule $XY \Rightarrow Z$ and all rules that are more specific must be omitted by the informative rule set. □

Clearly, this lemma is for those rules defined by Lemma 2.2.

Finally, we discuss the relationship between the two lemmas. If $sup(X) = sup(Xz)$, then $sup(X, Y) = sup(Xz, Y)$ for all Y. However, the reverse relationship does not hold. Hence, Lemma 2.7 is more general than Lemma 2.6. As a result, we can omit more rules by Lemma 2.7 than by Lemma 2.6. Lemma 2.6 is actually for derivable rules, which are a part of rules omitted by the informative rule set.

These two lemmas enable us to prune unwanted rules in a "forward" fashion before they are actually generated. In fact we can prune a set of rules when we prune each rule that is not in the informative rule set in the early stages of the computation. This allows us to construct efficient algorithms to generate the informative rule set.

2.5 Generating the informative rule set

2.5.1 Basic idea and storage structure

We proposed a direct algorithm to generate the informative rule set. Instead of first finding all frequent itemsets and then forming rules, the proposed algorithm generates the informative rule set directly. An advantage of a direct algorithm is that it avoids generating many frequent itemsets that lead to rules omitted by the informative rule set.

The proposed algorithm is a level-wise algorithm, which searches for rules from antecedent of 1-itemset to antecedent of l-itemset level by level. In each level, we select qualified rules, which may be in the informative rule set, and prune those unqualified rules. The efficiency of the proposed algorithm is based on the fact that a number of rules omitted by the informative rule set are prevented from being generated once a more general rule is pruned by Lemma 2.6 or 2.7. Consequently, the search space is reduced after each level's pruning. The number of pases over the database is bounded by the length of the longest rule in the informative rule set plus one.

In the proposed algorithm, we extend a set enumeration tree [92] as the storage structure, called a *candidate tree*. A simplified candidate tree omitting the target set is illustrated in Figure 2.1. The tree in Figure 2.1 is completely expanded, but in practice only a small part is expanded. We note that each set in the tree is unique and hence is used to identify the node, called the *identity set*. We also note that labels are locally distinct with each other under the same parent node in a layer, and labels along a path from the root to the node form exactly the identity set of the node. This is very convenient for retrieving the itemset and counting its frequency. In our algorithm each node is used to store a set of rule candidates.

2.5.2 The algorithm

The set of all items is used to build a candidate tree. A node in the candidate tree stores two sets $\{A, Z\}$. A is an itemset, the identity set of the node, and Z is a subset of the

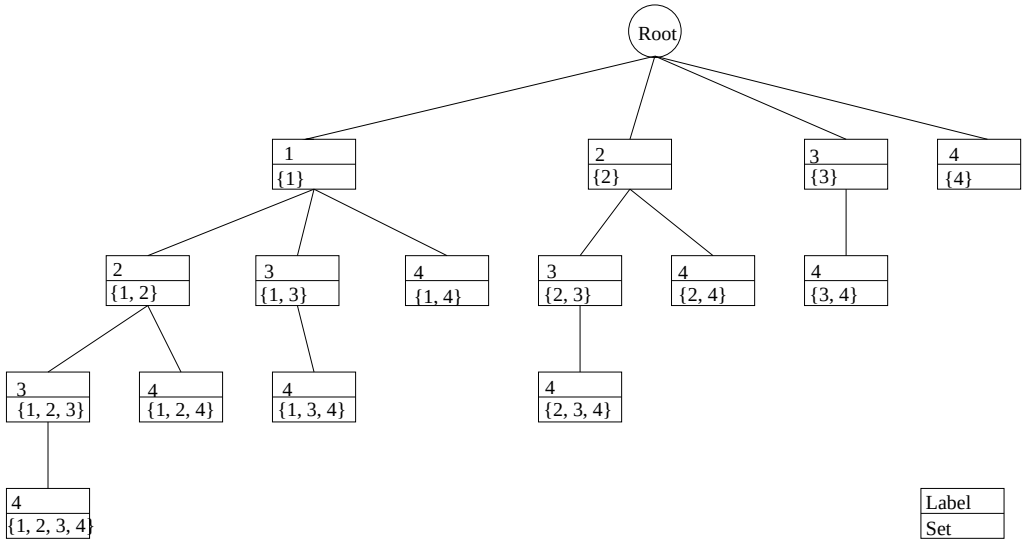


Figure 2.1: A fully expanded candidate tree over the set of items $\{1, 2, 3, 4\}$

identity itemset, called the potential target set where each item can be the consequence of an association rule. For example, $\{\{abc\}, \{ab\}\}$ is a set of candidates of two rules, namely, $bc \Rightarrow a$ and $ac \Rightarrow b$. It is clear that the potential target set is initialized by the itemset itself because all items are potential consequences of rules. When there is a case satisfying Lemma 2.7, for example, $sup(a, c) = sup(ab, c)$, then we remove c from the potential target set, and accordingly all rules such as $abX \rightarrow c$ cannot be generated afterwards.

We firstly illustrate how to generate a new candidate node. For example, if we have two sibling nodes $\{\{abc\}, \{ab\}\}$ and $\{\{abd\}, \{ad\}\}$, then the new candidate is $\{\{abcd\}, \{ad\}\}$ as a child node of $\{\{abc\}, \{ab\}\}$, where $\{ad\} = (\{ab\} \cup \{d\}) \cap (\{ad\} \cup \{c\})$. Here $\{c\}$ and $\{d\}$ are labels of two nodes. Given that $abcd$ is frequent, only candidate rules $bcd \Rightarrow a$ and $abc \Rightarrow d$ might be in the informative rule set.

We then show how to remove unqualified candidates. One way is by the frequency requirement. For example, if $sup(abcd) < \sigma$, then we remove the node whose identity set is $abcd$, called node $abcd$. Note that here a node in the candidate tree contains a set of candidate rules, and consequently a set of candidate rules are removed. Another method is by the properties of the informative rule set, and we illustrate it in two ways.

Firstly, consider a candidate node $\{A^l, Z\}$ where A^l means that A^l is a l-itemset. For an item $z \in Z$, when there is $sup((A^l \backslash z), z) = sup((A^{l-1} \backslash z), z)$ for $(A^l \backslash z) \supset (A^{l-1} \backslash z)$, then remove the z from Z by Lemma 2.7. Secondly, we say node $\{A^l, Z\}$ is *restricted* when there is $sup(A^l) = sup(A^{l-1})$ for $A^l \supset A^{l-1}$. A restricted node does not extend its potential target set and keeps it as that of node $\{A^{l-1}, Z\}$. This is because that all rules $A^{l-1}X \Rightarrow c$ for any X and c are omitted from the informative rule set by Lemma 2.6, and hence we need not generate such candidates. This potential target set is removable by Lemma 2.7, and a restricted node is *dead* when its potential target set is empty. All supersets of the itemset in a dead node are unqualified candidates, so we need not generate them.

We give the top level of the informative rule mining algorithm in the following.

Algorithm 2.1 Informative rule set generator

Input: Database D, the minimum support σ and the minimum confidence ψ.

Output: The informative rule set R.

```
(1)   set R = ∅
(2)   count support of 1-itemsets
(3)   initialize candidate tree T
(4)   generate new candidates as leaves of T
(5)   while (new candidate set is non-empty)
(6)       count support of the new candidates
(7)       prune the new candidate set
(8)       include qualified rules from T to R
(9)       generate new candidates as leaves of T
(10)  return rule set R
```

The first 3 lines are general description that are self-explanatory. We will elaborate the two functions, Candidate generator in line 4 and 9 and Pruning in line 6. They are listed as follows.

First of all, we introduce some notation in the functions: n_i is a candidate node in the candidate tree, labelled by an item (vertex) i_{n_i}, contains an identity itemset A_{n_i} and a potential target set Z_{n_i}; T_l is the l-th level of candidate tree; $\mathcal{P}^l(A)$ is the set of all l-subsets of A; n_A is a node whose identity itemset is A. All items are in lexicographic order.

Function Rule candidate generator

(1) for each node $n_i \in T_l$

(2) for each sibling node n_j $(i_{n_j} > i_{n_i})$

(3) generate a new candidate node n_k as a son of n_i such that

//Combining

(4) $A_{n_k} = A_{n_i} \cup A_{n_j}$

(5) $Z_{n_k} = (Z_{n_i} \cup i_{n_j}) \cap (Z_{n_j} \cup i_{n_i})$

//Pruning

(6) if $\exists A \in \mathcal{P}^l(A_{n_k})$ but $n_A \notin T_l$ then remove n_k

(7) else if n_A is restricted then mark n_k restricted and let $Z_{n_k} = Z_{n_A} \cap Z_{n_k}$

(8) else $Z_{n_k} = (Z_{n_A} \cup (A_{n_k} \backslash A)) \cap Z_{n_k}$

(9) if n_k is restricted and $Z_{n_k} = \emptyset$, remove node n_k

We generate the $(l+1)$-layer candidates from the l layer nodes. Firstly, we combine a pair of sibling nodes and insert their combination as a new node in the next layer. Secondly, if any of its l-sub itemset is not a frequent itemset then we remove the node. If an item is not qualified to be the target of a rule in the informative rule set, then we remove the target from the potential target set.

Note that in line 6, not only a superset of an infrequent itemset is removed, but also a superset of a frequent itemset in a dead node is removed. The former case is common in association rule mining, and the latter case is unique for the informative rule mining. A dead node is removed in line 9. Accordingly, in the informative rule mining, we need not generate all frequent itemsets.

Function Pruning

(1) for each $n_i \in T_{l+1}$

(2) if $sup(A_{n_i}) < \sigma$, remove node n_i and return

(3) if n_i is not restricted node, do

(4) if $\exists n_j \in T_l$ for $A_{n_j} \subset A_{n_i}$ such that $sup(A_{n_j}) = sup(A_{n_i})$
then mark n_i restricted and let $Z_{n_i} = Z_{n_i} \cap Z_{n_j}$ // Lemma 2.6

(5) for each $z \in Z_{n_i}$

(6) if $\exists n_j \in T_l$ for $(A_{n_j} \backslash z) \subset (A_{n_i} \backslash z)$ such that $sup((A_{n_j} \backslash z), z) = sup((A_{n_i} \backslash z), z)$
then $Z_i = Z_i \setminus z$. // Lemma 2.7

(7) if n_i is restricted and $Z_{n_i} = \emptyset$, remove node n_i

We prune a rule candidate from two aspects, frequency requirement for association rules and qualification requirement for the informative rule set. The method for pruning infrequent rules is the same as that of a general association rule mining algorithm. As for the method in pruning unqualified candidates for the informative rule set, we restrict the possible targets in the potential target set of a node (a possible target is equivalent to a rule candidate) and remove a restricted node when its potential target set is empty.

2.5.3 Correctness and efficiency

Lemma 2.8 *Algorithm* 2.1 *generates the informative rule set correctly.*

Proof We will prove the claim from two steps. One is that the candidate tree can generate all single consequence association rules directly, and the other is that the pruned rules are those which must be omitted by the informative rule set.

Basically, a candidate tree can enumerate all subsets of the set of all items, and stores every itemset in a node of the tree as the identity set of the node. The itemset stored in a child node is a superset of the itemset stored in its parent node, so a set of supersets are stored in a branch of the tree. Once we have removed those infrequent branches, all nodes left store frequent itemsets. Let the potential target set of an itemset be the itemset itself, then we can obtain all single consequence association

rules directly.

Now, we will prove that all pruned rule candidates are those which must be omitted by the informative rule set from three steps.

Firstly, in our algorithm, the potential target set is a subset of the itemset stored in a node, $Z \subseteq A$, and some items are omitted from set Z by Lemma 2.7. Specifically, if $sup(A, z) = sup(A', z)$ for $A' \supset A$ then all rules $A'' \Rightarrow z$ for $A'' \supseteq A'$ are omitted from the informative rule set. Hence, we can remove all rule candidates $A'' \Rightarrow z$, and equivalently, remove z from every potential target set of every node in the subtree rooted by node $n_{A'}$ in the algorithm.

Secondly, for all restricted nodes, we do not expand their potential target sets while expanding their itemset. Since $sup(A) = sup(A')$ for $A' \supset A$, all rules $A'' \Rightarrow c$ for $A'' \supseteq A'$ and any c are omitted from the informative rule set by Lemma 2.6. Given a restricted node A'' where $A'' \supset A$ and $sup(A'') = sup(A)$, all rules $A''\backslash z \Rightarrow z$ where $z \in \{A''\backslash A\}$ must be omitted from the informative set. The potential target set Z'' for node $n_{A''}$ must be a subset of A, and hence we need not expand Z''.

Finally, we do not generate a candidate node that stores a superset of the identity set in a dead node. We know that the potential target set of a restricted node A' is only a subset of A where A is the smallest subset of A' such that $sup(A) = sup(A')$. If all items in Z cannot be qualified consequences of A', then A' and all its supersets cannot contain rules in the informative rule set.

In summary, the algorithm generates the informative rule set correctly. □

It is very hard to give a closed form analysis of the algorithm. However, we expect improvements over other association rule mining algorithms for the following reasons. Firstly, it does not generate all frequent itemsets, because some frequent itemsets cannot contain rules in the informative rule set. Secondly, it does not test all possible rules in each generated frequent itemset because some items in an itemset are not qualified as consequences for rules in the informative rule set.

The number of passes over a database is bounded by the length of longest rule in the informative rule set plus one.

2.6 Experimental results

In this section, we show that the informative rule set is significantly smaller than both the association rule set and the non-redundant association rule set. We further show that it can be generated more efficiently with fewer passes over a database. Finally, we show that the efficiency improvement are gained from the fact that the proposed algorithm for the informative rule set accesses the database fewer times and generates fewer candidates than Apriori for the association rule set.

Since the informative rule set contains only single target rules, for a fair comparison, the association rule set and the non-redundant rule set in this section contain only single target rules as well. The reason for the comparison with the non-redundant rule set is that the non-redundant rule set can make the same predictions as the association rule set.

The two test transaction databases, T10.I6.D100K.N2K and T20.I6.D100K.N2K, are generated by the synthetic data generator from QUEST of IBM Almaden research center. Both databases contain 1000 items and 100,000 transactions. We chose the minimum support in the range such that 70% to 80% of all items are frequent, and fixed the minimum confidence to 0.5.

Sizes of different rule sets are listed in Figure 2.2. It is clear that the informative rule set is significantly smaller than both the association rule set and the non-redundant rule set. The size difference between an informative rule set and an association rule set becomes more evident when the minimum support decreases, as does the size difference between an informative rule set and a non-redundant rule set. This is because the length of rules becomes longer when the minimum support decreases, and long rules are more likely to be omitted by the informative rule set than short rules. By our discussion in Section 2.4, we know that all redundant rules are connected with at least one 100% confidence rule. However, in these randomly generated databases, there are not many 100% confidence rules. Hence there is little size difference between an association rule set and a non-redundant rule set. As a result, in the following comparisons, we only compare the informative rule set with the association rule set.

Now, we shall compare efficiencies of generating the informative rule set and the association rule set. We implemented Apriori on the same data structure as the proposed algorithm and generated only single target association rules. Our experiments were conducted on a Sun server with two 200 MHz UltraSPARC CPUs.

The times for generating association rule sets and informative rule sets are listed in Figure 2.3. We can see that the proposed algorithm for mining an informative rule set is more efficient than Apriori for mining a single target association rule set. This is because the proposed algorithm does not generate all frequent itemsets, and does not test all items as targets in a frequent itemset. The improvement of efficiency becomes more evident when the minimum support decreases. This is consistent with the deduction of rules being omitted from an association rule set as shown in Figure 2.2.

Further, the number of times to access a database of proposed algorithm is smaller than Apriori, as shown in Figure 2.4. This is because the proposed algorithm avoids generating many long frequent itemsets that contain no rules in an informative rule set. From the results, we also know that long rules are easier to be omitted by an informative rule set than short rules. Clearly, this number is significantly smaller than the number of frequent itemsets which are needed to access a database in other direct association rule generating algorithms.

To better understand efficiency improvement of the proposed algorithm over Apriori, we list the number of nodes in a candidate tree for generating both association and informative rule sets in Figure 2.5. The numbers are all frequent itemsets for Apriori to generate all association rules and partial frequent itemsets for the proposed algorithm to generate an informative association rule set. We can see that in mining the informative rule set, the searched itemsets is less than all frequent itemsets for forming all association rules. So, this is the reason for efficiency improvement and reduction in the number of times to access a database.

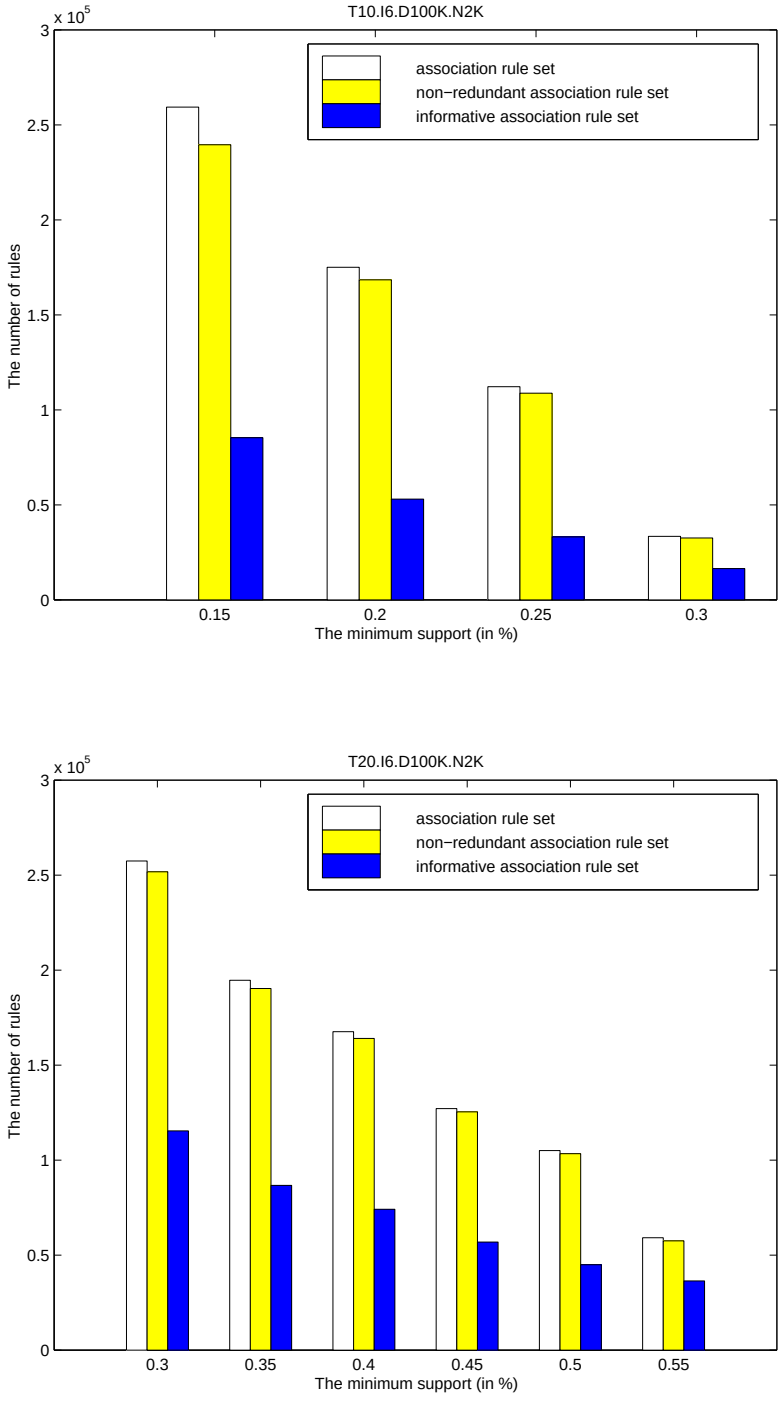


Figure 2.2: The comparison of sizes of association rule sets, non-redundant association rule sets and informative association rule sets

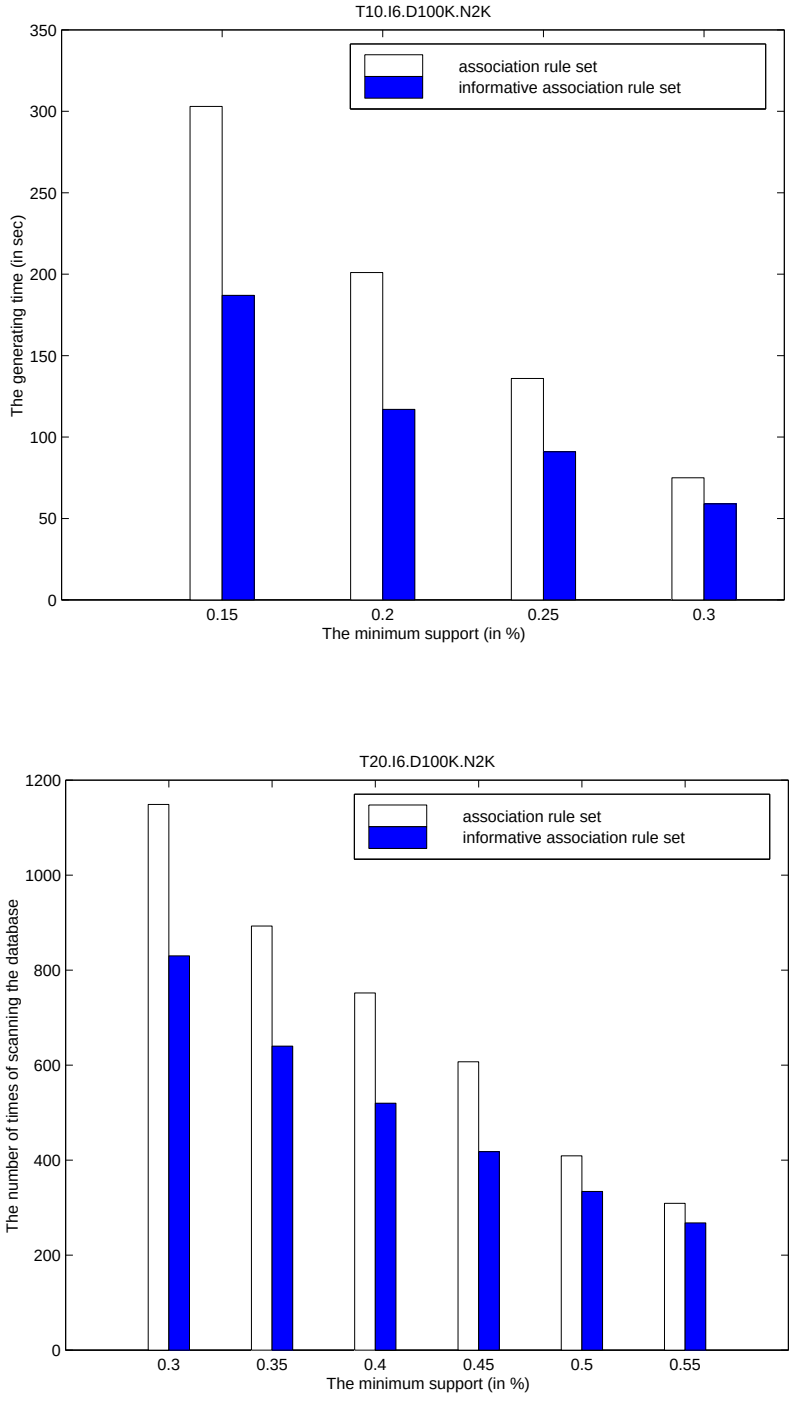


Figure 2.3: The comparison of generation time between association rule sets and informative association rule sets

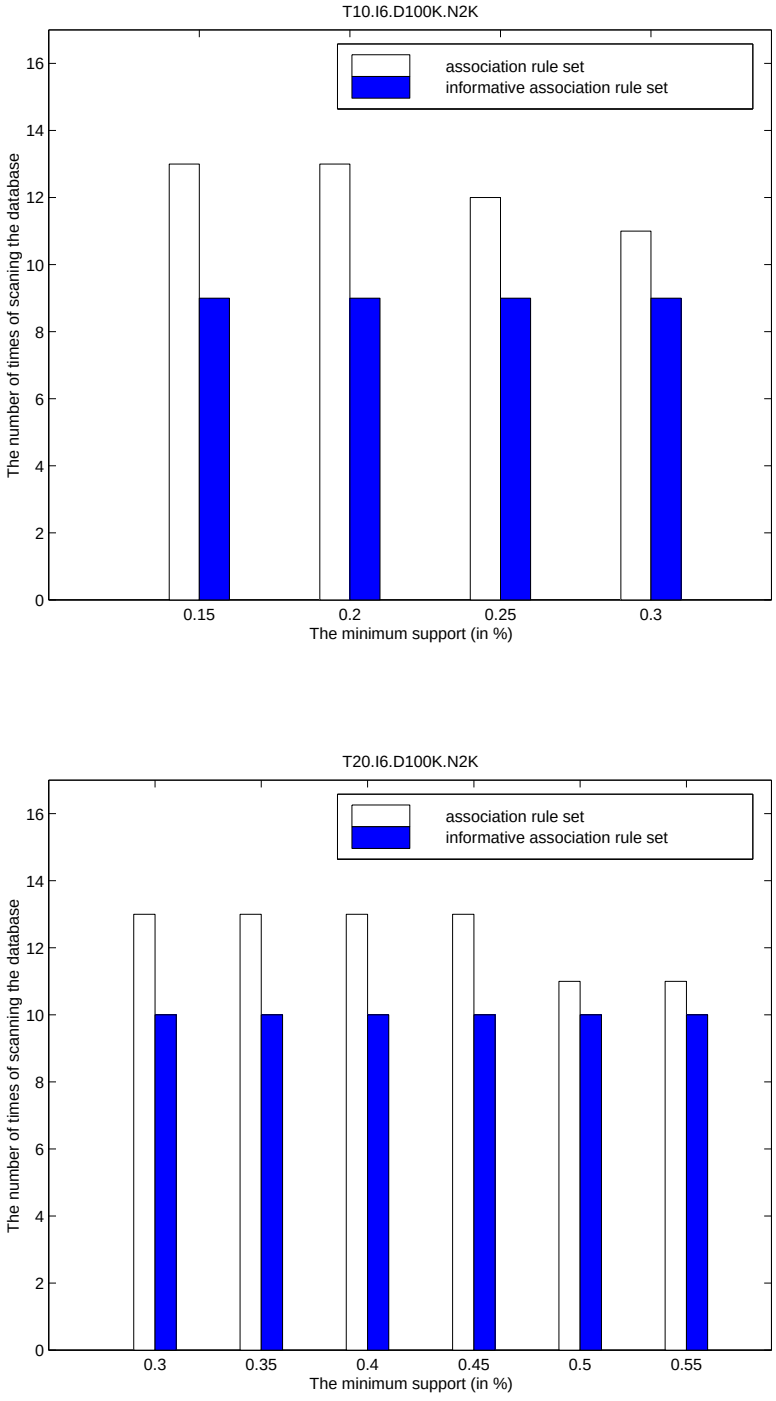


Figure 2.4: The comparison of passes over databases (IO) for generating association rule sets and informative association rule sets

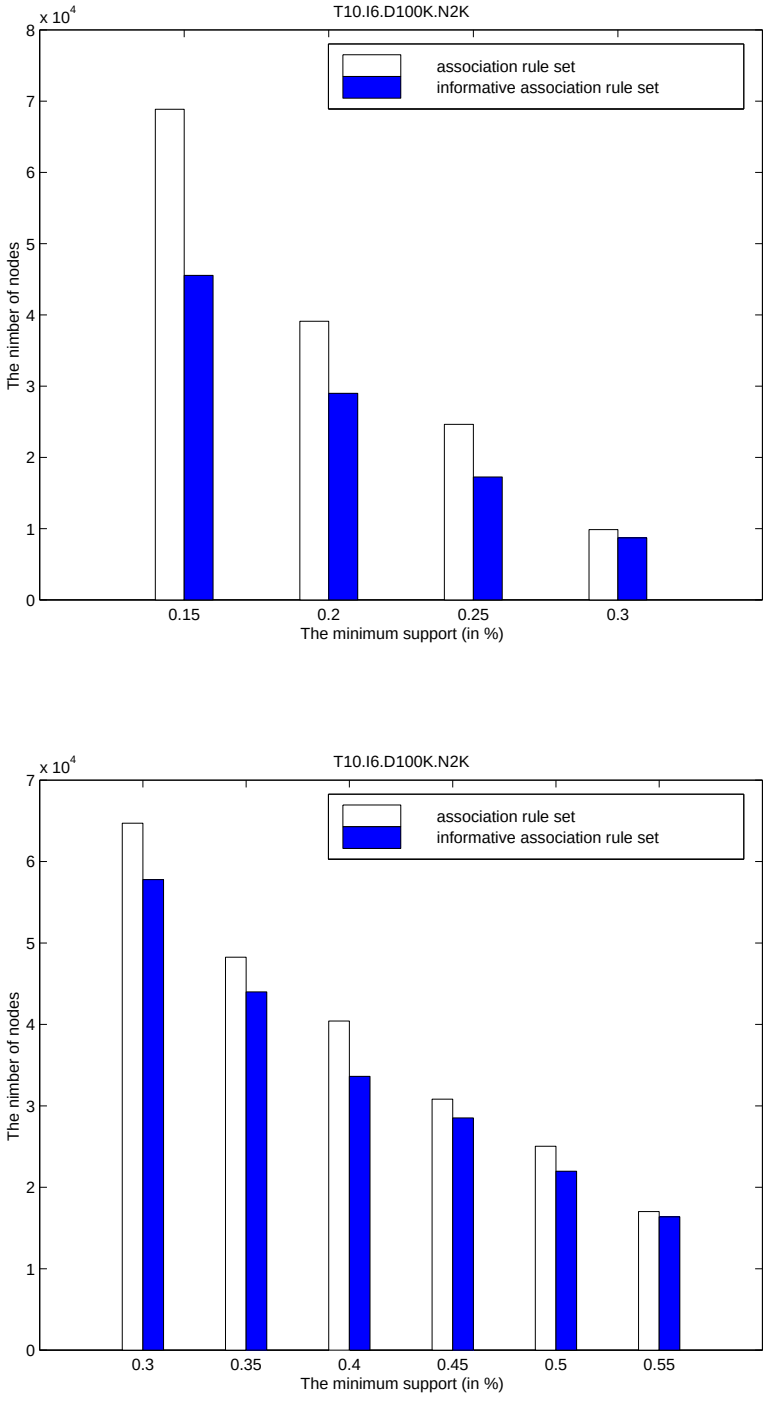


Figure 2.5: The comparison of candidate number for generating association rule sets and informative association rule sets

2.7 Discussion

2.7.1 Justification for confidence priority prediction model

In this part we will present discussions on why we choose the confidence priority model for prediction. Clearly, it is an extension of a classification model by allowing a set of items to be a prediction. Reasons for using confidence priority are listed as follows.

Firstly, confidence is the accuracy of a rule based on the data from which it is generated, and naturally, we prefer more accurate rules.

Secondly, in the estimation of true accuracy for a rule, both the support and the confidence of a rule have effects. In a large database, a small support (ratio) may be connected with a large number of transactions (an absolute number). When the absolute support number is large, the effect from the support on the true accuracy of a rule is very small. Further, when the confidence of a rule is high, the effect from the support on the true accuracy of a rule is very limited. Consequently, confidence approximates to the true accuracy of a rule in a large database with a high minimum confidence. Both requirements are usually satisfied in practice.

Thirdly, the predictions provided by the confidence priority model will not significantly affected by the changing of minimum confidence. In the confidence priority model, each prediction is made by a rule with the maximum confidence, and hence the distance to the minimum confidence is maximized. As a result, the change of the minimum confidence would not significantly affect the prediction.

Alternatively, we may have a support priority model. The support priority model is one to select a matched rule with the maximum support to make prediction on an input record. It reflects the emphasis on the popularity of a prediction.

Consider a prediction from the support priority model as a sequence of items as well. Clearly, for an input itemset, the prediction sequence from the informative association rule set is identical to the prediction sequence from the association rule set. This is because all highest support rules are in the informative association rule set. In fact, to generate the same prediction sequence as the association rule set, the support priority

model only needs a subset of informative rule set. The rule set is smaller, but it loses the highest confidence information which is crucial in predictions.

We may have a third option, which is to choose a maximum (in length) matching rule on an input record to make prediction. This model reflects the maximal utilization of the input information. However, it is clear that the length of long rules is subject to the choice of both the minimum confidence and the minimum support. Hence, this model that is too sensitive to input thresholds to be practical.

Consequently, we use the confidence priority model in this chapter. The resulting informative rule set contains the highest confidence rules as well as the highest support rules, so it is suitable for various applications.

2.7.2 Level-wise algorithms vs. other algorithms

In this chapter, we only compare our algorithm with Apriori, a level-wise association rule mining algorithm. In this part, we will show that the other fast algorithms may require significantly more memory than Apriori and hence may be unworkable when the minimum support is very small. In contrast, our proposed algorithm has no additional memory requirement.

In association rule mining, a cost step is one that counts frequency of itemsets. A straightforward way is to count them directly. When counters for all possible itemsets fit in main memory, it accesses a database only once. However, the problem is that there usually are too many possible itemsets to fit in main memory. For example, the number of all possible itemsets in a transaction database with m items is up to 2^m.

Apriori [4, 3] uses a level-wise method, where a level l candidate must be a union of two level $l-1$ frequent itemsets. However, Apriori accesses a database as many times as the length of the maximum frequent itemset plus one, and this accounts for much mining time.

Many efficient algorithms are proposed that access a database fewer passes than Apriori to improve efficiency [98, 114, 42]. In this section, we will compare a typical two-pass algorithm [114, 98, 57] with Apriori to show that the number of candidates in

a two-pass algorithm can be very large when the minimum support is low and hence may be impractical when the minimum support is very small.

In a two-pass algorithm, a frequent itemset candidate is a union of 2-frequent itemsets. Hence, the number of passes over a database can be as small as 2, one for finding all frequent 2-itemsets and the other for verifying frequent itemsets from all union sets of those two frequent itemsets. We implemented Apriori and a two-pass algorithm in the same storage structure, prefix tree (or set enumeration tree), and compared their performances. Here, we omit all details [57] here and directly present comparison results between the number of accessing times, candidate number and generating times of the two methods.

We can see as shown in Figure 2.6 that the number of accessing times for a two-pass algorithm is significantly fewer than Apriori, while the memory usage represented by the number of candidates is significantly larger when the minimum support is small as shown in Figure 2.7. Overall, their performances are shown in Figure 2.8. When the minimum support is not so small, a two-pass algorithm is more efficient than Apriori, but worse otherwise. This is a result of a sharp increase in candidate number for a two-pass algorithm when the minimum support is small. In experiments, the two-pass algorithm ran out of memory when the minimum supports were less than 0.1% and 0.55% respectively. In contrast, Apriori worked well when the minimum supports were as small as 0.05% and 0.25%. Hence, efficiency improvement of a two-pass algorithm is at the cost of large memory consumption.

Similarly, other one-phase algorithms [42, 1], which project the whole database to a tree, have more candidates than a two-pass algorithm since every unique occurred itemsets in a database is a candidate. Hence, the memory requirement for a one-phase algorithm is larger than that of a two-pass algorithm.

In this thesis, we use level-wise algorithms. Our proposed algorithms accesses a database less often than Apriori and require less memory because of the utilization of additional upward closure properties.

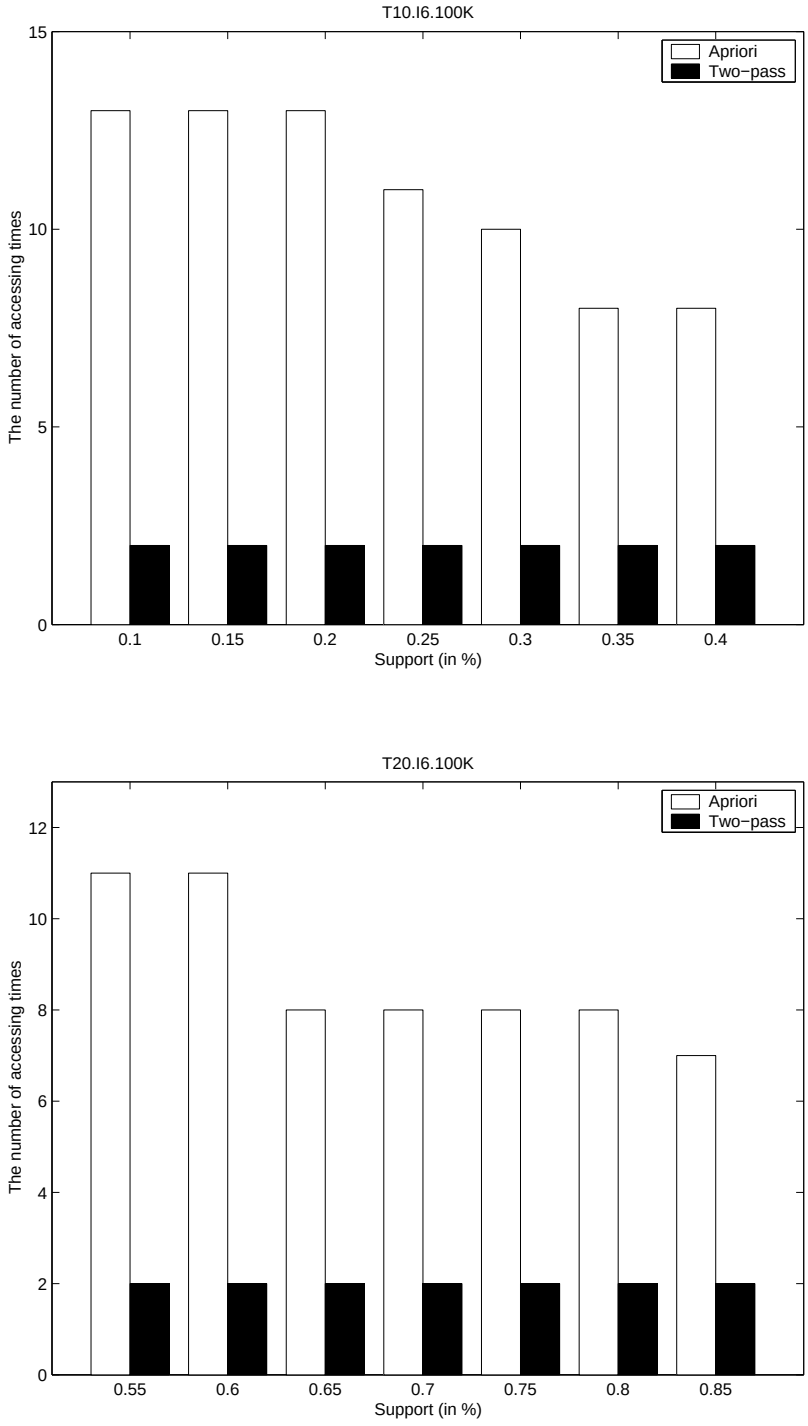


Figure 2.6: The comparison of passes over databases (IO) between Apriori and a two-pass algorithm

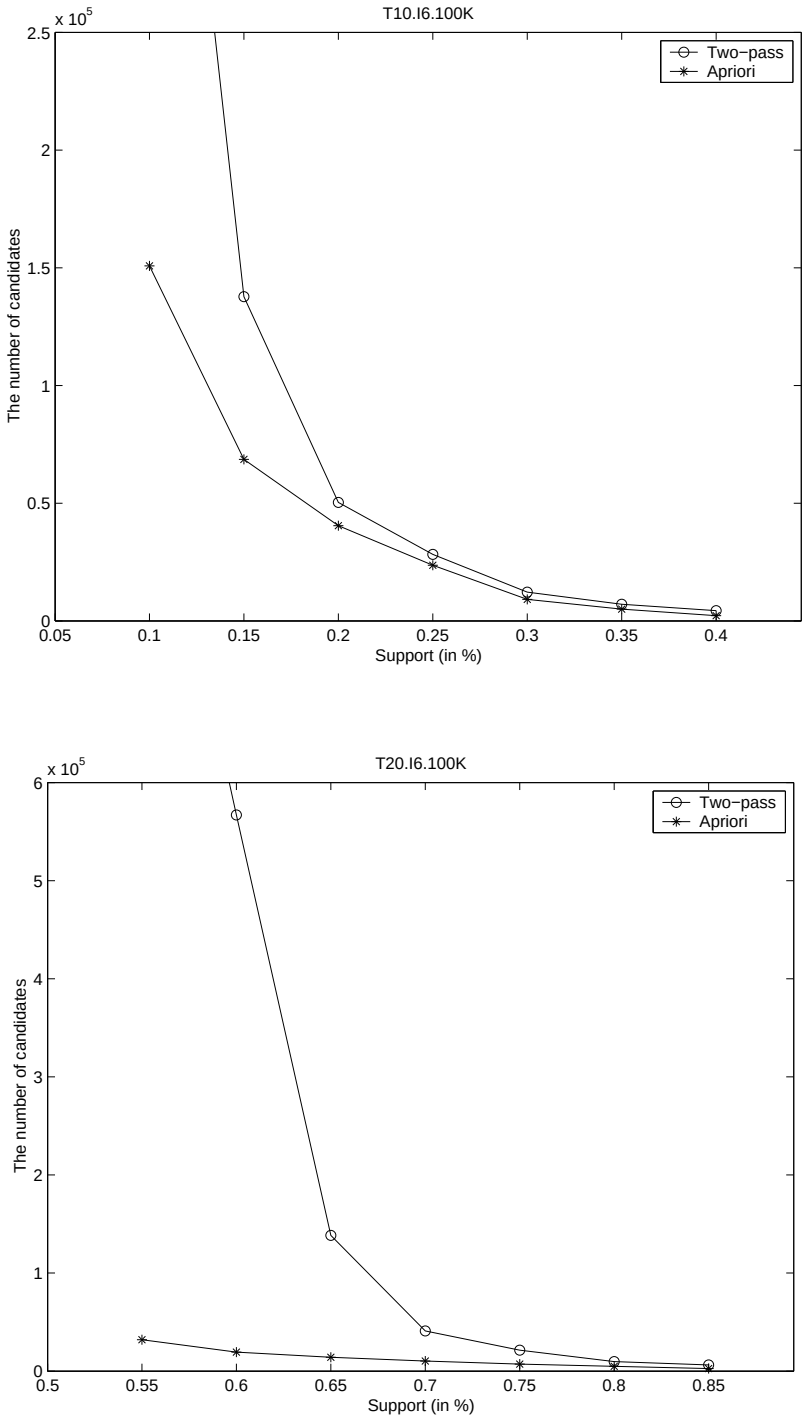


Figure 2.7: The comparison of candidate number between Apriori and a two-pass algorithm

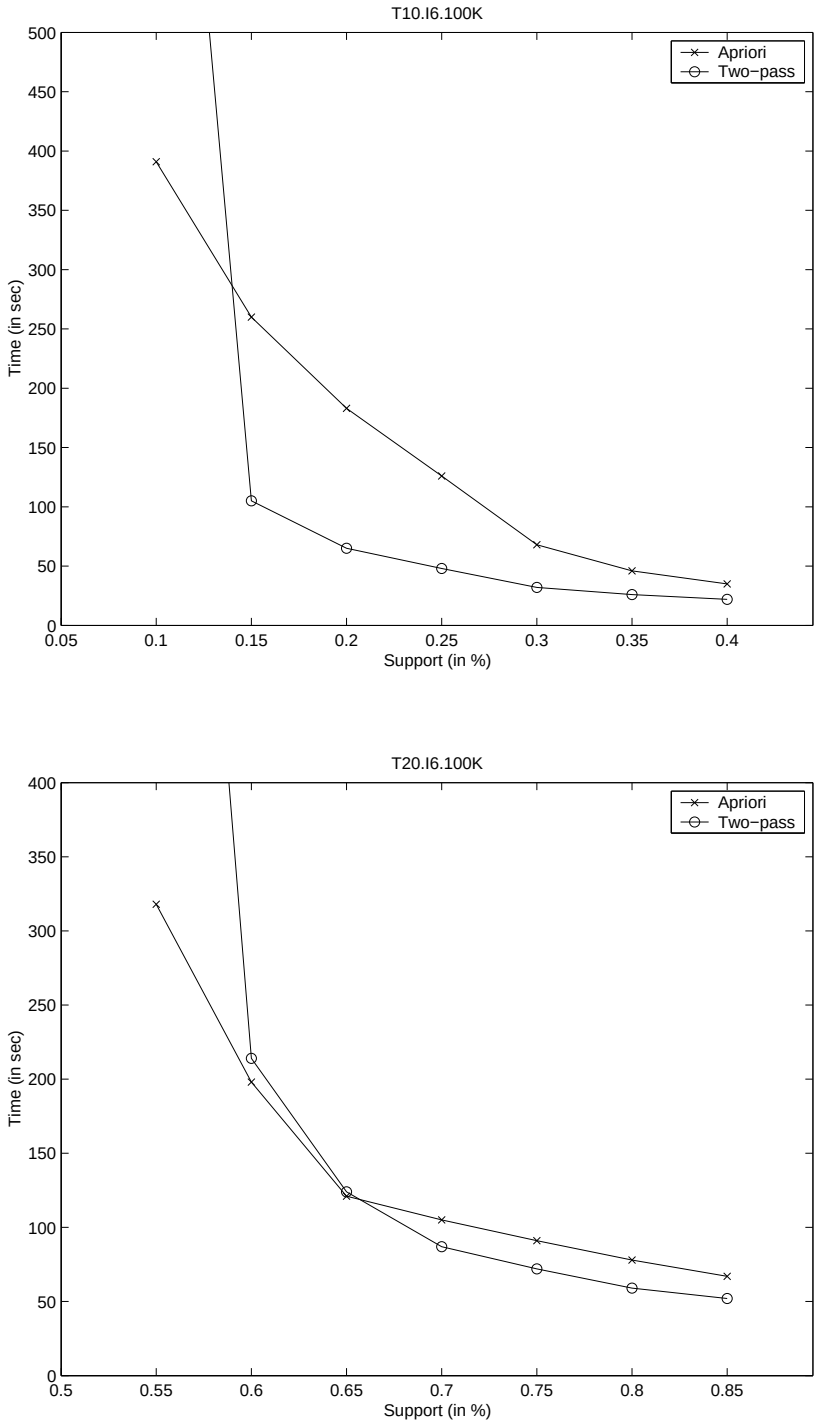


Figure 2.8: The comparison of generation time between Apriori and a two-pass algorithm

2.8 Dealing with numerical attributes

Mining quantitative association rules is an important topic of data mining since most real world databases have both numerical and categorical attributes. There have been many proposals for mining quantitative association rules [37, 55, 106, 29]. However, most of them are based on supervised discretization methods, i.e. the consequences of rules are pre-specified, In the unconstrained association rule mining, we may not know which items or itemsets will be consequences and all are potential consequences. Hence, a supervised discretization method may not be applicable. There are two typical unsupervised discretization methods, equal width and equal depth. The latter has been used in quantitative association rule mining [37, 78, 102]. Actually, they consider only the width of intervals or the density over intervals, and in this section we will present an adaptive generalized method considering both.

2.8.1 A new criterion

Let us have a look at what is problem with equal-width and equal-depth discretization methods.

1. Equal-width discretization divides a continuous attribute range into N intervals of equal width. For example, ages from 20 to 60 can be divided into four intervals of width 10 years. This method is easy to be implemented, but has a clear drawback that there may be too few instances in some intervals and too many in other intervals, and both cases hinder mining high quality association rules.

2. Equal-depth (or equal-cardinality) discretization divides a continuous attribute range into N intervals so that there are $1/N$ of the total instances in each interval. This method avoids the possible imbalance partition as in the equal-width discretization, but naturally distributed data may not so even and some intervals may contain more attribute values while others less.

Problems of two discretization methods are that they only consider either interval width or instance density. We will present a method that considers both.

Our work is motivated by the concept of clustering [56]. It initially places each numerical attribute value in a separate interval, and then selectively merges similar adjacent intervals. It uses a merging criterion that considers both value densities and value distances of numerical attributes, and produces proper value density and suitable interval width so that association rules can be easily found from them. After numerical attributes are discretized as a set of disjoint intervals, each interval can be interpreted as an item, thus transforming quantitative association rule mining into general association rule mining.

The goal of our work is to partition a numerical attribute into a set of disjoint intervals with a suitable number of attribute values in each interval. Intervals with too few attribute values may prevent itemsets from having sufficient support and intervals with too many attribute values would fail to discriminate between attribute values and would hence fail to lead to useful association rules. Hence, we present a merging algorithm to produce a set of intervals with suitable attribute values.

Initially, suppose that a numerical attribute has m distinct values, $\{x_1, x_2, \ldots, x_m\}$. Without loss of generality we further assume that $x_i < x_{i+1}$ for all $1 \leq x \leq m-1$ (we can simply sort these values otherwise). Define m intervals $I_1, \ldots, I_m$ such that each I_i includes x_i, $1 \leq i \leq m$. Let each interval I_i have a *representative centre* c_i initially defined to be x_i with n_i instances, $1 \leq i \leq m$.

In general, suppose that an interval I contains attribute values $\{x_1, x_2, \ldots x_k\}$ and attribute value x_i for $1 \leq i \leq k$ has n_i instances. The total instances in the interval I is $N_I = \sum_{i=1}^{k} n_i$. The representative center c and the average intra-interval distance of I with respect to c to be

$$\begin{aligned} c &= \frac{1}{N} \sum_{i=1}^{k} n_i x_i \\ \text{Dist}(I, c) &= \frac{1}{N} \sum_{i=1}^{k} n_i \mid x_i - c \mid, \end{aligned} \tag{2.1}$$

Assume that two adjacent intervals $I_i = \{x_1, \ldots, x_i\}$ and $I_j = \{x_{i+1}, \ldots, x_{i+j}\}$ have representative centres $c_i = \frac{\sum_{p=1}^{i} x_p n_p}{\sum_{p=1}^{i} n_p}$ and $c_j = \frac{\sum_{p=i+1}^{i+j} x_p n_p}{\sum_{p=i+1}^{i+j} n_p}$, where attribute value x_p contains n_p instances, $1 \leq p \leq i+j \leq m$. Clearly, the number of attribute instances

in I_i is $N_i = \sum_{p=1}^{i} n_p$. and in I_j is $N_j = \sum_{p=i+1}^{i+j} n_p$. The union, $I = I_i \cup I_j$, of the two intervals containing $i + j$ attribute values and $N_i + N_j = \sum_{p=1}^{i+j} n_p$ instances in total thus has its representative centre given by the average weighted value of (c_i, n_i) and (c_j, n_j):

$$c = \frac{c_i \sum_{p=1}^{i} n_p + c_j \sum_{p=i+1}^{i+j} n_p}{\sum_{p=1}^{i+j} n_p} = \frac{c_i N_i + c_j N_j}{N_i + N_j}. \tag{2.2}$$

The intra-interval distance of I with respect to c calculated by Equation (2.1) is then uniquely defined by c_i and c_j as follows. When $x_i \leq c \leq x_{i+1}$ which holds in our algorithm, noting that $\sum_{p=1}^{i} x_p n_p = c_i N_i$ and $\sum_{p=i+1}^{i+j} x_p n_p = c_j N_j$ we can derive this distance as follows:

$$\begin{aligned}
\text{Dist}(I, c) &= \frac{1}{N_i + N_j} \sum_{p=1}^{i+j} n_p \mid x_p - c \mid \\
&= \frac{1}{N_i + N_j} (\sum_{p=1}^{i} n_p (c - x_p) + \sum_{p=i+1}^{i+j} n_p (x_p - c)) \\
&= \frac{1}{N_i + N_j} (\sum_{p=i+1}^{i+j} n_p x_p - \sum_{p=1}^{i} n_p x_p + (\sum_{p=1}^{i} n_p - \sum_{p=i+1}^{i+j} n_p) c) \\
&= \frac{1}{N_i + N_j} (c_j N_j - c_i N_i + (N_i - N_j) \frac{c_i N_i + c_j N_j}{N_i + N_j}) \\
&= \frac{2 N_i N_j}{(N_i + N_j)^2} (c_j - c_i).
\end{aligned} \tag{2.3}$$

To let our criterion be balanced between the number of instances in an interval and the interval width, we require the difference to be measured by (number in a interval) $\times$ (the interval width). Thus, we now define the difference between I_i and I_j, denoted by $\text{Diff}(c_i, c_j)$, to be proportional to be

$$\text{Diff}(c_i, c_j) = \frac{N_i N_j}{N_i + N_j} (c_j - c_i). \tag{2.4}$$

We can see that the difference between two intervals is determined not only by the distance between their representative centers but also by the number of instances in each interval. If two pairs of intervals are the same distance apart, the pair with smaller number of instances in each interval has a smaller difference than the pair with larger

number of instances in each interval. This reflects the intuition to keep high-density interval apart and merge low-density intervals together. This is because we may incur significant errors rate if we merger two high-density intervals wrongly, with insignificant errors otherwise.

2.8.2 The merging algorithm

Now we consider how to use the proposed criteria to merge values in numerical attribute. Given m consecutive intervals $I_1, \ldots, I_m$, whose representative centers are $c_1, \ldots, c_m$, there are $m-1$ pairs of adjacent intervals. We test every pair of adjacent intervals, and then merge the two with the smallest Diff, say I_k and I_{k+1}. The merged interval $I_k = I_k \cup I_{k+1}$ has representative centre c_k and number of instances N_k updated as follows.

$$\begin{aligned} c_k = &\quad (N_k c_k + N_{k+1} c_{k+1})/(N_k + N_{k+1}); \\ N_k = &\quad N_k + N_{k+1}; \end{aligned}$$

We repeatedly merge the pair of adjacent intervals with minimum difference in this way.

We now propose a criterion for terminating this merging process before we have reduced all intervals to a single large interval. If the density of each interval is large enough to form a rule, namely $N_i/N > \sigma$, or the representative centers of each pair of adjacent intervals are so far apart that they are unlikely to be in one group, for example $(c_{i+1} - c_i) > 3\overline{d}$, where $\overline{d}$ is the average distance of adjacent values of a numerical attribute, the numerical attribute merging procedure stops.

After completing the above procedure, the whole range of numerical attribute values is partitioned into a set of adjacent intervals, where the number of instances in an interval is large enough to reach the user specified minimum support or an interval is too isolated to be merged into an adjacent interval.

The above procedure may be summarised as follows.

Algorithm 2.2 Numerical attribute merging algorithm

Input: An ordered sequence of numeric attributes $\{x_1, \ldots, x_m\}$.

Output: An ordered sequence of disjoint intervals $I_1, \ldots, I_{m'}$ covering $x_1, \ldots, x_m$, $m' < m$.

(1) for each x_i $(1 \leq i \leq m)$ do
(2) let I_i contain x_i;
(3) let $c_i = x_i$ be the representative centre of I_i;
(4) let $m' = m$;
(5) count N_i for $1 \leq i < m$;
(6) for each interval pair (I_i, I_{i+1}) $(1 \leq i < m')$ do
(7) let Diff $(c_i, c_{i+1}) = \frac{N_i N_{i+1}}{N_i + N_{i+1}}(c_{i+1} - c_i)$;
(8) while (termination condition is not satisfied) do
(9) let k be such that Diff (c_k, c_{k+1}) is minimal;
(10) let $c_k = (N_k c_k + N_{k+1} c_{k+1})/(N_k + N_{k+1})$;
(11) let $N_k = N_k + N_{k+1}$;
(12) merge I_k and I_{k+1} into a new interval I_k;
(13) let $m' = m' - 1$;
(14) recompute Diff(c_{k-1}, c_k) and Diff(c_k, c_{k+1});
(15) recompute $\overline{d} = \frac{1}{m'} \sum_{i=1}^{m'-1} (c_{i+1} - c_i)$
(16) output intervals $I_1, \ldots, I_{m'}$;

Since this algorithm considers both instance densities and value distances, it has the merits of both equal-depth and equal-width methods. Let us consider two extreme cases. If the termination condition of the algorithm does not include the restriction of the density in an interval, the merging result will approximate to that of the equal width method. On the other hand, if the termination criterion has no distance restriction, then the merging result will approximate to that of the equal depth method. Therefore our proposed method is an adaptive method that generalizes the equal-depth and equal-width methods.

2.8.3 Implementation and experiments

We implemented our quantitative association rule mining algorithm and tested it on some databases from the machine learning database repository at the University of California [15]. The detailed descriptions of databases are in Appendix A. In the experiment, we did not find all rules but those whose consequences are labelled classes. However, we did not use the class information in the process of discretization. We use the local support that will introduced in the next chapter.

The quality of finding rule sets depends critically on the discretization of continuous attributes. If discretization produces a set of suitable intervals, the computed association rules will have a high overall accuracy. Equivalently, we use the overall coverage here, which is the ratio of the number of transactions identified by rules to the number of all transactions. The method proposed in this section has been evaluated by comparing its performance with equal-width and equal-depth discretization methods, where numerical attributes are partitioned into 5 or 10 equal value intervals or equal density intervals respectively. They are denoted by equal-width 5, equal-width 10, equal-depth 5 and equal-depth 10. The experimental results are displayed in Figure 2.9.

From Figure 2.9, we can see that the overall results of our numerical attribute merging method is better than others. In 24 trials, only 1 trial is obviously worse, 4 trials are marginally worse, and the other 19 trials are better than or equal to those of both equal-width and equal-depth methods. We observe that no other single method in the experiment achieves this performance.

Consequently, the method proposed is better than both equal-depth and equal-width methods.

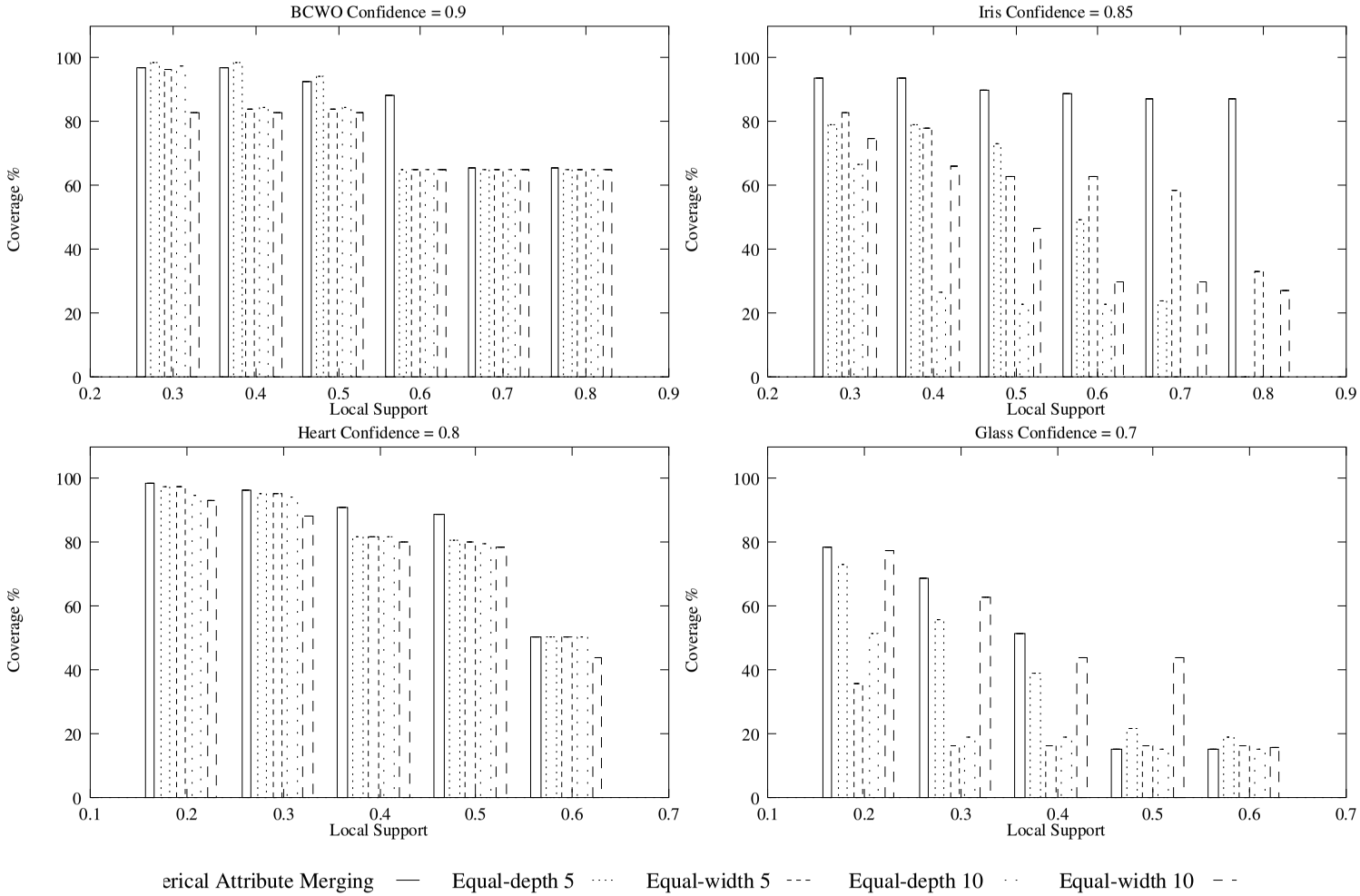


Figure 2.9: The comparison three unsupervised methods in association rule mining (equal-depth, equal-width and the proposed method)

2.9 Conclusion

We have defined a new rule set namely the informative rule set that presents prediction sequences equal to those presented by the association rule set using the confidence priority. The informative rule set is significantly smaller than the association rule set, especially when the minimum support is small. We have characterized the relationships between the informative rule set and the non-redundant association rule set, and revealed that the informative rule set is a subset of the non-redundant association rule set. We have studied the upward closure properties of informative rule set for omission of uninformative association rules, and presented a direct algorithm to efficiently generate the informative rule set without generating all frequent itemsets. The experimental results confirm that the informative rule set is significantly smaller than both the association rule set and the non-redundant association rule set for a given database, that can be generated more efficiently than the association rule set. The experimental results also show that this efficiency improvement results from that the generation of the informative rule set needs fewer candidates and database accesses than that of the association rule set rather than large memory usage like some other efficient algorithms. The number of database accesses of the proposed algorithm is significantly fewer than other direct methods for generating association rules on all items.

Further, we presented an adaptive numerical attribute merging algorithm for generating quantitative association rules without consequence specification, which generalise both equal-width and equal-depth discretization methods and is better than both methods as shown in our experiments.

Although the informative rule set provides the same prediction sequence as the association rule set based on the confidence priority, there may exist other definitions of "interestingness" for different applications. How to use the informative rule set to make predictions under different "interestingness" criteria remains a subject of future research.

2.10 Appendix: proof for Lemma 2.4

Before proving Lemma 4, we introduce some terms used in [111]. Given an itemset X, Let mapping $t(X)$ be the set of identifiers of transactions containing the itemset X. Given a tidset Y, let mapping $i(Y)$ be the maximum itemset that is contained in all transactions in Y. Let $c_{it}(X)$ denote the composition of two mappings $i \circ t(X) = i(t(X))$, and $c_{ti}(Y) = t \circ i(X) = t(i(Y))$. Itemset X is closed if $X = c_{it}(X)$. The support of an itemset equals that of its closed itemset.

We first restate the two theorems in paper [111].

Theorem 5 Let R_i stand for a 100% confidence rule $X^i \Rightarrow Y^i$, and let $\mathcal{R} = \{R_1, \ldots, R_n\}$ be a set of rules such that $I_1 = c_{it}(X^i \cup Y^i)$, and $I_2 = c_{it}(Y^i)$ for all rules R_i. Then all the rules are equal to the 100% confidence rule $I_1 \Rightarrow I_2$. Further, all rules other than the most general ones are redundant.

Theorem 6 Let R_i stand for a rule $X^i \Rightarrow Y^i$ with confidence less than 100 %, and let $\mathcal{R} = \{R_1, \ldots, R_n\}$ be a set of rules such that $I_1 = c_{it}(X^i)$, and $I_2 = c_{it}(X^i \cup Y^i)$ for all rules R_i. Then all the rules are equal to rule $I_1 \Rightarrow I_2$. Further, all rules other than the most general ones are redundant.

The lemma needs to prove:

Lemma 2.4 Every rule that is redundant by [111] (Theorem 5 and Theorem 6) is derivable.

Proof For convenience, we omit the upper script of X and Y. We note that if $I = c_{it}(X)$ then both $I \supseteq X$ and $t(I) = t(X)$ hold, which will be used throughout this proof.

Firstly, let us look at Theorem 5. Suppose that $X \Rightarrow Y$ is one of the most general rules in the rule set $\mathcal{R}$. Since $X \Rightarrow Y$ is a 100% confidence rule, we have $t(X) \subseteq t(Y)$. Let $XZ \Rightarrow I_2$ be an equivalent rule of $I_1 \Rightarrow I_2$ and $Z \neq \emptyset$. From the condition given

by Theorem 5, we have $t(XZ) = t(XY) = t(X) \cap t(Y) = t(X)$. Hence we obtain $t(X) \subseteq t(Z)$. As a result, rule $XZ \Rightarrow I_2$ is derivable by Lemma 2.3. Let $I_1 \Rightarrow YZ'$ be another equivalent rule of $I_1 \Rightarrow I_2$ and $Z' \neq \emptyset$. Since $t(YZ') = t(Y)$, we have $t(Y) \subseteq t(Z')$. As a result, $I_1 \Rightarrow YZ'$ is derivable by Lemma 2.3. Hence, we can conclude that all equivalent rules of $I_1 \Rightarrow I_2$ other than the most general ones given in Theorem 5 are derivable.

Next, let us look at Theorem 6. Suppose that $X \Rightarrow Y$ is one of the most general rules in the rule set $\mathcal{R}$. Let $XZ \Rightarrow I_2$ be an equivalent rule of $I_1 \Rightarrow I_2$ and $Z \neq \emptyset$. Since $t(XZ) = t(X)$, we have $t(X) \subseteq t(Z)$. Hence, rule $XZ \Rightarrow I_2$ is derivable by Lemma 2.3. Let $I_1 \Rightarrow XYZ'$ be another equivalent rule of $I_1 \Rightarrow I_2$ and $Z' \neq \emptyset$. From the condition given by Theorem 6, we have $t(XYZ') = t(XY)$. Hence, We obtain $t(XY) \subseteq t(Z')$. As a result, rule $I_1 \Rightarrow XY$ is derivable by Lemma 2.3. Furthermore, $I_1 \Rightarrow XY$ can be derived from $X \Rightarrow XY$, or equivalently from $X \Rightarrow Y$. Hence, we can conclude that all equivalent rules of $I_1 \Rightarrow I_2$ other than the most general ones given in Theorem 6 are derivable. □

Chapter 3

The optimal class association rule set

Association rule mining techniques have been used to generate classification rule sets from relational databases. However, relational databases are usually more dense than transaction databases, so mining on them for the complete class association rule set, which is a set of association rules whose consequences are classes, may be difficult due to combinatorial explosion. Based on the analysis of prediction mechanism, we define the optimal class association rule set to be the set of all potentially predictive class association rules. Using this rule set instead of the complete class association rule set we can avoid redundant computation that would otherwise be required for mining predictive association rules and hence improve the efficiency of the mining process significantly. We present an efficient algorithm for generating the optimal class association rule set using upward closure properties for pruning weak rules before they are actually generated. We analyze the reasons for the efficiency improvement of the proposed algorithm over Apriori on dense databases theoretically, and confirm experimentally that it generates the optimal class association rule set, which is very much smaller than the complete class association rule set, in significantly less time than generating the complete class association rule set.

Partial work in this chapter has been published in Proceedings of the 5th Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining (PAKDD 2001) [60] as a full paper. The subsequent work appeared in journal Knowledge-Based System [62].

3.1 Introduction

3.1.1 Predictive association rules

The goal of association rule mining is to find all rules satisfying some basic requirements, such as the minimum support and the minimum confidence. Association rule mining was initially proposed to solve market basket problems in transaction databases, and has then been extended to solve other problems such as classification. A set of association rules for the purpose of classification is called a predictive association rule set. Usually, predictive association rules are based on relational databases, and the consequences of rules are in a pre-specified column, called the class attribute. Clearly, a relational database can be mapped to a transaction database when an attribute-value pair is considered as an item. After having mapped a relational database into a transaction database, a complete class association rule set a set of association rules with the specified classes as their consequences, and a predictive association rule set is a subset of the complete class association rule set. Generally, predictive association rule mining involves the following two steps.

1. Find all class association rules from a database, and then
2. Prune and organize the found class association rules and return a sequence of predictive rules.

This approach generates the predictive association rule set via the complete class association rule set, and bas the following two problems.

- Since a relational database is usually more dense than a transaction database, it may be difficult to find all class association rules from a relational database due

to the huge number of rules.

- Too many generated class association rules reduce efficiency of subsequently post pruning. This is because the set of generated rules is the input of the post pruning whose efficiency is mainly determined by the number of input rules.

To avoid the above problems, it is therefore desirable to find a small subset of the complete class association rule set that makes predictions as accurately as a classification rule set does, so that this subset can replace the complete class association rule set to avoid the above two problems. Our proposed optimal rule set, which is the set all potentially predictive class association rules, is a such rule set. We further present an efficient algorithm to directly generate the optimal class association rule set by taking advantage of upward closure properties of weak rules that will be discussed in Section 3.4. A clear advantage of the work is that we can obtain predictive association rules via the optimal class association rule set efficiently instead via the complete class association rule set that may not be available because of its expensive computational cost from dense relational databases.

3.1.2 Related work

Association rule mining [2] is a central task of data mining and has shown applications in various areas [18, 8, 66]. Currently most algorithms for mining association rules are based on Apriori [3], and used the so-called "downward closure" property which states that all subsets of a frequent itemset must be frequent. Example of these algorithms can be found in [50, 74, 84]. A symmetric expression of downward closure property is *upward closure* property — all supersets of an infrequent itemset must be infrequent. We will use this term throughout this thesis.

Finding classification rules has been an important research focus in the machine learning community [89, 22]. Classification rules can be viewed as a special form of association rules, since a set of association rules with pre-specified consequences (classes) can be used for classification. Techniques for association rule mining have already been applied to generation of classification rules [8, 66]. Particularly, results in [66] by Bing

Liu et al are very encouraging, because it shows that more accurate classifiers can be built from an association rule set than from the state of the art rule generation system C4.5Rules [89]. However, the algorithm presented in [66] may not be very efficient since it generates classification rules via the complete class association rule set, which may be very large for a relational database.

Traditional association rules are defined by support and confidence, which may not be suitable for a variety of applications. Hence a number of interestingness criteria have been proposed to quantify the interestingness of a rule, such as gain [36], chi-square [18], interest [19](or lift [108]), conviction [19], and etc. For predictions, accuracy is certainly a crucial parameter that must be considered and some discussions on estimation of predictive accuracy appeared in [80, 14]. In addition, the traditional support, called global support, may prevent rules from being generated among small-distributed classes and produce too many rules among large-distributed classes. Both cases hinder the generation of high quality predictive class association rule sets. Some solutions for this problem appeared in [67, 69, 58], including multiple support and local support.

Generally speaking, a class association rule set is a type of target-constraint association rules. Constraint rule sets [12] and interesting rule sets [11] belong to this type. Problems with these rule sets are that they either exclude some useful predictive association rules, or contain many redundant rules that are of no use for prediction. Moreover, algorithms for mining these rule sets handle only one class at a time (building one enumeration tree), so they may not be efficient for generating rules on multiple classes, especially when the number of classes is large.

In this chapter we only address the first step of mining predictive association rules. Related work on pruning and organizing the generated rules can be found in [104, 68, 82].

3.1.3 Contributions

Main contributions of the work in this chapter are listed as follows.

We define the optimal class association rule set to be the set of all potentially

predictive class association rules, which can be used as a substitute for the complete class association rule set for further extracting predictive association rules. Thus, the generation of predictive association rules through the optimal class association rule set approach is more efficient than through the complete class association rule set approach that was used in some previous proposals.

We present an efficient algorithm for generating the optimal class association rule set with respect to all classes. This algorithm achieves significant efficiency improvement over Apriori on dense relational database. We also analyze the reasons for such improvement theoretically.

3.2 The complete class association rule set

Classification rules and association rules differ in the following aspects. Classification rules are generated from a relational database for solving classification problems, whereas association rules are originally generated from a transaction database for market basket problems. Classification rules always have pre-specified consequences (classes) that never appear in the antecedent of a classification rule, whereas association rules usually have no pre-specified consequences and the consequence of an association rule may appear in the antecedent of another rule. The goal of classification rule mining is to obtain a small set of simple and accurate rules, whereas the primary goal of association rule mining is to find all rules satisfying some thresholds. A classification rule mining algorithm usually uses a heuristic criterion and cannot guarantee finding an optimal rule set, whereas an association rule mining algorithm uses a systematic searching method which can produce an optimal rule set.

Despite these differences, a relational database can be mapped to a transaction database when the numerical attributes are discretized. Hence, we can consider a classification rule set as an association rule set with specified consequences.

In classification studies, the basic requirements of a rule are accuracy and coverage. More specifically, a classification rule covers few negative instances in the training database and identifies instances that have not been identified by other rules. A classi-

fication algorithm usually uses implicit minimum accuracy and support although these requirements are not explicit stated. For example, the accuracy of a classification rule is usually high, and those small coverage rules are more likely to be removed in the post pruning. Hence, classification rules also can be defined by an association rule like definition.

Given a relational database D with n attributes, a record of D is a n-tuple. For convenience of description, we consider a record as a set of attribute-value pairs, denoted by T. A *pattern* is a set of attribute-value pairs. The *support* of a pattern P is the ratio of the number of records containing P to the number of records in the database, denoted by $sup(P)$. An *implication* is a formula $P \Rightarrow c$, where P is a pattern and c is a class. The support of the implication $P \Rightarrow c$ is $sup(P \cup c)$. The confidence of the implication is $sup(P \cup c)/sup(P)$, denoted by $conf(P \Rightarrow c)$. The *covered set* of the rule is the set of all records containing the antecedent of the rule, denoted by $cov(P \Rightarrow c)$.

We notice that main goal of classification rule mining is for prediction. Confidence, which is a training accuracy, is not suitable for this purpose. Usually, we obtain accuracy on test data only when we actually conduct experiments on sufficient test data. As a result, we cannot obtain test accuracy in the rule generation stage. However, we need to use this information to forward prune complex rules with low accuracy. Hence, we resort to this theoretical estimation. We use a statistical estimate of accuracy to replace confidence. Using a result from [80], we adopt the lower bound of test (true) accuracy of a hypothesis as the accuracy of the hypothesis. We define the *accuracy* of an implication to be

$$acc(A \Rightarrow c) = conf(A \Rightarrow c) - z_N \sqrt{\frac{conf(A \Rightarrow c)(1 - conf(A \Rightarrow c))}{|cov(A \Rightarrow c)|}}$$

where z_N is a constant related with a statistical confidence interval, for example, $z_N = 1.96$ when the confidence interval is 95%.

In a relational database with a class attribute, records distributed against the classes may be unbalanced, and a (global) support may produce too many rules among large-distributed classes, and too few rules among small-distributed classes. Hence, the support is not suitable for the generation of classification rules. For example, in Hy-

pothyroid database used in our experiment, 95.2 % records belong to class negative and 4.8% to class hypothyroid. So, 5% global support will be too large for class hypothyroid but too small for class negative. Consequently, we define the *local support* of the implication to be $sup(P \cup c)/sup(c)$, denoted by $lsup(P \Rightarrow c)$.

We say $A \Rightarrow c$ is a *class association rule* if $acc(A \Rightarrow c) \geq \kappa$ and $lsup(A \Rightarrow c) \geq \mu$, where κ and μ are user specified minimum accuracy and local support respectively. Usually, the minimum accuracy requirement of a class association rule is high so it is natural to exclude conflicting rules, such as $A \Rightarrow c_1$ and $A \Rightarrow c_2$, in a class association rule set.

Class association rules differ from association rules in the following three aspects.

Firstly, the consequences of class rules are fixed on classes that never appear in the antecedent of any rule while the consequence of an association rule may be any item and may occur in antecedent of another rule.

Secondly, the support is local rather than global, because the significance of a rule is dependent on the proportion of occurrences of its consequence it accounts for. In addition, the global support may prevent us from finding rules for some small distributed classes.

Thirdly, the confidence is replaced by the accuracy. This is because a class rule is used for prediction on unseen instances. The confidence, the accuracy on training data, is not suitable for this purpose. We use a statistical estimate of accuracy to replace confidence. This definition is a result of statistics [80]. The requirement of using this accuracy is that the support of a rule is not too small, for example, the absolute support number is not less than 30. If the support of a rule is too small, we need another estimation of testing accuracy on very small sample data. Laplace accuracy can then be used instead [14]. It is $acc(A \Rightarrow c) = \frac{|cov(Ac)|+1}{|cov(A)|+|C|}$ where $|C|$ is the number of all classes.

Definition 3.1 The *complete class association rule set* is the set of all class association rules wrt a database, the minimum local support and the minimum accuracy.

Given a database D, the minimum local support μ and the minimum accuracy κ,

the complete (class association) rule set[1] is denoted by $R_c(\mu, \kappa)$, or simply R_c.

Usually, the complete rule set is very large because of the density of a relational database. On the other hand, many rules in the complete rule set have no predictive power at all. In the next section, we will consider a subset of the complete rule set which includes all potentially predictive rules.

3.3 The optimal class association rule set

Before we are able to give definition for the optimal rule set, we study how a rule and a rule set make predictions.

In rule set based classification practice, a set of rules is usually sorted by decreasing accuracy, and tailed by a default prediction. This ordered rule set is called a rule based classifier. In classifying an unseen instance (an input record without class attribute information), the first rule that matches the case classifies it. If no rule matches the instance, the default prediction is used. In this chapter, we ignore the effect of the default prediction since we will concentrate on the predictive power of a rule set. In addition, the drawback for using the default class will be discussed in Chapter 4. In the following, we will formalize this procedure.

For a rule r, we use $cond(r)$ to represent its antecedent (conditions), and $cons(r)$ to denote its consequence. Given a test record T, we say rule r cover T if $cond(r) \subseteq T$. A rule can make a prediction on its covered record, denoted by $r(T) \rightarrow cons(r)$. If $cons(r)$ is the class of T, then the rule makes a correct prediction. Otherwise, it makes a wrong prediction. We say the accuracy of a prediction equals the accuracy of the rule making the prediction, denoted by $acc(r(T) \rightarrow c)$. If a rule gives the correct prediction on a record, then we say the rule *identifies* the record (instance).

Given a rule set R and an input T, there may be more than one rule in R that can make the prediction, such as, $r_1(T) \rightarrow c_1, r_2(T) \rightarrow c_2, \ldots$. We say that the prediction made by R is the same as the prediction made by r if r is the rule with the highest

[1] In the rest of this chapter, we consistently discuss class association rules, so we omit words "class association" afterwards.

accuracy of all r_i for $cond(r_i) \subseteq T$. The accuracy of the prediction equals the accuracy of rule r. If there is more than one rule with the same highest accuracy, we choose the one with the highest support among them. If the rules have the same accuracy and support, we choose the one with the shortest antecedent. This is because a simple rule is usually preferred in classification. If a rule set cannot make a prediction for a record, then we say that the rule set gives an arbitrary prediction on the record with the accuracy of zero.

We will consider how to compare the predictive power of rules. We use $r_2 \subset r_1$ to represent $cond(r_2) \subset cond(r_1)$ and $cons(r_2) = cons(r_1)$.

Definition 3.2 Given two rules r_1 and r_2, we say that r_2 is *stronger* than r_1 iff $r_2 \subset r_1 \wedge acc(r_2) \geq acc(r_1)$. We denote rule r_2 is stronger than rule r_1 by $r_2 > r_1$. We say a rule in R is (maximally) *strong* if there is no other rule in R that is stronger than it. Otherwise, the rule is *weak*.

It is clear that only a strong rule can make a prediction in the complete rule set. Thus, we have

Definition 3.3 We call the set of all (maximally) strong rules in the complete rule set as the optimal rule set wrt the complete rule set.

Clearly, the optimal rule set is unique for a complete rule set. Let R_o stand for the optimal rule set.

Finally, we present the most important property of the optimal class association rule set. Suppose we are building a classifier from the complete rule set. It is clear that rules not in the optimal rule set never provide predictions in the classifier built from the complete rule set. We say a rule is a *potentially predictive* rule if it is possibly used to make a precondition in the classifier built from the complete rule set. Then we have:

Theorem 3.1 *The optimal rule set is the set of all potentially predictive rules in the complete class association rule set.*

Proof We will prove this Theorem in two steps. Firstly, all rules in the optimal are potentially predictive rules. Secondly, a rule that is not in the optimal rule set cannot be a potentially predictive rule.

Generally, an input record may be any set of attribute-value pairs, but only a record that is a superset of the antecedent of a rule in the complete rule set can be classified by the classifier built from the complete rule set. If several rules can be used to classify the input record, only strong rules actually make predictions. The rule that eventually makes the prediction must be a strong rule with the highest accuracy among all matched rules as. As an input record may include any patterns, any strong rule may be the rule that makes the prediction. Since the optimal rule set includes all strong rules from the complete rule set, it includes all potentially predictive rules in the complete class association rule set.

Now suppose a rule that is not in the optimal rule set may make prediction on an input record. By the definition there must be a strong rule in the optimal rule set that also may make a prediction. Clearly, the prediction will be made by the strong rule from the above discussion. Hence, a rule that is not in the optimal rule set cannot be a potentially predictive rule.

Consequently, the theorem is proved. □

The optimal rule set already includes all potentially predictive rules. Thus, it is not necessary to keep a rule set that is larger than the optimal rule set for generating predictive association rules.

In the next section, we will present an efficient algorithm to generate the optimal rule set.

3.4 Generating the optimal rule set

3.4.1 The algorithm

A straightforward method to obtain the optimal rule set R_o is to first generate the complete rule set R_c and then prune all weak rules from it. However, generation of

the complete rule set R_c is very expensive and may be impossible when the minimum support is low in a relational database. In this section, we present an efficient algorithm that can find the optimal class association rule set directly without generating R_c first.

Since the optimal rule set is the set of all strong rules, upward closure properties for pruning weak rules are very useful in the design of an efficient optimal rule mining algorithm. Before presenting those upward closure properties, we review a previous definition. We say that rule r_2 is more specific than rule r_1 if $cond(r_1) \subset cond(r_2) \wedge cons(r_1) = cons(r_2)$. Conversely, r_2 is more general than of r_1 if $cond(r_1) \supset cond(r_2) \wedge cons(r_1) = cons(r_2)$. To simplify the presentation, we define $sup(X, c) = sup(X) - sup(Xc)$, and $cov(X, c) = cov(X) \backslash cov(Xc)$

Now we discuss upward closure properties for pruning weak rules.

Lemma 3.1 *If $sup(X, c) = sup(XY, c)$, then $XY \Rightarrow c$ and all more specific rules are weak.*

Proof We rewrite the confidence of rule $A \Rightarrow c$ as $\frac{sup(Ac)}{sup(Ac)+sup(A,c)}$. We know that function $f(u) = \frac{u}{u+v}$ is monotonically increasing with u when v is a constant. Since $sup(X, c) = sup(XY, c)$, using $sup(Xc) \geq sup(XYc)$, we obtain $conf(X \Rightarrow c) \geq conf(XY \Rightarrow c)$. Using relation $|cov(X \Rightarrow c)| \geq |cov(XY \Rightarrow c)|$, we have $acc(X \Rightarrow c) \geq acc(XY \Rightarrow c)$. As a result, $X \Rightarrow c > XY \Rightarrow c$

Since $sup(XZ, c) = sup(XYZ, c)$ for all Z if $sup(X, c) = sup(XY, c)$, we have $XZ \Rightarrow c > XYZ \Rightarrow c$ for all Z.

Consequently, $XY \Rightarrow c$ and all more specific rules are weak. □

We can perceive the lemma as follows: adding a pattern to the conditions of a rule is to make the rule more precise (with less negative examples), and we shall omit the pattern that fails to do so. We then have a look at a special case of the above lemma.

Lemma 3.2 *If $sup(X) = sup(XY)$, then $XY \Rightarrow c$ and all more specific rules are weak for all $c \in C$.*

Proof This can be proved by using relation $sup(X, c) = sup(XY, c)$ for all $c \in C$ if $sup(X) = sup(XY)$. □

Clearly, Lemma 3.1 and Lemma 3.2 are very helpful for searching strong rules, since we can remove a set of weak rules as soon as we find one that satisfies the above Lemmas. Hence, the search space for strong rules is reduced. From proofs, we see that Lemma 3.1 is more general than Lemma 3.2.

We use a level-wise algorithm to generate the optimal class association rule set directly. The basic structure is based on the set enumeration tree as shown in Figure 2.1, called the candidate tree. A node of the candidate tree is labelled by an attribute-value pair and stores two sets $\{A, Z\}$ where $A \cap Z = \emptyset$. A is the set of attribute-value pairs in the path from the root to the node, and is the antecedent of a possible rule. Since A is unique in a candidate tree, we use it as the identity of the node. We call a node as node A if it stores identity set A. The potential target set Z is a set of classes that may be consequences of A, which is initialized by all classes.

Since the identity set is disjoint from the target set, so the candidate generation is very simple. For a pair of sibling nodes $\{A_1, Z_1\}$ and $\{A_2, Z_2\}$, a new node can be generated and then attached as a child of the preceding node as $\{A_1 \cup A_2, Z_1 \cap Z_2\}$.

We first illustrate how to remove infrequent candidates. Please note that in the algorithm, the support is local, so a candidate is frequent with respect to a class. A pattern is infrequent only if it is not frequent with conjunction with all classes. For example, node $\{\{abc\}, \{xy\}\}$, where each letter represents an attribute-value pair, is infrequent only when $lsup(abcx) < \mu$ and $lsup(abcy) < \mu$. If only one holds, for example $lsup(abcx) < \mu$, the node is represented by $\{\{abc\}, \{y\}\}$ which means only one candidate rule $abc \Rightarrow y$ is stored in the node.

We then show how to prune weak rules. If we find out that $sup(ab, y) = sup(abc, y)$, then we may remove y from the potential target set by Lemma 3.1. Consequently, all more specific rules with the consequence of y, for example $abcW \Rightarrow y$ for any W are removed. Further, if we find out $sup(ab) = sup(abc)$, then all class in the potential

target set are removed by Lemma 3.2. As a result, all more specific rules $abcW \Rightarrow y$ and $abcV \Rightarrow x$ for any W and V are permanently removed. In other words, the node is terminated and no more child nodes will be generated from the node. More detailed descriptions are listed as follows.

Algorithm 3.1 Optimal Rule Set Generator

Input: Database D with class attribute C, minimum local support μ and minimum accuracy κ.

Output: The optimal rule set R.

(1) set optimal rule set $R = \emptyset$
(2) count support of 1-patterns
(3) initialize candidate tree T
(4) select strong rules from T and include them in R
(5) generate new candidates as leaves of T
(6) while (new candidate set is non-empty)
(7) count support of the new candidates
(8) prune the new candidate set
(9) select strong rules from T and include them in R
(10) generate new candidates as leaves of T
(11) return rule set R

In the following, we present and explain two key functions in the proposed algorithm.

Function: Candidate Generator

This function generates candidates for strong rules. Let n_i denote a node of the candidate tree, A_i be the pattern (identity set) of node n_i, $Z(A_i)$ be the potential target set of A_i. We use $\mathcal{P}^p(A_k)$ to denote the set of all p-subsets of A_k. Let σ be the (global) minimum support, or $\sigma = min(\mu \times sup(c))$ for $c \in C$. Suppose all patterns are stored

in the lexicographic order, and we use $A_i > A_j$ to indicate that A_i precedes A_j.

(1) for each node n_i at the p-th layer
(2) for each sibling node n_i and n_j $(A_i > A_j)$
(3) generate a new candidate n_k as a son of n_i such that // combining
(4) $A_k = A_i \cup A_j$
(5) $Z(A_k) = Z(A_i) \cap Z(A_j)$
(6) for each $z \in Z(A_k)$ // testing
(7) if $\exists A \in \mathcal{P}^p(A_k)$ such that $sup(A \cup z) \leq \sigma$
(8) then $Z(A_k) = Z(A_k) \setminus z$
(9) if $Z_k = \emptyset$ then remove node n_k

We generate the $(p+1)$-layer candidates from the p layer nodes in the candidate tree. First, we combine a pair of sibling nodes and insert their combination as a new node in the next layer. We initialize the new node by manipulating information from the two nodes. For example the identity set A of a new node is the union of identity sets of the two sibling nodes and the target set of a new node Z is the intersection of two target sets of the two sibling nodes. If any of its p-sub patterns cannot get enough support with any of the possible targets (classes), then we remove the class from the target set. When there is no possible target left, remove the new candidate.

Function: Pruning

This function prunes weak rules and infrequent candidates in the $(p+1)$-th layer of candidate tree. Let T_{p+1} be the $(p+1)$-layer of the candidate tree.

(1) for each $n_i \in T_{p+1}$
(2) for each $A \in \mathcal{P}^p(A_i)$ //A is a p-sub pattern of A_i
(3) if $sup(A) = sup(A_i)$ then remove node n_i //Lemma 3.2
(4) else for each $z_j \in Z(A_i)$
(5) if $lsup(A_i \cup z_j) < \mu$ then $Z(A_i) = Z(A_i) \setminus z_j$

// the minimum support requirement

(6) else if $sup(A, z_j) = sup(A_i, z_j)$ then $Z(A_i) = Z(A_i) \setminus z_j$

// Lemma 3.1

(7) if $Z(A) = \emptyset$ then remove node n_i

We prune a leaf from two aspects, frequent rule requirement and strong rule requirement. Let us consider a candidate n_i in the $(p+1)$-th layer of tree. The removal of weak rules is based on the two lemmas. To examine satisfaction of Lemma 3.2, we test support of pattern A_i stored in the leaf with the support of its sub patterns by Lemma 3.2. There may be many such sub patterns when size of A_i is large. However, we only need to compare its p-sub patterns since upward closure properties of weak rules. As a result, the number of such comparisons is bounded by $p+1$. Once we find that the support of A_i equals to the support of any of its p sub pattern A, we remove the leaf from the candidate tree. So all its super-patterns will not be generated in all deeper layers. In this way, a number of weak rules are removed. The test of satisfaction for Lemma 3.1 is in similar way, but the satisfaction is with respect to a particular target (class). That is, we only remove a target (class) from the potential target set in the leaf. The pruning of those infrequent rule candidates is also with respect to a particular target (class). When the potential target set is empty, the node is removed permanently, and no other more specific rule will be generated afterwards. In our experiments, we will show how effective the weak rule pruning is in dense databases.

3.4.2 Correctness and efficiency

In this subsection, we will first prove the proposed algorithm is correct and then show the algorithm is more efficient than Apriori for generating the optimal class association rule set from a dense database.

First of all, we will prove the correctness of the algorithm.

Lemma 3.3 *Algorithm* 3.1 *generates the optimal rule set correctly.*

Proof This claim can be proved by the following two steps. The candidate generator will generate all possible rule candidates, and the pruned rules are infrequent rules or weak rules.

The base structure of the candidate generation is the set enumeration tree where all patterns can be enumerated at nodes of the tree. Note infrequent rules and weak rules are upward closed, if a node is unqualified for generating a rule occurring in the optimal rule set, so are all nodes in whole branch rooted by the node. In addition, a potential target set is initiated by all classes, and hence no potential class association rule is omitted.

Now, we need to prove that all pruned rules are weak rules. But this follows from Lemma 3.1 and Lemma 3.2.

In summary, the algorithm generates the optimal rule set correctly. □

Now we consider the efficiency of the algorithm.

Let us consider the complexity for finding frequent itemsets by Apriori. To facilitate our discussion, we study at an example.

Example 3.1 We have a set of items {A, B, C, D, E, F }. Suppose that the maximum frequent itemset sets are $F^+ = \{ABC, BE, CE\}$ and the minimum infrequent itemsets are $F^-\{D, AE, BCE\}$. From the Figure 3.1, we can see clearly that Apriori will search all patterns in $\mathcal{P}(F^+) \cup F^-$.

Thus, Toivonen et. al. [72, 40] concluded that the number of searched patterns for Apriori in finding all frequent patterns is $\mathcal{P}(F^+) \cup F^-$ where F^+ is the set of all maximum frequent itemsets and F^- is the set of all minimum infrequent itemsets. As a result, the number of patterns searched for the complete rule set by an Apriori like algorithm is bounded by $|C| \mid \mathcal{P}(F^+) \cup F^- \mid$ where $|C|$ is the number of all classes.

In the following, we will discuss the complexity for generating the optimal class rule set. We define;

Definition 3.4 Pattern A is (a) *qualified* (candidate) if there is a class c such that $lsup(Ac) > \mu$ and $\nexists A' \subset A$ such that $sup(A, c) = sup(A', c)$.

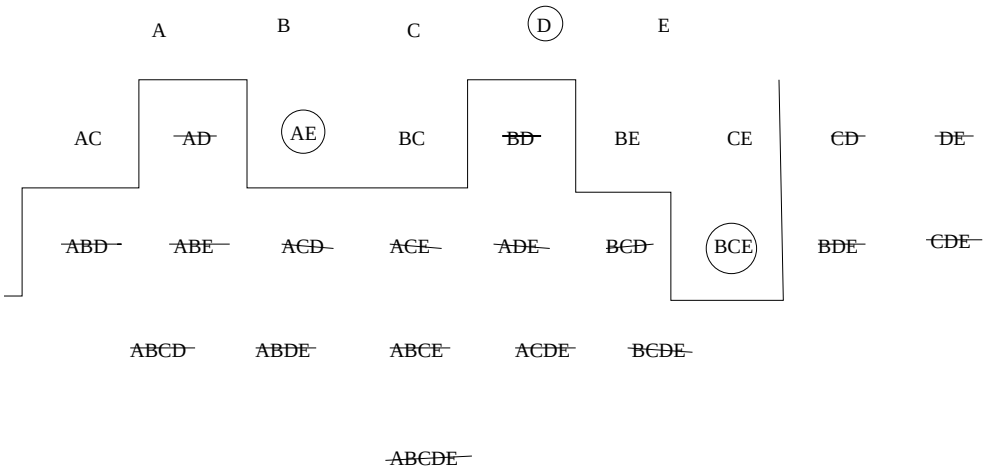


Figure 3.1: Searched and un-searched patterns

It is clear from lemma 3.1 that only rules obtained from qualified patterns can occur in the optimal class rule set.

Let P be the set of all patterns, and Q be the set of all qualified patterns. We define maximum qualified patterns as

$Q^+ = \{p \in Q \mid \nexists p' \in Q \text{ and } p' \supset p\}$

and minimum unqualified patterns as;

$Q^- = \{p \in P \backslash Q \mid \nexists p' \in P \backslash Q \text{ and } p' \subset p\}$

We have an immediate Theorem,

Theorem 3.2 *The number of searched patterns for generating the optimal class association rule set by Algorithm 3.1 is bounded by* $|C| \mid \mathcal{P}(Q^+) \cup Q^-|$.

Proof A pattern in Q^- is unqualified, and based on upward closure properties of infrequent pattern and weak rules, we need not search any super-patterns of a pattern in Q^-. On the contrary, we have to search all patterns in $\mathcal{P}(Q^+)$.

For each pattern, there are at most $|C|$ rule candidates.

Hence, the theorem is proved. □

It is clearly that $|C||\mathcal{P}(F^+) \cup F^-| \geq |C||\mathcal{P}(Q^+) \cup Q^-|$ if frequent pattern is decided by local support. Hence, Algorithm 3.1 will be more efficient than Apriori in the optimal class association rule set generation.

We will further show that the efficiency improvement will be more significant when a database is dense. Let us look at an example;

Example 3.2 Consider a dense data set in which all records have the same set of attribute-value pairs with the number of m and a class, then the maximal length of patterns in F^+ and F^- are m and 0, but the maximal length of patterns in Q^+ and Q^- are 1 and 2. Apriori will search 2^m candidates while our proposed algorithm only search up to m^2 candidates.

Consequently, the proposed algorithm for generating the optimal rule set will be more efficient than Apriori followed by post-pruning, and this will be confirmed by experimental results.

Clearly, pattern A is qualified if there is a class c such that $lsup(Ac) > \mu$ and $\nexists A' \subset A$ such that $cov(A, c) \subseteq cov(A', c)$. When the length of a pattern becomes large, its covering set usually shrinks, and hence is more likely to be a subset of the covering set of another pattern. Consequently, long rules are more likely to be excluded from the optimal rule set than short rules. This will also be confirmed by the experimental results.

3.5 Experimental results

We have implemented the proposed algorithms and evaluated them on 6 real world databases from UCI ML Repository [15], which are detailed in Appendix A.

In our experiments, we set the minimum accuracy as 0.5 and the minimum local support of 0.1 for both the complete and the optimal rule sets. We generate the complete rule set by Apriori with the same data structure as the proposed algorithm. Note that when the optimal rule set is generated by Apriori, additional pruning cost are needed to remove weak rules from the complete rule set. So the cost for generating the optimal rule set by Apriori is more expensive than that for generating the complete rule set. As a result, it will be evident enough to show efficiency improvement by comparing the efficiency for generating the optimal rule set with that for generating the complete

rule set. To present competitive results, we restrict the maximum layer of candidate trees to 4 since rule length constraint is an effective way to avoid combinatorial explosion and has been used in practice. For example, [66] restricted the maximum size of the found rule sets. Justification for such constraint is that long rules usually have very limited predictive power in practice. In fact, the proposed algorithm performs more efficiently than Apriori when there is no such restriction, and this is clear from the second part of our experiments.

Comparisons of rule set size and generation time between a complete rule set and an optimal rule set are listed in Figure 3.2. It is easy to see that the size of an optimal rule set is on average only 6% the size of the corresponding complete rule set. Considering that the optimal rule set includes all potentially predictive rules from the complete rule set, this rule set size reduction is very impressive. Similarly, the time for generating rules is significantly shorter as well. We have obtained reduction of 75% in the generation time on average. Moreover, by using a smaller optimal rule set instead of a lager complete rule set as an input for the post pruning, we will gain more efficiency improvement in the second stage of predictive association rule generation.

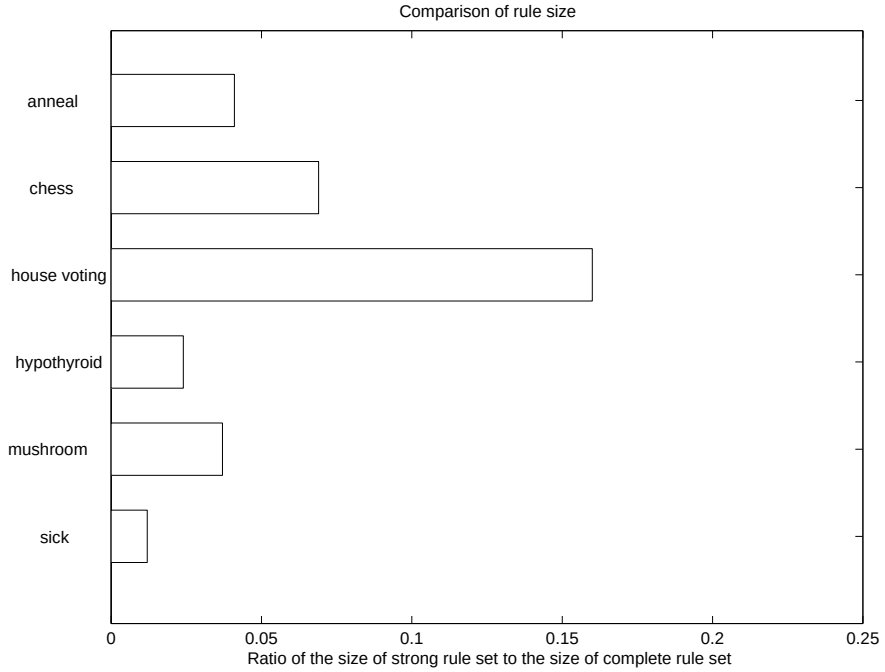
To understand reasons for such significant efficiency improvement and to confirm the analysis presented in the previous section, we illustrate the number of nodes in each layer of the candidate trees of two databases in Figure 3.3. In this experiment, we lift the restriction of maximum number of layers. We can see that the number of nodes expands at a sharp exponential rate for generating the complete rule set. In contrast, the number of nodes increases slowly for generating the optimal rule set, reaches a low maximum quickly, and then decreases gradually. When a tree for generating the optimal rule set stops growing, its corresponding tree for generating the complete rule set just passes its maximum. In deep tree levels, after 4 in our case, the nodes being searched for the generation of optimal rule sets are only 1% of that for complete rule sets. This shows how much redundancy we have eliminated in the generation of optimal rule sets. In our experiment, more than 95% of the time is used for such redundant computation when there is no maximum layer restriction. Besides, from this detailed

illustration of candidate tree growing without length restriction, we can understand that the proposed algorithm will perform even more efficiently when there is no maximum layer number restriction in comparison with Apriori for generating the complete sets. This also confirms our analysis that long rules are more likely to be excluded from the optimal rule set.

3.6 Conclusion

In this chapter, we studied the problem of efficiently mining predictive association rules from dense relational databases. We defined the optimal class association rule set, which is the set of all potentially predictive class association rules from the complete class association rule set. Hence it can be used as a substitute of the complete class association rule set for finding predictive association rules. We presented an efficient algorithm to generate the optimal class association rule set. Our algorithm avoids much redundant computation required in generating the complete class association rule set, and hence improves efficiency of the mining predictive association rules significantly. We analysed the efficiency of the proposed algorithm theoretically and confirmed the analysis by testing the algorithm on some real world databases. Our experimental results show that the optimal class association rule set has a significantly smaller size and requires significantly less time to generate in comparison with the complete class association rule set generated by Apriori. Our proposed algorithm is very efficient in dense databases because it produces significantly less candidates, especially in deep tree levels.

Mining the optimal class association rule set is the first step for the generation of predictive association rules, and post-pruning and organization of the found optimal class association rule set is the second step, which needs further study.



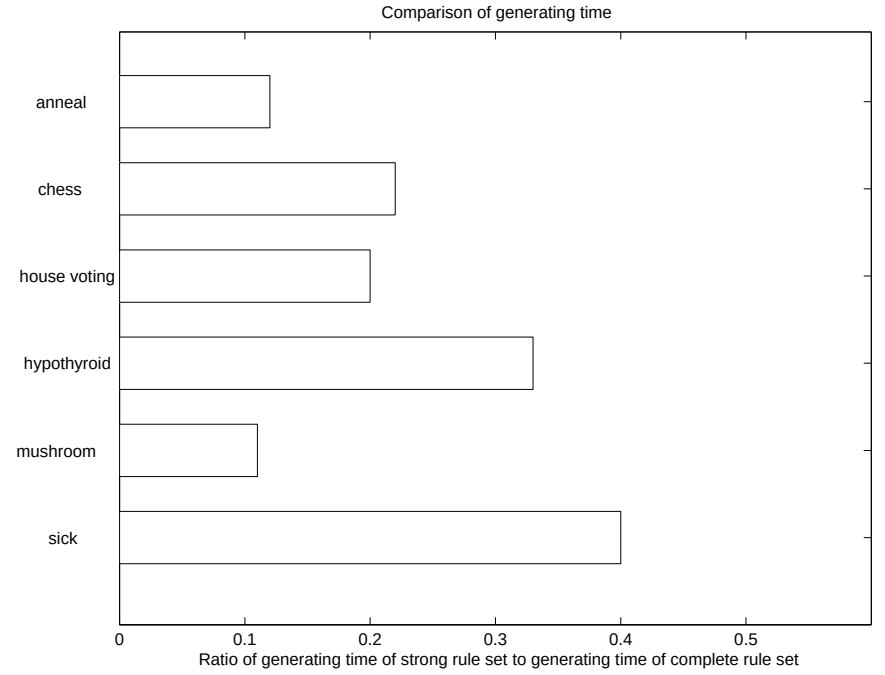


Figure 3.2: The comparison of size and generation time for optimal rule set R_o and complete class association rule set R_c (in the ratio of R_o to R_c)

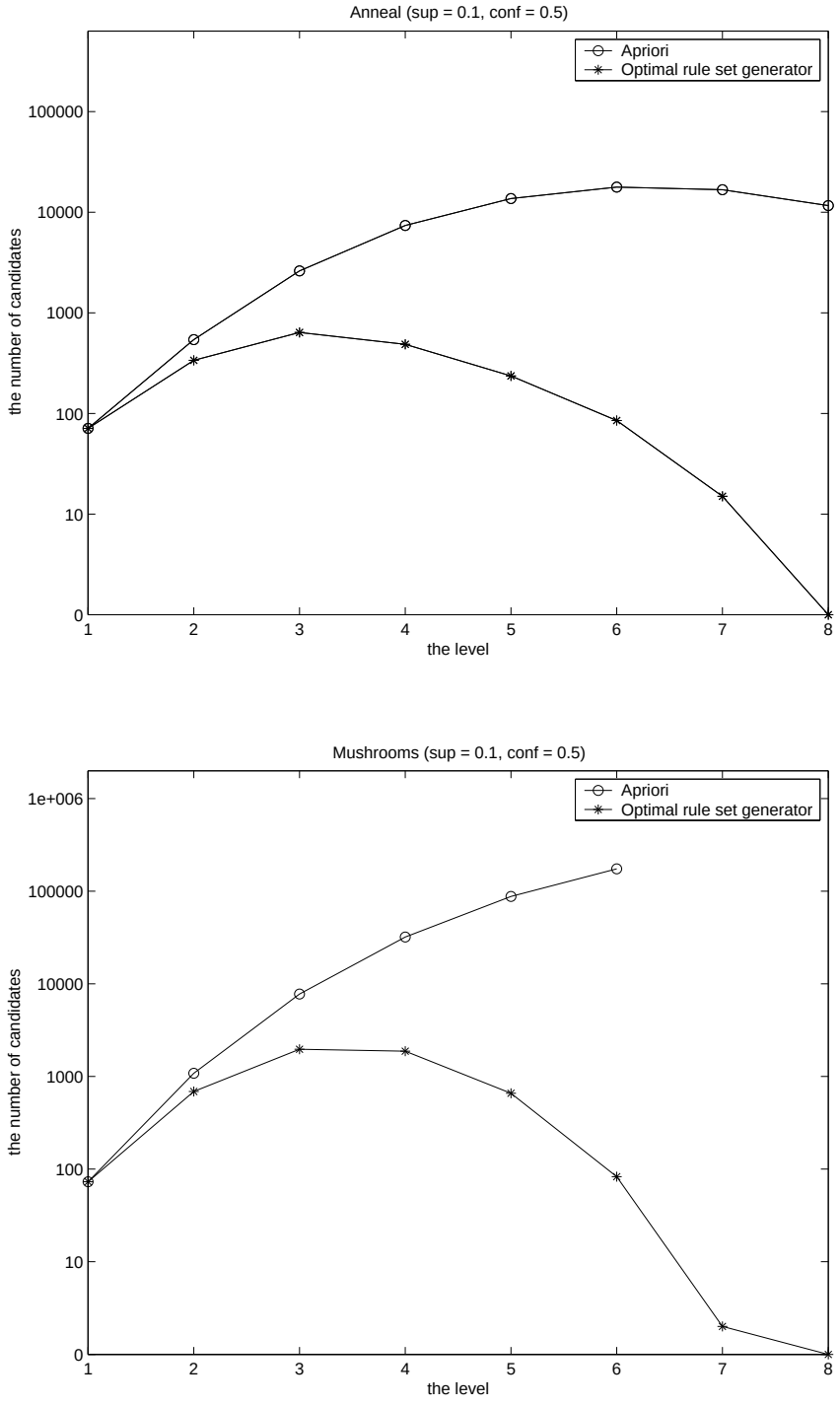


Figure 3.3: The comparison of candidate number for generating the optimal class association rule set and the complete class association rule set

Chapter 4

Robust classification rule sets

We study the problem of computing classification rule sets from relational databases so that accurate predictions can be made on test data with missing attribute values. Traditional classifiers perform badly when test data are not as complete as the training data because they tailor a training database too much. We introduce the concept of one rule set being more robust than another, that is, able to make more accurate predictions on test data with missing attribute values. We show that the optimal class association rule set is as robust as the complete class association rule set. We then introduce the *k-optimal* rule set, which provides predictions exactly the same as the optimal class association rule set on test data with up to k missing attribute values. This leads to a hierarchy of k-optimal rule sets in which decreasing size corresponds to decreasing robustness, and they all more robust than a traditional classification rule set. We present two methods to find k-optimal rule sets, an optimal association rule mining approach and a heuristic approximate approach. We show experimentally that a k-optimal rule set generated by the optimal association rule mining approach performs better than that by the heuristic approximate approach and both rule sets perform significantly better than a typical classification rule set (C4.5Rules) on incomplete test data.

Partial work in this chapter has been published in Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD02)

as a poster paper [63].

4.1 Introduction

4.1.1 Motivation

Automatic classification has been a goal for machine learning and data mining, and rule based methods are widely accepted due to their understandability and explanatory. Rule based classification usually involves two stages, learning and testing. Consider a relational database where each record is assigned a category (class), called a training database. In the learning stage, we generate a rule set where each rule associates a pattern with a class. Then in the test stage, we apply this rule set to test data without class information, and to predict the class that a record in the test database belongs to. If the predictive class is the class that the record supposed to belong to, then the prediction is correct. Otherwise, it is a wrong prediction. The proportional of correct predictions from test data is accuracy and surely a high accuracy is preferred.

In the machine learning community, many classification rule systems have been proposed, and they produce satisfactory accuracy in many applications. However, when the test data is not as complete as the training data, a classification rule set may perform poorly because it tailors the training data too much. We will give the following example to show this.

Example 4.1 Given a well-known data set listed in Table 4.1, a decision tree (e.g. ID3 [89]) can be constructed as in Figure 4.1.

The following 5 rules are from the decision tree.

1. If outlook is sunny and humidity is high, then do not play tennis.
2. If outlook is sunny and humidity is normal, then play tennis.
3. If outlook is overcast, then play tennis.
4. If outlook is rain and wind is strong, then do not play tennis.
5. If outlook is rain and wind is weak, then play tennis.

We note that all rules include the attribute outlook. Suppose that we have a test

index	Outlook	Temperature	Humidity	Wind	PlayTennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Table 4.1: A training data set

data set in which outlook information is unknown. Then these rules cannot make any predictions. Hence, this rule set is not robust at all. However, we may have another rule set that can make some predictions in the presence of missing i.e. outlook information in test data.

In real world applications, missing data in a database is very common, especially in a test database. For example, we may generate a diagnostic rule set from records with the complete check results. But when we apply the rule set, some records may miss one or more check results for some reasons. Hence, a rule set that can make reasonably accurate predictions in the presence of missing attribute values in test data is highly desirable in practice. We say that a rule set is more robust than another rule set if it can make more accurate predictions on incomplete test data than the other rule set. In the following, we will explore this problem and its solutions.

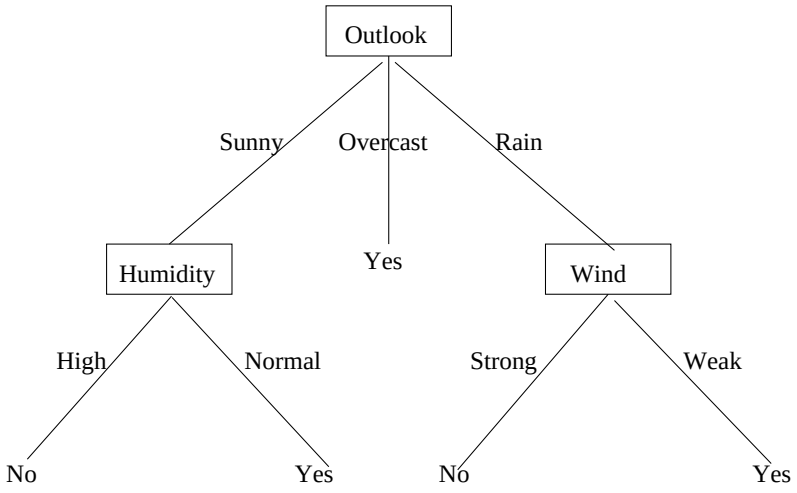


Figure 4.1: A decision tree from the training data set

4.1.2 Related work

Classification rule mining algorithms have been mainly developed for category prediction by the machine learning community. They use heuristic methods to find simple rule sets to explain training data well, and are generally categorized into two groups [80], simultaneously covering algorithms namely C4.5 [89], and sequential covering algorithms such as AQ15 [76] and CN2 [22]. Most algorithms generate simple and accurate rule sets that cover all training data, but these rule sets may not be robust in the presence of missing values as we will discuss in this chapter.

There are many proposals for improving predictive accuracy of traditional classifiers, among which Bagging [16] and Boosting [33, 34] are significant in reducing predictive errors. Both techniques utilize voting (weights are involved in Boosting) from a set of classifiers obtained by sampling the training database. However, Bagging and Boosting make predictions hard to understand by users. In this chapter, we also consider multiple rule sets, but we disturb a database systematically and use the union of all rule sets.

In the last few years, association rule mining has been extensively studied. Some methods have been proposed to generate classification rule sets by association rule mining approach, and more details can be referred to Chapter 3.

There is some previous work on handling missing attribute values in training data

[79, 89], but to the best of our knowledge there is no report on how to deal with the missing attribute values on test data.

4.1.3 Contributions

Main contributions of the work in this chapter are listed as follows.

We investigate a new problem of mining robust classification rule sets, which are used to make predictions on test database that is not as complete as the training database. We define a criterion to compare robustness among rule sets from the training database. We study the robustness of a number of rule sets from a traditional classification rule set to the optimal class association rule set.

We present an algorithm to generate a robust rule set that performs better than that from an extension of C4.5Rules on test data with missing attribute values, and show both rule sets perform significantly better than a rule set from C4.5Rules on test data with missing attribute values.

4.2 Robustness of the optimal class association rule set

We follow definitions in the previous chapter. Consider a relational database with class information, called a *training database*. A pattern is a set of attribute-value pairs and a value in the class attribute is a class. A rule is an implication between a pattern and a class whose local support and accuracy are at least the specified minimum local support and confidence respectively. The set of all rules from a database is the complete (class association) rule set[1], denoted by R_c. Let a database without class information be a *test database*. Rule r may make prediction on record T if r covers T, i.e. $cond(r) \subseteq T$. The potential prediction is $cons(r)$ and the predictive accuracy is $acc(r)$. For a record in the test database, we say a rule is a predictive rule if there is no other rule in the complete rule set that can provide prediction with higher accuracy. More formally,

[1] In the rest of this chapter, we shorten the complete class association rule set to the complete rule set, and the optimal class association rule set to the optimal rule set.

Definition 4.1 Let T be a record in database D and R a rule set for D. A rule r in R is *predictive for T wrt R* if r covers T. If two rules cover T we choose the one with the greater accuracy. If two rules have the same accuracy we choose the one with higher support. If two rules have the same support we choose the one with the shorter condition.

Please note that in the above definition, we take the support and the number of conditions of a rule into consideration. This is because they have been minor criteria for sorting rules in a rule based classifier in previous practice, such as in [66]. It is easy to understand the preference of the highest support rule among a number of rules with the same accuracy. The preference for a short rule is consistent with the preference for a simple rule in traditional classification rule generation practice.

As both accuracy and support are real numbers, in a large database it is very unlikely that a record supports two rules with the same accuracy and support. Therefore, we suppose that each record has a unique predictive rule for a given database and rule set in the rest of chapter.

Suppose that we have an incomplete test record, i.e. some attribute information is missing. It is clear that we prefer a rule set that can make correct prediction on this incomplete record. More formally, we will give a definition for the robustness as following. Note that we say that a rule set gives any prediction on a record with accuracy of zero when it cannot provide a prediction on the record.

Definition 4.2 Let D be a database and R_1 and R_2 be two rule sets for D. Rule set R_1 is *more robust* than R_2 if, for all $T' \subseteq T$ and $T \in D$, predictions made by R_1 are at least as accurate as those by R_2.

Suppose that R_1 is more robust than R_2. For test data that are as complete as the training data both rule sets give the same number of correct predictions with the same accuracy. For test data that are not as complete as the training data, rule set R_1 can provide at least the same number of correct predictions as rule set R_2 with at least the same accuracy. Hence, a robust rule set has more predictive power when test data are

not as complete as the training data.

Naturally, more rules will enhance the robustness of a rule set, and the complete rule set is the most robust rule set. However, this rule set is usually too large and includes many rules without prediction power. Hence, we can go further to simplify it.

Clearly there are natural connections between strong rules and predictive rules since a strong rule is a potentially predictive rule. Hence, we have,

Theorem 4.1 *For every rule set $R \subseteq R_c$ for database D, the optimal class association rule set R_o is the smallest rule set that is as robust as the complete class association rule set.*

Proof Since the optimal rule set includes all potentially predictive rules by theorem 3.1, the optimal rule set is as robust as the complete rule set.

Now we will prove the minimum property. Suppose that we may omit a rule r from the optimal rule set R_o and the new rule set $R'_o = R_o \setminus r$ is still as robust as the complete rule set R_c. Consider a test record that is matched only by rule r. By the definition of the optimal rule set, there may not be a rule in R'_o matching the test record or at most there is a matching rule with a lower accuracy than r. Hence, the prediction made from R'_o cannot be as accurate as that from R_o. As a result, R'_o cannot be as robust as R_c.

The theorem is proved. □

This means that no matter what an input record is (complete or incomplete), that the optimal rule set gives exact the same prediction on the record at the same accuracy as the complete rule set. The optimal rule set is the minimum most robust rule set.

Let us look at differences between the complete rule set and the optimal rule set through an example.

Example 4.2 From the database in the introduction, there are 21 rules in the complete rule set when the minimum support is 2/14, and the minimum confidence is 80% (we use confidence here for convenience). However, there are only 10 rules in the optimal rule

set which are listed as follows. (Numbers in parentheses are support and confidence.)

1. If outlook is sunny and humidity is high, then do not play tennis. $(3/14, 100\%)$
2. If outlook is sunny and humidity is normal, then play tennis. $(2/14, 100\%)$
3. If outlook is overcast, then play tennis. $(4/14, 100\%)$
4. If outlook is rain and wind is strong, then do not play tennis. $(2/14, 100\%)$
5. If outlook is rain and wind is weak, then play tennis. $(3/14, 100\%)$
6. If humidity is normal and wind is weak, then play tennis. $(3/14, 100\%)$
7. If temperature is cool and wind is weak, then play tennis. $(2/14, 100\%)$
8. If temperature is mild and humidity is normal, then play tennis. $(2/14, 100\%)$
9. If outlook is sunny and temperature is hot, then do not play tennis. $(2/14, 100\%)$
10. If humidity is normal, then play tennis. $(6/14, 87\%)$.

Since the complete rule set is larger, we do not show it here. However, to demonstrate why some rules in the complete rule set are unnecessary in prediction, we list 7 rules including attribute value overcast as follows:

3. if outlook is overcast then play tennis. (4/14, 100%)
11. if outlook is overcast and temperature is hot, then play tennis. (2/14, 100%)
12. if outlook is overcast and humidity is high then play tennis. (2/14, 100%)
13. if outlook is overcast and humidity is normal then play tennis. (2/14, 100%)
14. if outlook is overcast and wind is strong, then play tennis. (2/14, 100%)
15. if outlook is overcast and wind is weak, then play tennis. (2/14, 100%)
16. if outlook is overcast and temperature is hot and wind is weak, then play tennis. (2/14, 100%)

However, there is only rule 3 included in the optimal rule set from the above 7 rules. Clearly, the other 6 rules cannot be predictive rule in any cases since they have lower support (confidence is the same) and are more specific than rule 3.

In the above example, the underlying database is very small, so the size difference between the complete rule set and the optimal rule set is not very significant. As we showed in the previous chapter, the optimal rule set can be less than 1% of the complete rule set.

Even though the optimal rule set is much smaller than the complete rule set, it is still much larger than a traditional classification rule set. Some rules in the optimal rule set may be unnecessary when the number of missing attribute values is limited. Hence, we may further simplify the optimal rule set. Besides, we are interested in the relationships between the optimal rule set and a traditional classification rule set. These are goals in the next section.

4.3 Robustness of k-optimal class association rule sets

In this section, we have a default rule set, namely the complete rule set, from the training database. When we say a predictive rule without mentioning a rule set, then it is with respect to the complete rule set. The test database is the same as the training database without class information.

Robustness mainly concerns missing attribute values in test databases, and hence we first define a k-incomplete database to be a new database with exactly k missing values from every record of the test database.

Definition 4.3 Let D be the test database and $k \geq 0$. The k-incomplete database $D_k = \{ T' \mid T' \subset T, T \in D, |T| - |T'| = k\}$.

For convenience of discussion, we consider all k-incomplete databases of D as a set of $\binom{n}{k}$ (n is the number of attributes for D) databases in which each omit exactly k attribute (column) information from D. For example, all 1-incomplete databases contains a set of n databases where each omits one attribute (column) information from D. We note that the 0-incomplete database of D is D itself.

Let us represent the optimal rule set in terms of incomplete databases. The following lemma is a variation of Theorem 3.1 in the previous chapter.

Lemma 4.1 *R_o is the set of predictive rules for records in k-incomplete databases wrt R_c where $0 \leq k \leq n$.*

Proof From Theorem 3.1 we know that all rules in the optimal rule set are potentially

predictive rules, and here we will further prove that all rules in the optimal rule set will be predictive rules in some cases.

Let a test record in a k-incomplete database contain only the antecedent of rule r. Obviously only r can be the predictive rule of the test record because other more general rules in R_o have lower accuracy than r has. Hence, all rules in the optimal rule set will be predictive rules for some incomplete test records.

Consequently, the lemma is proved. □

The optimal rule set preserves all potentially predictive rules from a training database for all incomplete databases. Now we consider how to preserve all potentially predictive rules for some incomplete test databases.

Definition 4.4 The *k-optimal* rule set ($k \geq 0$) over a database is the set of all predictive rules on all k-incomplete databases.

We then have the following result.

Lemma 4.2 *The k-optimal rule set provides the same predictions as the optimal rule set on all p-incomplete databases for* $0 \leq p \leq k$.

Proof It is clear that the k-optimal rule set contains predictive rules for all p-incomplete databases with $0 \leq p \leq k$ from the definition of the k-optimal rule set, and hence this lemma is holds immediately. □

We can understand a k-optimal rule set in the following way. A k-optimal rule set is a subset of the optimal rule set that makes prediction as well as the optimal rule set on a test database with k missing attribute value per record. As a special case, 0-optimal rule set makes predictions as well as the optimal rule set on a complete test database.

Theorem 4.2 *The* $(k+1)$*-optimal rule set* ($k \geq 0$) *is at least as robust as the k-optimal rule set.*

Proof For those records in all p-incomplete databases for $p \leq k$, both rule sets must give the same predictions because both provide predictions as accurate as the optimal rule set.

For those records in a $(k+1)$-incomplete database, the $(k+1)$-optimal rule set provides predictions as accurate as the optimal rule set and the k-optimal rule set does not. Hence, there may be some records that are identified by the $(k+1)$-optimal rule set but not by k-optimal rule set, or are identified with lower accuracy.

Consequently, a $(k+1)$-optimal rule set ($k \geq 0$) is at least as robust as a k-optimal rule set. □

Clearly, a k-optimal rule set is a subset of the optimal rule set.

We give an example to show k-optimal rule sets and their predictive capabilities.

Example 4.3 In the database of playing tennis, with the minimum support of 2/14 and the minimum confidence of 80% (for convenience, we use confidence here), we have the optimal rule set with 10 rules as shown in Example 4.2. These 10 rules can identify all records in the data set. We have the 0-optimal rule set as follows, where two numbers in the parentheses are support and confidence respectively.

1. If outlook is sunny and humidity is high, then do not play tennis. $(3/14, 100\%)$
2. If outlook is sunny and humidity is normal, then play tennis. $(2/14, 100\%)$
3. If outlook is overcast, then play tennis. $(4/14, 100\%)$
4. If outlook is rain and wind is strong, then do not play tennis. $(2/14, 100\%)$
5. If outlook is rain and wind is weak, then play tennis. $(3/14, 100\%)$
6. If humidity is normal and wind is weak, then play tennis. $(3/14, 100\%)$

Rules 1, 2, 3, 4 and 5 identify different instances in the database, so they all must be included in the 0-optimal rule set. As to rule 6, it is the predictive rule of record 9. When identifying record 9, rule 6 has higher support than the rule 2 has and hence be included. This consideration results that the 0-optimal rule set is more robust than the rule set from the decision tree. When the outlook information is missing, the rule set from decision tree identifies nothing while the 0-optimal rule set still identifies 3

instances. Clearly, the 0-optimal rule set provides exactly the same predictions as the optimal rule set on the complete test database.

With the following 4 additional rules, the rule set becomes 1-optimal.

7. If temperature is cool and wind is weak, then play tennis. $(2/14, 100\%)$

8. If temperature is mild and humidity is normal, then play tennis. $(2/14, 100\%)$

9. If outlook is sunny and temperature is hot, then do not play tennis. $(2/14, 100\%)$

10. If humidity is normal, then play tennis $(6/14, 87\%)$.

This rule set can give more correct predictions on incomplete test data than the 0-optimal rule set. For example, when outlook information is missing, the 1-complete rule set identifies 6 records; which is 3 more than the 0-complete rule set does; when temperature, 14, equal; when humidity, 11, 2 more; and when wind, 11, 2 more. The improvement is clear and positive. In this case, 1-optimal rule set equals the optimal rule set, but in general, it only provides exactly the same predictions as the optimal rule set on all 1-incomplete test databases.

Since the example database has only four attributes, the 1-optimal rule set is in the same size as the optimal rule set. However, in most real world databases where the number of attributes is large, the 1-optimal rule set is usually much smaller than the optimal rule set as shown in our experiments.

The k-optimal rule sets form a hierarchy.

Lemma 4.3 *Let R^k and $R^{(k+1)}$ be the k-optimal and the $(k+1)$-optimal rule sets for D and R_c. Then $R^k \subseteq R^{k+1}$.*

Proof R^k contains the set of all predictive rules over all p-incomplete databases of the training database for $p \leq k$. $R^{(k+1)}$ contains the set of all predictive rules over all p-incomplete databases of the training database for $p \leq k$ and all predictive rules over $(k+1)$-incomplete databases as well. The predictive rule for a record is unique as supposed, so, $R^k \subseteq R^{k+1}$. □

Till now, we have introduced the set of optimal rule sets, and we observe that the

following chain always holds these optimal rule sets.

$$R_c \supseteq R_o \supseteq \cdots \supseteq R^{k+1} \supseteq R^k \supseteq \cdots \supseteq R^0$$

From this relation, we can see that the robustness of a k-optimal rule set for $k \geq 0$ is due to that it preserves more potentially predictive rules in case that some rules are paralysed by missing values in a test database.

Usually, a traditional classification rule set is smaller than a 0-complete rule set, since most post pruning algorithms of traditional classification systems work in a way to reduce the size of an output rule set. Because of the heuristic trait of traditional classification rule generation algorithms, we cannot characterize the exact relationship between a traditional classification rule set and a k-optimal rule set. From our observations, most rules in a traditional classification rule set are in the 0-optimal rule set. For example, the rule set from the decision tree on the tennis database is a subset of 0-optimal rule set. Generally, a traditional classification rule set is less robust than a 0-optimal rule set.

Finally, we will consider a property that will help us to find k-optimal rule sets. We can interpret the k-optimal rule set through a set of 0-optimal rule sets.

Lemma 4.4 *The union of all 0-optimal rule sets over all k-incomplete databases is the k-optimal rule set.*

Proof A 0-optimal rule set contains all predictive rules on the training data. Using each of every k-incomplete databases as a training database, we obtain a 0-optimal rule set. Clearly, the generated 0-optimal rule set is as robust as the complete rule set on the k-incomplete database, and the union of the all these 0-optimal rule sets is as robust as the complete rules set on all k-incomplete databases. Hence, the union of all 0-optimal rule sets over all k-incomplete databases is k-optimal. □

This lemma suggests that we can generate a k-optimal rule set by generating 0-optimal rule sets on a set of incomplete databases of the training database.

4.4 Generating k-optimal rule sets

We now consider two different methods for constructing robust rule sets from a given database. The first method extends the traditional classification rule generation technique and the second extends the optimal class association rule mining technique. Both methods aim at generating a k-optimal rule set, for arbitrary $k \geq 0$.

4.4.1 A multiple decision tree approach

Heuristic methods have been playing an important role in classification problems, so here we first discuss how to generate k-optimal rule sets by a heuristic method.

Given a set of k-incomplete databases of training data, we can construct a set of rule sets where each corresponding with a incomplete database. Intuitively, the union will withstand up to k missing values to some extent.

We use C4.5Rules as the base rule generator in this algorithm. Although constructing multiple classification rule sets has been discussed before, this is the first proposal to systematically disturb the training data and use the union of all rule sets.

Algorithm 4.1 Multiple tree algorithm

Input: Database D, integer $k \geq 1$

Output: Rule set R

(1) set $R = \emptyset$
(2) for each k-attribute set X
(3) form a new database D' by projecting D to the remaining attributes
(4) build a decision tree from D' by C4.5
(5) call c4.5Rules to generate a rule set R'
(6) let $R = R \cup R'$
(7) return R

We now study the robustness of the output rule set of the algorithm.

Clearly, the output rule set is at least as robust as each component.

Suppose that each R' is the 0-optimal rule set for the corresponding k-incomplete database. Then the output rule set must be k-optimal rule set by Lemma 4.4. We know that a traditional classification rule set is less robust than the 0-optimal rule set as indicted in the previous section. Therefore, the output rule set is at most as robust as the k-optimal rule set. This will be confirmed by our experiment.

Now, we consider efficiency of the algorithm. This algorithm may be expensive when k is large. This is because $\binom{n}{k}$ rule sets have to be generated where much redundant computation is involved. Thus, we consider another method.

4.4.2 An optimal class association rule set approach

In this section, we present a "precise" method to compute k-optimal rule set from a given database. A naive method would perform the following three steps:

1. generate the complete rule set,
2. find a 0-optimal rule set for every k-incomplete database, and
3. form the union of all 0-optimal rule sets.

This method would be inefficient. Firstly, the complete rule set is usually very large, and is too expensive to compute for some databases. Secondly, the process of finding a 0-optimal rule set from the large complete rule set is expensive too.

In our proposed algorithm, we will find the smaller optimal rule set directly and compute the k-optimal rule set from the optimal class rule set in a single pass over the database.

An efficient algorithm for generating the optimal rule set is presented in the previous Chapter. Here, we only present an algorithm to compute the k-optimal rule set from the optimal rule set.

Given a rule r, let $Attr(r)$ be the set of attributes whose values appear in the antecedent of r. A p-attribute pattern is an attribute set containing p attributes. Given a record T and an attribute set X, let $Omit(T, X)$ be a new partial record projected from T without attribute values from X.

Algorithm 4.2 k-optimal rule set generator

Input: Database D, optimal rule set R_o and integer $k \geq 0$

Output: k-optimal rule set R

(1) set $R = \emptyset$
(2) for each record T_i in D
(3) set $R_i = \emptyset$ and let R'_i include all rules covering T_i
(4) for each k-attribute set X let $T = Omit(T_i, X)$
(5) if there is no predictive rule for T in R_i
(6) then selected a predictive rule r' for T and move it from R'_i into R_i
(7) let $R = R \bigcup R_i$
(8) return R

The correctness of mining the optimal rule set is proved in the previous chapter. Here, we only prove that the above algorithm produces the k-optimal rule set correctly given the input of the optimal rule set.

Lemma 4.5 *Algorithm* 4.2 *produces a k-optimal rule set correctly from the optimal class association rule set.*

Proof This algorithm selects predictive rules for all k-missing patterns on each of every record in the training database in line (5) to (7). From Lemma 4.4 in the previous section, the algorithm selects the k-optimal rule set correctly. □

4.5 Experimental results

In this section, we will compare experimentally sizes and generation times of different rule sets and their performances on incomplete test databases. These results in turn materialise concepts we presented in Section 4.3.

We use four databases from UCI ML Repository [15] in our experiments and a brief summary of the databases is listed in Appendix A. Our experiments were conducted on a Sun server with two 200 MHz UltraSPARC CPUs.

The experimental settings is listed in Table 4.2. Min Sup is the minimum local support, Min Acc is the minimum accuracy and Max Length is the length of antecedent of the longest rule in an optimal rule set. To save time, in database Hypothyroid we stopped executing the program before it found the complete optimal rule set.

Database	Min Sup	Min Acc	Max Length
Anneal	0.05	0.95	7 (unconstrained)
Congressional Voting	0.1	0.95	9 (unconstrained)
Hypothyroid	0.1	0.95	4 (constrained)
Mushrooms	0.2	0.95	6 (unconstrained)

Table 4.2: The experimental setting

Sizes and generation time of different rule sets are listed in Table 4.3. The size of a k-optimal rule set is much smaller than that of the optimal rule set and is a little larger than that of a traditional classification rule set. There is no indication which of the optimal rule set approach and the multiple C4.5 rules approach is more efficient when $k \leq 1$, because the generation time of multiple C4.5Rules is longer than those for 0-optimal rule sets and 1-optimal rule sets in databases Mushrooms and Anneal but is shorter in databases Voting and Hypothyroid. However, when k is larger than 1, the optimal rule set approach will be much more efficient than the multiple C4.5 approach because the generating time for the optimal rule approach will not increase whereas the number of trees for the multiple tree approach will increase at a binomial coefficient

rate.

	Mushrooms		Voting	
Rule set	Size	Time (sec)	Size	Time (sec)
Optimal	312	18	1133	6
1-optimal	67	18	127	6
0-optimal	39	18	57	6
Multiple c4.5Rules ($k = 1$)	46	195	32	1
Single c4.5Rules	16	9	7	<1
	Anneal		Hypothyroid	
Rule set	Size	Time (sec)	Size	Time (sec)
Optimal	219	2	146	44
1-optimal	70	2	56	44
0-optimal	44	2	32	44
Multiple c4.5Rules ($k = 1$)	70	20	21	8
Single c4.5Rules	22	<1	7	<1

Table 4.3: Overall comparison in size and generation time of different rule sets

In our experiments, all rule sets are tested without default prediction. This is because the default prediction may disguise the true accuracy. Consider database Hypothyroid where 95.2% records belong to class Negative and 4.8 % records belong to class Hypothyroid: if we set the default prediction as Negative, then a classifier without any rule will give 95.2% accuracy. Clearly, this accuracy is misleading. In addition, this database also provides a good example for the necessity of local support. 5% global support is very small for the Negative class, but is too large for the Hypothyroid class.

We evaluated the predictive power of a rule set by the identification accuracy, which is the accuracy without default prediction.

Identification accuracy = (the number of identified instances) / (the number of all instances in a database)

The identification accuracy is the proportion of identified instances by the rule set in a database. The higher the accuracy, the better the predictive power. Its range is between 0 to 100%.

We tested all generated rule sets on l-incomplete test databases ($0 \leq l \leq 6$) of four databases, and reported their identification accuracy in Figures 4.2 and 4.3. In our experiment, the number of missing values is compared with the training data. When a training database already has missing values, then the missing values are additional. Each point in Figures 4.2 and 4.3 is the average of ten trials.

From Figures 4.2 and 4.3, we can see that when test data is incomplete, a rule set from single C4.5Rules performs poorly while both "precise" (the optimal rule set approach) and approximate (the multiple C4.5Rules) k-optimal rule sets perform significantly better. In all cases, an optimal rule set performs best, and a 1-optimal rule set the second best. These results are consistent with Theorems 4.1 and 4.2. Rule sets from multiple C4.5Rules perform better than those from single c4.5Rules but worse than 1-optimal rule sets. This result coincides with our analysis in 4.4.1. We also note that all 1-optimal rule sets perform exactly the same as the optimal rule sets when the number of missing values is not more than 1 per record as stated by Lemma 4.2. We also note that a 1-optimal rule set performs better on incomplete test databases than an approximate 1-optimal rule set from the multiple tree method. Further, approximate k-optimal rule sets (from multiple C4.5Rules) perform unstably: they sometimes perform better than 0-optimal rule set, but sometimes not.

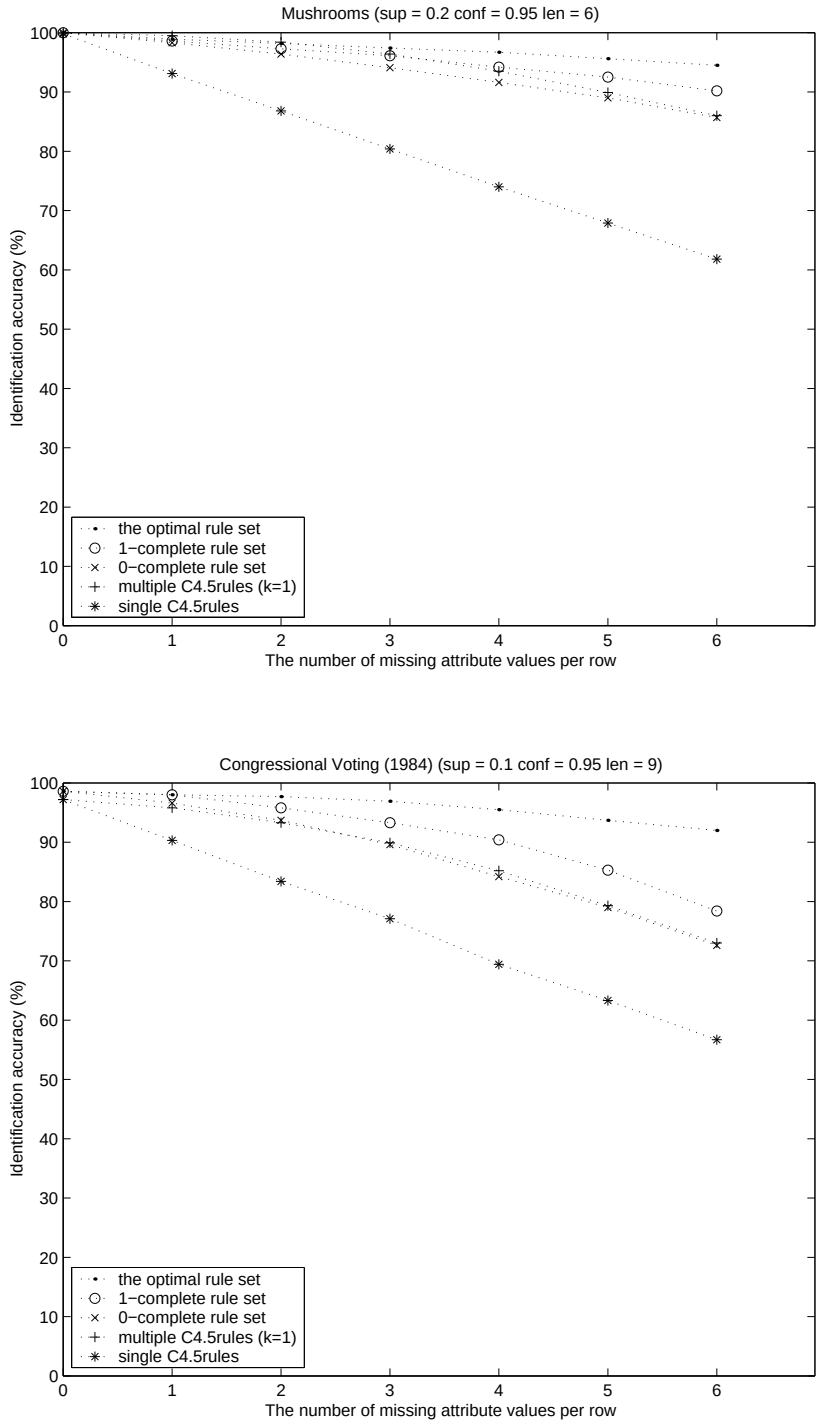


Figure 4.2: The comparison of robustness of different rule sets (1)

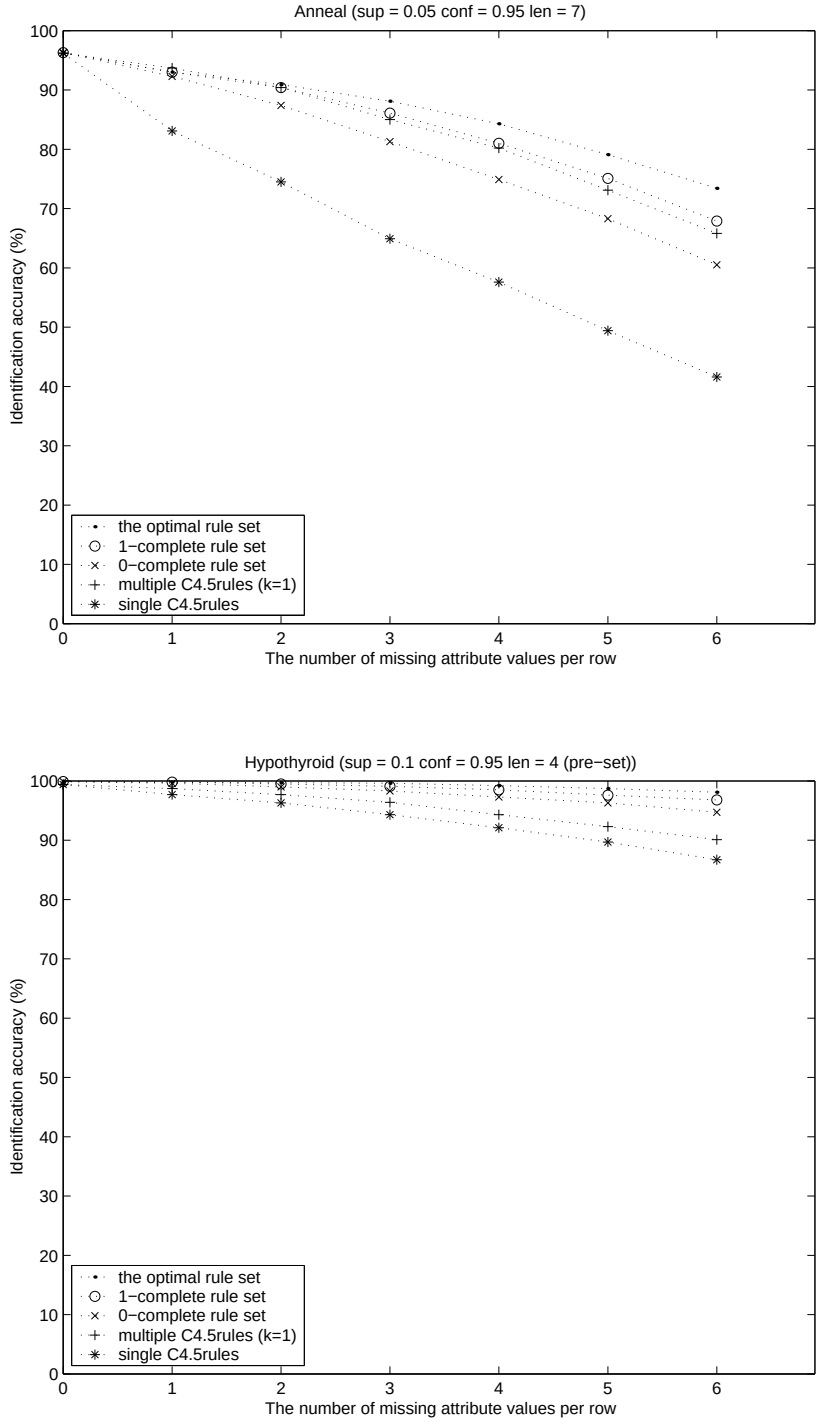


Figure 4.3: The comparison of robustness of different rule sets (2)

4.6 Discussion

In this part, we will present further discussions on the relationships between association rule sets and classification rule sets.

4.6.1 Association rule sets and classification rule sets

The goal of association rule mining is to find all rules based on some hard thresholds, such as the minimum support and the minimum confidence. However, the rule set may not cover the whole database. The purpose of classification rule mining is to find a small rule set to best fit a training database by a heuristic criterion, and no hard thresholds are involved in a traditional classification rule generating algorithm. An apparent difference is that a classification rule set is small while an association rule is usually very large. There are some other differences listed as follows. Classification rules are generated from relational databases for solving classification problems, whereas association rules are originally generated from transaction databases for solving market basket problems. Classification rules always have pre-specified consequences (classes) that never appear in the antecedent of a classification rule, whereas association rules usually have no pre-specified consequences and the consequences of an association rule may be in the antecedent of another association rule. A classification rule mining algorithm usually uses a heuristic criterion and cannot guarantee finding the optimal rule set, whereas an association rule mining algorithm obtains all rules satisfying the requirements. In spite of these differences, we can consider a classification rule set as a subset of the association rule set with pre-specified targets after a relational database is mapped into a transaction database. Next, we will discuss the necessary of the minimum support and confidence in classification rule generation.

Usually, a classification rule is highly accurate, and all classification rule mining algorithms have an implicit minimum accuracy requirement. For example a stopping criterion for constructing a decision tree is purity (a node contains only instances of one class), equivalently 100% accuracy. Normally, we may have several options for the minimum accuracy of a classification rule. The base minimum accuracy for classification

rule $X \rightarrow c$ is $sup(c)$ because a rule whose accuracy is less than that of a random guess is useless in prediction. 0.5 is another option since it avoids conflicting rules in the generated rule set. In practice, the minimum accuracy requirement is very high, for example, in our experiment the minimum accuracy is 95%.

An argument against the need for the minimum support is that a highly accurate rule is useful regardless of its support because it at least identifies some instances. However, the over-fitting problem of a classification rule set is largely due to the lack of the minimum support requirement. We will show how generated rules are misleading when there is no support constraint. For example, there are two rules from the example in the introduction when there is not the minimum support requirement

1. If outlook is sunny, temperature is mild and the wind is strong, then play tennis (1/14, 100%), and

2. If outlook is sunny, temperature is mild and the wind is weak, then do not play tennis (1/14, 100%).

These rules are clearly wrong by common sense since they suggest playing tennis if wind is strong and not playing tennis if wind is weak. Thus, the minimum support is necessary to avoid those rules fitting noisy data. In fact, a variation of support requirement has been used in the practice of classification rule mining, the bias of simplicity, which we will discuss in the next subsection.

4.6.2 Simplicity vs. robustness

The bias for traditional classification rule learning is a simple rule set fitting the training database because a small fitted rule set usually provides higher accurate predictions on unseen test data than a large fitted rule set. From our understanding, this requirement is a variant of the minimum support requirement. Rules in a small fitted rule set have higher support than those in a large fitted rule set for a database. Statistically, a high support rule is more reliable than a low support rule and hence test accuracy is more consistent with training accuracy than a low support rule. After we set the minimum support of a rule, this bias loses most of its reason for existence.

Another argument in favor of for simplicity is that a simple rule set is more understandable. Even further, to make the rule set more simple, a classifier, which is a sequence of rules and a default prediction, is actually used to make predictions. A classifier makes predictions in the following way: for a test record, use the highest precedence rule to check if it matches the test record. If so, then the class of the record is predicted to be the consequence of the rule. If not, try the next highest precedence rule. If no rule can make a prediction on the record, the prediction is the default class. The reason why a classifier is more simple than a rule set is that a default class may function as several rules.

However, such a simple classifier makes predictions hard to understand because of its "grey" box like mechanism. A prediction made by a classifier may be determined by a default value. Hence, a simple classifier is not necessary more understandable. In contrast, a rule set, which may be a little larger, make predictions more understandable, because it always connected a predictions with a rule. Consider a rule set is manipulated by computers, several dozens of additional rules make almost no additional cost in manipulation, but their robustness will be improved greatly as shown in experiments.

Robustness is very important in applying a rule set. Incomplete data are commonplace in practice, so applicability of a rule set will be very limited if it is not robust. We have revealed that a traditional classification rule set is less robust than a k-optimal rule set. Here, we further state that a classifier is less robust than a classification rule set. This is because a rule set may give a null prediction whereas a classifier may give a wrong prediction for an incomplete record. Besides, the default prediction may make the predictive accuracy unreliable. For example, a doctor may diagnose all patients who see him as normal (his default prediction), and still obtain 95% accuracy. This is very dangerous.

4.7 Conclusion

In this chapter, we discussed a new problem, finding robust rule sets to predict on a test database that is not as complete as the training database. We defined a criterion to compare the robustness for different rule sets from a database. We revealed that the optimal rule set is as robust as the complete rule set with the smallest size, and defined k-optimal rule sets for test databases with limited missing attribute values to obtain simple rule sets. We characterized the relationships among k-optimal rule sets and a traditional classification rule set. We proposed a method to find k-optimal sets through the optimal association rule approach. We showed experimentally that a k-optimal rule set generated from the proposed algorithm performs better than a k-optimal rule set generated by an extension of C4.5Rules on incomplete test databases, and that both rule sets perform significantly better than a traditional classification rule set on incomplete test databases.

Given the frequent missing value in real world databases, the k-optimal rule sets have significant potential in future applications.

Since Bagging and Boosting use multiple classifiers, their robustness properties deserves future study.

Chapter 5

Conclusion

5.1 Summary

In this thesis, we introduced the informative association rule set for market basket predictions in transaction databases and the optimal class association rule set for classification in relational databases, and proposed two efficient direct methods to generate them. We also studied how to compare the robustness of different rule sets, and introduced methods to generate rule sets that are more robust than those from a traditional classification rule generation method. In addition, we proposed an adaptive generalized quantitative association mining method for traditional association rule mining without pre-specified consequences, which is experimentally shown to be better than equal-width and equal-depth discretization methods.

In mining association rules in transaction databases, we have defined a new rule set, namely the informative association rule set, which presents predictive sequences equal to those presented by an association rule set by the confidence priority. The informative rule set is significantly smaller than the association rule set, especially when the minimum support is small. We have studied the relationships between the informative rule set and the non-redundant association rule set, revealed that the informative rule set is a subset of the non-redundant association rule set for a given database, and showed experimentally that the informative association rule set is significantly smaller than

the non-redundant rule set. We have also studied the upward closure properties of the informative rule set for omission of uninformative association rules, and presented a direct algorithm to efficiently generate the informative rule set without generating all frequent itemsets. The proposed algorithm accesses a database less often than other direct algorithms, and generates the informative association rule set more efficiently than Apriori for generating the association rule set, which is an intermediate rule set for the naive informative rule set generation. The experimental results confirm that the informative rule set is significantly smaller the association rule set and can be generated more efficiently. The experimental results also show that this efficiency improvement is a result of fewer candidates generated and fewer database accesses required by the proposed algorithm than by Apriori rather than more memory as did in some other efficient association rule mining algorithms.

For mining quantitative association rules without pre-specified consequences as in traditional association rule mining, we presented an adaptive numerical attribute merging algorithm for generating quantitative association rules which generalizes both equal-width and equal-depth methods and is better than both of them as shown in our experiments.

For mining classification rules from relational databases through association rule mining approach, we studied the problem of efficiently generating predictive class association rules from dense relational databases. We defined the optimal class association rule set, which contains all potentially predictive class association rules. Hence it can be used as a substitute of the complete class association rule set for finding predictive association rules. We presented an efficient algorithm for generating the optimal class association rule set. Our algorithm avoids much redundant computation required in mining the complete class association rule set, and hence improves the efficiency of mining predictive association rules significantly. We analysed the efficiency of the proposed algorithm theoretically and confirmed the analysis by testing the algorithm on some real world databases. Our experimental results show that the optimal class association rule set has a significantly smaller size and requires significantly less time

to generate in comparison with the complete class association rule set.

In appearance, the informative association rule set and the optimal class association rule set are very similar. Here I mainly depict their differences. The informative rule set is generated from transaction databases for market basket predictions, whereas the optimal rule set is generated from relational databases for classification. In the informative association rule set, any item can be the consequence of a rule. In the optimal class association rule set, only classes can be consequences of rules and they never appear in the antecedent of a rule. In the design of algorithms for generating both rule sets, we use the same upwards closure properties for forward pruning. These properties are more efficient in optimal rule set generation because the density of relational databases is higher. The two algorithms differ in the storage structures for rule candidates and the implementation of pruning.

For classification rule generation, we discussed a new problem, finding robust rule sets to predict on a test database that is not as complete as the training database. We defined a criterion to compare the robustness for different rule sets from a database. We revealed that the optimal rule set is as robust as the complete rule set with the smallest size, and defined k-optimal rule sets for test data with limited missing attribute values. We characterized the relationships among k-optimal rule sets and a traditional classification rule set. We introduced methods to generate k-optimal rule sets from two different approaches. We showed experimentally that k-optimal rule sets perform better than traditional classification rule sets on incomplete test data, and a k-optimal rule set generated by the optimal rule set approach performs better than one generated by an extension from C4.5Rules.

We explored the relationships between association rule sets and classification rule sets. Conceptually, a complete class association rule set is a set of target constrained association rules, an optimal class association rule set is a set of all potentially predictive class association rules, and an actually predictive class association rule set (classification rule set) is a subset of an optimal class association rule set. In between a traditional classification rule set and the optimal class association rule set, there are a hierarchy

of k-optimal rule sets where increasing size corresponds to increasing robustness.

5.2 Future work

Future research arising from work in this thesis will be in the following problems.

Market basket prediction is of relevance to e-commerce applications. In this thesis we used a model that is extended from the traditional classification where accuracy is the most important criterion. However, in practice some commercial criteria may also be important. How to build a market basket prediction model for e-commerce application remains a future research topic.

In the generation of all rule sets in this thesis, two thresholds, namely the minimum support and the minimum confidence, are required, and this is clearly an obstacle for automatic rule generation. We may set them as low as possible to include all predictive information, but the efficiency of rule generation will be significantly affected. Hence, how to automatically choose suitable thresholds remains a crucial step to make optimal rule mining automatic.

The algorithms presented in this thesis are very efficient in comparison with other existing algorithms, but may still not be satisfactory for the rapidly growing database sizes. Hence how to use sampling or parallel techniques to speed up rule mining algorithms remains a problem for future work.

In Chapter 4 we revealed some limitations of traditional rule based classifiers, such as poor understandability by default prediction and low robustness. Our proposed rule based prediction model is to make a prediction based on a rule. It is interesting to see a prediction model which makes a prediction based on several rules.

In this thesis, we explored the robustness of rule sets and revealed that a rule set from multiple decision trees is more robust than one from a single decision tree. As to the classification based on multiple classifiers, some other techniques, such as Bagging and Boosting, have been studied before. The robustness of these composite classifiers is an interesting topic for further investigation.

In this thesis we consistently preferred more accurate rules to less accurate rules.

However, there are cases where less accurate rules may be preferable; for example, we may prefer to use rules that match more attributes of a test record than than rules that match fewer attributes. Whether and when to consider such less accurate rules are questions that deserve further study.

Although rule generation has been studied for many years, we believe it will continue to be studied for many more years. This is because rules are a simple and useful form of knowledge representation for data description and prediction. The problems indicated above suggest that more research is needed to find more efficient and effective ways of generating and using rule sets for practical applications.

Appendix A

Brief descriptions of databases used in some experiments

Databases	Size	Attribute number	Class number
Anneal	898	38 Categorical	5
Chess	3196	36 Categorical	5
Congressional(House) Voting	435	16 Categorical	2
Hypothyroid	3163	25 Categorical	2
Mushrooms	8124	22 Categorical	2
Sick	3772	29 Categorical discretized by [54]	2
Heart Disease	270	7 Numerical + 6 Categorical	2
Glass Identification	214	9 Numerical	3
Iris Plant	150	4 Numerical	3
Wisconsin Breast Cancer (original)	699	10 Numerical	2
More details please refer to [15]			

Table A.1: Brief descriptions of databases used in some experiments

Bibliography

[1] R. Agarwal, C. Aggarwal, and V. Prasad. A tree projection algorithm for generation of frequent itemsets. In *Proceedings of the High Performance Data Mining Workshop*, Puerto Rico, 1999.

[2] R. Agrawal, T. Imielinski, and A. Swami. Mining associations between sets of items in massive databases. In *Proc. of the ACM SIGMOD Int'l Conference on Management of Data*, pages 207–216, 1993.

[3] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. Fast discovery of association rules. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Fayyad U. and et al, editors, Advances in Knowledge Discovery and Data Mining*, pages 307–328. MIT Press, 1996.

[4] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the Twentieth International Conference on Very Large Databases*, pages 487–499, Santiago, Chile, 1994.

[5] R. Agrawal and R. Srikant. Mining generalized association rules. In *Proc. 21st Int. Conf. Very Large Data Bases, VLDB*, pages 407–419. Morgan Kaufmann, 1995.

[6] R. Agrawal and R. Srikant. Mining sequential patterns. In *Proc. 11th Int. Conf. Data Engineering, ICDE*, pages 3–14. IEEE Press, 1995.

[7] D. W. Aha, D. Kibler, and M. K. Albert. Instance-based learning algorithms. *Machine Learning*, 6(1):37–66, Jan 1991.

[8] K. Ali, S. Manganaris, and R. Srikant. Partial classification using association rules. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97)*, page 115, Menlo Park, CA, 1997. AAAI Press.

[9] A. Amir, R. Feldman, and R. Kashi. A new and versatile method for association generation. In *Proceedings of the 1st European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD97)*, volume 1263 of *LNAI*, pages 221–231, Berlin, 1997. Springer.

[10] N. F. Ayan, A. U. Tansel, and E. Arkun. An efficient algorithm to update large itemsets with early pruning. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 287–291, N.Y., 1999. ACM Press.

[11] R. Bayardo and R. Agrawal. Mining the most interesting rules. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 145–154, N.Y., 1999. ACM Press.

[12] R. Bayardo, R. Agrawal, and D. Gunopulos. Constraint-based rule mining in large, dense database. In *Proc. of the 15th Int'l Conf. on Data Engineering*, pages 188–197, 1999.

[13] R. J. Bayardo. Efficiently mining long patterns from databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD-98)*, ACM SIGMOD Record 27(2), pages 85–93, New York, 1998. ACM Press.

[14] E. A. Bender. *Mathematical Methods in Artificial Intelligence*. IEEE Computer Society Press, 1996.

[15] E. Keogh C. Blake and C. J. Merz. UCI repository of machine learning databases, http://www.ics.uci.edu/∼mlearn/MLRepository.html, 1998.

[16] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

[17] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth International Group, 1984.

[18] S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: Generalizing association rules to correlations. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 26(2):265, 1997.

[19] S. Brin, R. Motwani, J. D. Ullman, and Shalom Tsur. Dynamic itemset counting and implication rules for market basket data. In *Proceedings, ACM SIGMOD International Conference on Management of Data: SIGMOD 1997: May 13–15, 1997, Tucson, Arizona, USA*, volume 26(2), pages 255–264, NY, USA, 1997. ACM Press.

[20] J. Catlett. On changing continuous attributes into ordered discrete attributes. In *Proceedings of the European Working Session on Learning : Machine Learning (EWSL-91)*, volume 482 of *LNAI*, pages 164–178, Porto, Portugal, 1991. Springer Verlag.

[21] J. Cerquides and R. L. de Màntaras. Proposal and empirical comparison of a parallelizable distance-based discretization method. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97)*, page 139. AAAI Press, 1997.

[22] P. Clark and R. Boswell. Rule induction with CN2: Some recent improvements. In *Machine Learning - EWSL-91*, pages 151–163, 1991.

[23] P. Clark and T. Niblett. The CN2 induction algorithm. *Machine Learning*, 3(4):261–283, 1989.

[24] S. Clearwater and F. Provost. RL4: A tool for knowledge-based induction. In *the Second International IEEE Conference on Tools for Artificial Intelligence*, pages 24–30, 1990.

[25] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J. Ullman, and C. Yang. Finding interesting associations without support pruning. In *16th*

International Conference on Data Engineering (ICDE' 00), pages 489–500, Washington - Brussels - Tokyo, 2000. IEEE.

[26] G. Dong and J. Li. Interestingness of discovered association rules in terms of neighborhood-based unexpectedness. *Lecture Notes in Computer Science*, 1394:72–86, 1998.

[27] G. Dong, X. Zhang, L. Wong, and J. Li. CAEP: Classification by aggregating emerging patterns. In *Proceedings of the 2nd International Conference on Discovery Science (DS-99)*, volume 1721 of *LNAI*, pages 30–42, Berlin, 1999. Springer.

[28] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *Proc. 12th International Conference on Machine Learning*, pages 194–202. Morgan Kaufmann, 1995.

[29] X. Du, Z. Liu, and N. Ishii. Mining association rules on related numeric attributes. In *Proceedings of the 3rd Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining (PAKDD-99)*, volume 1574 of *LNAI*, pages 44–53, Berlin, 1999. Springer.

[30] U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93)*, pages 1022–1027, San Francisco, 1993. Morgan Kaufmann.

[31] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining*. AIII Press/MIT Press, March 1996.

[32] A. A. Freitas. On objective measures of rule surprisingness. In *Principles of Data Mining and Knowledge Discovery (PKDD'98)*, pages 1–9. Springer-Verlag, 1998.

[33] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *International Conference on Machine Learning*, pages 148–156, 1996.

[34] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[35] N. Friedman, D. Geiger, M. Goldszmidt, G. Provan, P. Langley, , and P. Smyth. Bayesian network classifiers. *Machine Learning*, 29:131, 1997.

[36] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Data mining using two-dimensional optimized association rules: scheme, algorithms, and visualization. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada, June 4–6, 1996*, pages 13–23, New York, 1996. ACM Press.

[37] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Mining optimized association rules for numeric attributes. In *Proceedings of the Fifteenth ACM Symposium on Principles of Database Systems, PODS 1996*, pages 182–191. ACM Press, 1996.

[38] J. Fürnkranz. Separate-and-conquer rule learning. *Artificial Intelligence Review*, 13(1):3–54, January 1999.

[39] R. Goodman and P. Smyth. Information-theoretic rule induction. In *Proceedings of the 8th European Conference on Artificial Intelligence*, pages 357–362. Pitman Publishers, 1988.

[40] D. Gunopulos, R. Khardon, and H. Mannila. Data mining, hypergraph transversals, and machine learning (extended abstract). In *Proceedings of the Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 209–216. ACM Press, 1997.

[41] J. Han and M. Kamber. *Data mining: Concepts and Techniques*. Morgan Kaufmann, 2001.

[42] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proc. 2000 ACM-SIGMOD Int. Conf. on Management of Data (SIGMOD'00)*, pages 1–12, May, 2000.

[43] D. J. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining*. MIT Press, 2001.

[44] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1975.

[45] C. Hidber. Online association rule mining. In *Proc. ACM SIGMOD Int. Conf. Management of Data*, pages 145–156, 1999.

[46] R. J. Hilderman and H. J. Hamilton. Heuristics for ranking the interestingness of discovered knowledge. *Lecture Notes in Computer Science*, 1574:204–209, 1999.

[47] K. M. Ho and P. D. Scott. Zeta: A global method for discretization of continuous variables. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97)*, page 191. AAAI Press, 1997.

[48] J. H. Holland. Escaping brittleness: the possibilities of general purpose algorithms applied to parallel rule-based systems. In R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, editors, *Machine Learning, an Artificial Intelligence approach*, volume 2, pages 593–623. Morgan Kaufmann, San Mateo, California, 1986.

[49] M. Holsheimer, M. Kersten, H. Mannila, and Toivonen. A perspective on databases and data mining. In *1st Intl. Conf. Knowledge Discovery and Data Mining*, page 10, 1995.

[50] M. Houtsma and A. Swami. Set-oriented mining for association rules in relational databases. In *Proceedings of the 11th International Conference on Data Engineering*, pages 25–34, Los Alamitos, CA, USA, 1995. IEEE Computer Society Press.

[51] F. Hussain, H. Liu, E. Suzuki, and H. Lu. Exception rule mining with a relative interestingness measure. In *Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining (PAKDD 00)*, pages 86–97. LNCS, 2000.

[52] M. Kamber and R. Shinghal. Proposed interestingness measure for characteristic rules. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*, page 1393, Menlo Park, 1996. AAAI Press / MIT Press.

[53] R. Kohavi and M. Sahami. Error-based and entropy-based discretization of continuous features. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD 96)*, pages 114–119. AAAI Press, 1996.

[54] R. Kohavi, D. Sommerfield, and J. Dougherty. Data mining using MLC++: A machine learning library in C++. In *Tools with Artificial Intelligence*, pages 234–245. IEEE Computer Society Press, 1996.

[55] B. Lent, A. Swami, and J. Widom. Clustering association rules. In *Proceedings of the 13th International Conference on Data Engineering (ICDE'97)*, pages 220–231, Washington - Brussels - Tokyo, 1997. IEEE.

[56] K. Leonard and R. Peter. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley-Interscience Publication, 1990.

[57] J. Li, H. Shen, and P. Pritchard. Knowledge network based association discovery. In *Proc. of 1999 International Conference on Parallel and Distributed Processing Techniques and Applications, (PDPTA'99)*, pages 1558–1563, 1999.

[58] J. Li, H. Shen, and P. Pritchard. Mining significant association rules. In *Proc. of 1999 International Conference on Parallel and Distributed Processing Techniques and Applications, (PDPTA'99)*, pages 1458–1461, 1999.

[59] J. Li, H. Shen, and R. Topor. An adaptive method of numerical attribute merging for quantitative association rule mining. In *The 5th International Computer Science Conference (ICSC'99)*, pages 41–50. Springer, 1999.

[60] J. Li, H. Shen, and R. Topor. Mining optimal class association rule set. In *Proceedings of the 5th Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining (PAKDD 2001)*, pages 364–375. Springer, 2001.

[61] J. Li, H. Shen, and R. Topor. Mining the smallest association rule set for prediction. In *Proceedings of 2001 IEEE International Conference on Data Mining (ICDM 2001)*, pages 361–368. IEEE Computer Society Press, 2001.

[62] J. Li, H. Shen, and R. Topor. Mining the optimal class association rule set. *Knowledge-Based System*, 15(7):399–405, 2002.

[63] J. Li, R. Topor, and H. Shen. Construct robust rule sets for classification. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD02)*, pages 564–569, Edmonton, Canada, 2002.

[64] W. Li, J. Han, and J. Pei. CMAR: Accurate and efficient classification based on multiple class-association rules. In *Proceedings 2001 IEEE International Conference on Data Mining (ICDM 2001)*, pages 369–376. IEEE Computer Society Press, 2001.

[65] D.-I. Lin and Z. M. Kedem. Pincer search: A new algorithm for discovering the maximum frequent set. *Lecture Notes in Computer Science*, 1377:105, 1998.

[66] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98)*, pages 27–31, 1998.

[67] B. Liu, W. Hsu, and Y. Ma. Mining association rules with multiple minimum supports. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 337–341, N.Y., 1999. ACM Press.

[68] B. Liu, W. Hsu, and Y. Ma. Pruning and summarizing the discovered associations. In *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining (SIGKDD 99)*, 1999.

[69] B. Liu, Y. Ma, and C. Wong. Improving an association rule based classifier. In *4th European Conference on Principles and Practice of Knowledge Discovery in Databases PKDD*, pages 504–509, 2000.

[70] B. Liu, Y. Ma, and C. K. Wong. Improving an exhaustive search based rule learner. In *Proceedings of the Fourth European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD-2000)*, pages 13–16, 2000.

[71] J. A. Major and J. Mangano. Selecting among rules induced from a hurricane databases. In *Proc. AAAI-93 Workshop on Knowledge Discovery in Databases*, pages 28–44, 1993.

[72] H. Mannila and H. Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery Journal*, 1(3):241–258, 1997.

[73] H. Mannila, H. Toivonen, and A. I. Verkamo. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery Journal*, 1(3):259–289, 1997.

[74] H. Mannila, H. Toivonen, and I. Verkamo. Efficient algorithms for discovering association rules. In *AAAI Wkshp. Knowledge Discovery in Databases*, pages 181–192. AAAI Press, July 1994.

[75] D. Meretakis and B. Wüthrich. Extending naive Bayes classifiers using long itemsets. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 165–174, N.Y., 1999. ACM Press.

[76] R. Michalski, I. Mozetic, J. Hong, and N. Lavrac. The AQ15 inductive learning system: an overview and experiments. In *Proceedings of IMAL 1986*, Orsay, 1986. Université de Paris-Sud.

[77] R. S. Michalski. Pattern recognition as rule-guided inductive inference. In *Proceedings of IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 349–361, 1980.

[78] R. J. Miller and Y. Yang. Association rules over interval data. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 26(2):452, 1997.

[79] J. Mingers. An empirical comparison of selection measures for decision tree induction. *Machine Learning*, 3:319–342, 1989.

[80] T. M. Mitchell. *Machine Learning.* McGraw-Hill, 1997.

[81] P. M. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. *IEEE Transactions on Computers*, 26:917–922, 1977.

[82] R. Ng, L. Lakshmanan, J. Han, and A. Pang. Exploratory mining and pruning optimizations of constrained associations rules. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD-98)*, ACM SIGMOD Record 27(2), pages 13–24, New York, 1998. ACM Press.

[83] G. Pagallo and D. Haussler. Boolean feature discovery in empirical learning. *Machine Learning*, 5(1):71–99, 1990.

[84] J. S. Park, M. Chen, and P. S. Yu. An effective hash based algorithm for mining association rules. In *ACM SIGMOD Intl. Conf. Management of Data*, 1995.

[85] M.J. Pazzani, C.A. Brunk, and G. Silverstein. A knowledge-intensive approach to learning relational concepts. In *Proceedings of the 8th International Workshop on Machine Learning*, pages 432–436. Morgan Kaufmann, 1991.

[86] G. Piatetsky-Shapiro. Discovery, analysis and presentation of strong rules. In G. Piatetsky-Shapiro, editor, *Knowledge Discovery in Databases*, pages 229–248. AAAI Press / The MIT Press, Menlo Park, California, 1991.

[87] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[88] J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5(3):239–266, 1990.

[89] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[90] B. D. Ripley. *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge, 1996.

[91] D. E. Rumelhart and J. L. McClelland. Learning internal representations by error propogation. In *Explorations in the Micro-Structure of Cognition Vol. 1 : Foundations*, pages 318–362. MIT Press, Cambridge, MA, 1986.

[92] R. Rymon. Search through systematic set enumeration. In *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning*, pages 539–552, Cambridge, MA, oct 1992. Morgan Kaufmann.

[93] R. Rymon. An SE-tree based characterization of the induction problem. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 268–275, Amherst, 1993. Morgan Kaufmann.

[94] A. Savasere, R. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in large databases. In *Proceedings of 21th International Conference on Very Large Data Bases (VLDB95)*, pages 432–444, 1995.

[95] J. C. Schlimmer. Efficiently inducing determinations: A complete and systematic search algorithm that uses optimal pruning. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 284–290, Amherst, 1993. Morgan Kaufmann.

[96] R. Sedgewick. *Algorithms*, pages 145–162. Addison-Wesley Series in Computer Sience. Addison-Wesley Publishing Company, Inc., second edition, 1988.

[97] R. Segal and O. Etzioni. Learning decision lists using homogeneous rules. In *Proceedings of the 12th National Conference on Artificial Intelligence. Volume 1*, pages 619–625, Menlo Park, CA, USA, 1994. AAAI Press.

[98] L. Shen, H. Shen, and L. Cheng. New algorithms for efficient mining of association rules. *Information Sciences*, 118:251–268, 1999.

[99] P. Shenoy, J. R. Haritsa, S. Sudarshan, G. Bhalotia, M. Bawa, and D. Shah. Turbo-charging vertical mining of large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD-99)*, ACM SIGMOD Record 29(2), pages 22–33, Dallas, Texas, 1999. ACM Press.

[100] A. Siberschatz and A. Tuzhilin. On subjective measures of interestingness in knowledge discovery. In *In Proc. 1st International Conference on Knowledge Discovery and Data Mining*, pages 275–281, Montreal, Cnaada, 1995.

[101] R. Srikant and R. Agrawal. Mining generalized association rules. In *Proceedings of the 21st International Conference on Very Large Data Bases (VLDB '95)*, pages 407–419, CA, 1995. Morgan Kaufmann.

[102] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 25(2):1, 1996.

[103] S. Thomas, S. Bodagala, K. Alsabti, and S. Ranka. An efficient algorithm for the incremental updation of association rules in large databases. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97)*, page 263. AAAI Press, 1997.

[104] H. Toivonen, M. Klemettinen, P RonKainen, K Hatonen, and H. Mannila. Pruning and grouping discovered association rules. In *Workshop Notes of the ECML-95 Workshop on Statistics, Machine Learning, and Knowledge Discovery in Databases*, pages 47–52, 1995.

[105] J. Ullman. A survey of association-rule mining. In *ICDS: International Conference on Data Discovery, DS*. LNCS, 2000.

[106] K. Wang, S. H. W. Tay, and B. Liu. Interestingness-based interval merger for numeric association rules. In *Proc. 4th Int. Conf. Knowledge Discovery and Data Mining, (KDD 98)*, pages 121–128. AAAI Press, 1998.

[107] G. I. Webb. OPUS: An efficient admissible algorithm for unordered search. In *Journal of Artificial Intelligence Research*, volume 3, pages 431–465, 1995.

[108] G. I. Webb. Efficient search for association rules. In *Proceedinmgs of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-00)*, pages 99–107, N. Y., 2000. ACM Press.

[109] S. M. Weiss and N. Indurkhya. Reduced complexity rule induction. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, pages 678–684. Morgan Kaufmann, 1991.

[110] M. J. Zaki. Parallel and distributed association mining: A survey. *IEEE Concurrency*, 7(4):14–25, 1999.

[111] M. J. Zaki. Generating non-redundant association rules. In *6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 34–43, August 2000.

[112] M. J. Zaki and C. T. Ho. *Large-scale parallel data mining*, volume 1759 of *Lecture Notes in Computer Science*. Springer-Verlag Inc., New York, NY, USA, 2000.

[113] M. J. Zaki and C. J. Hsiao. Charm: An efficient algorithm for closed association rule mining. In *2nd SIAM International Conference on Data Mining (to appear)*, 2002.

[114] M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. New algorithms for fast discovery of association rules. In *Proceedings of the Third International Con-*

ference on Knowledge Discovery and Data Mining (KDD-97), page 283. AAAI Press, 1997.

[115] Z. Zheng, R. Kohavi, and L. Mason. Real world performance of association rule algorithms. In *Proceedings of the seventh International Conference on Knowledge Discovery and Data Mining (SIGKDD-01)*, 2001.