Mining Interesting Rules from a Large Set of Association Rules

By

Muhammad Sheikh Sadi

A Thesis Submitted in partial fulfillment of the requirements for the degree of "Masters of Science in Computer Science and Engineering"

Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology
Dhaka, Bangladesh
December, 2004

The thesis titled **"Mining Interesting Rules from a Large Set of Association Rules"** submitted by Muhammad Sheikh Sadi Roll No. 100105004 Session OCT 2001 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of Master of Science and Engineering in Computer Science and Engineering (M.Sc. Engg.) held on ..08.12.2004..

BOARD OF EXAMINERS

1. _____ Chairman
Dr. Md. Monirul Islam
(Supervisor)
Assistant Professor
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology (BUET)
Dhaka-1000, Bangladesh

2. _____ 8/12/04 Member
Dr. Md. Shamsul Alam (Ex-officio)
Professor & Head
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology (BUET)
Dhaka-1000, Bangladesh

3. _____ Member
Dr. Mohammad Kaykobad
Professor
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology (BUET)
Dhaka-1000, Bangladesh

4. _____ Member
Dr. Hafiz Muhammad Hasan Babu (External)
Associate Professor and Chairman
Department of Computer Science and Engineering
Dhaka University
Dhaka-1000, Bangladesh

ACKNOWLEDGEMENT

First and foremost I would like to thank my supervisor, Dr. Md. Monirul Islam, who gave me the intellectual latitude to perform this task and provided such a stimulating yet safe haven for my research. His guidance, cooperation and suggestion were a continual source of inspiration. His dedication to his chosen field, genuine love of science and incisive advice was a welcome anchor throughout my graduate career.

I would like to thank Professor Dr. Md. Shamsul Alam, Head, Department Computer Science and Engineering, BUET for his support and continuous encouragement throughout the research work.

I am also grateful to Professor Dr. Mohammad Kaykobad for his guidance and cooperation. I wish to give special thanks to Professor Dr. Chowdhury Mofizur Rahman who befriended me just as I began exploring data mining techniques and encouraged me to explore them fully.

I would also like to thank Mr. Mohammad Mehedy Masud and Mr. Abu Wasif, faculty members of CSE department, BUET for their kind cooperation in my work and for providing such a wonderful distraction during the preparation of this document. I am also grateful to all of the teachers of CSE department, all of my friends and all of the wonderful folks who run the department of Computer Science and Engineering.

Abstract

The goal of data mining is to discover knowledge and reveal new, interesting and previously unknown information to the user. A central data-mining tool is *association rules*. This thesis work presents a constructive approach to mine interesting rules (CAMIR) from a large set of association rules. CAMIR first enumerates primitive rules with large interestingness and then generates rules with value grouping by merging interesting primitive rules. It assumes all attributes are nominal and there is no missing attribute value. Inputs of the algorithm are the number of attributes in bodies of rules, lower bound of support, accuracy and interestingness. The essence of CAMIR is that it requires minimum number of user specified parameters and users do not require any prior knowledge about the target problem. Here, the computation of interestingness of rules is simple and generalized. It has been applied on a number of benchmark data sets. They are mushroom, letter recognition, breast cancer, statlog (heart), hepatitis and liver disorders. The experimental result shows that CAMIR can return a large number of interesting rules within shorter interval of time in comparison with other existing algorithms.

Table of Contents

List of Figures

List of Tables

List of Elaborations

DIG Discover Interesting Rules with Grouping Attribute Values

CAMIR Constructive approach to Mine Interesting Rules

KDD Knowledge Discovery in Databases

Chapter 1

Introduction

- Background
- Existing Works
- Objectives of the Thesis
- Thesis Organization

1.1 Background

Data mining in general is the search for hidden patterns that may exist in large databases. It can be defined as the non-trivial extraction of implicit, previously unknown and potentially useful knowledge from data [1-3]. Databases today can range in size into the terabytes — more than 1,000,000,000,000 bytes of data. Within these masses of data, useful information lies in hidden form. But when there are so many trees, how do you draw meaningful conclusions about the forest? The newest answer is data mining, which is being used both to increase revenues through improved marketing and to reduce costs through detecting and preventing waste and fraud.

Data mining finds patterns and relationships in data by using sophisticated techniques to build abstract representations of reality. A good model is a useful guide to understand business and making decisions. There are two main kinds of models in data mining: predictive and descriptive.

Predictive models can be used to forecast explicit values based on patterns determined from known results. For example, from a database of customers who have already responded to a particular offer, a model can be built that predicts which prospects are likeliest to respond to the same offer. Descriptive models describe patterns in existing data, and are generally used to create meaningful subgroups such as demographic clusters.

In addition to algorithms, data mining software usually has features to simplify the graphical representation of the data (visualization tools) plus interfaces to common database formats. Data mining is only one step in the knowledge discovery process. Other steps include identifying the problem to be solved, collecting and preparing the right data, interpreting and deploying models, and monitoring the results. The real key to success, however, is to have a thorough understanding of the data and of the business. Algorithms can provide meaningful results only when sensibly directed.

Innovative organizations are already using data mining to locate and to appeal higher-value customers, reconfigure their product offerings to increase sales, and minimize losses due to error or fraud. The list of data mining applications is surprisingly broad. The most beneficial use of data mining technique is in market

analysis. Several hospitals of Singapore are using data mining techniques in their practical applications.

One of the most researched areas in data mining is finding association rules from binary data [1-3]. After performing data mining tasks there are a large number of association rules [3]. An association rule is the attribute-value conditions that occur frequently together in a given set of data. More formally, association rules are of the form X=>Y, that is, "A_1^.....^A_m => B_1^.....^B_n", where A_i (for i = 1,.....,m) and B_j (for j = 1,.....n) are attribute-value pairs. The association rule X=>Y is interpreted as "database tuples that satisfy the conditions in X are also likely to satisfy the condition in Y."

An association rule is of the form X=> Y where X and Y are disjoint conjunctions of attribute-value pairs [4]. The confidence of the rule is the conditional probability of Y given X, P (Y|X), and the support of the rule is the prior probability of X and Y, P (X and Y). Here probability is taken to be the observed frequency in the data set [5].

The number of discovered association rules could however be so large that browsing such rules would be difficult for users. All of these rules haven't much importance to users. Moreover, it requires a large amount of time to work on the whole domain of rules. It is, therefore, necessary to find out interesting rules from large number of rules. Data mining techniques can uncover interesting rules hidden in *large data sets* efficiently [6].

The interestingness of rules strongly depends on what a user of the system already knows and what he or she wants to do with the discovered rules [7-8]. General rules summarize the characteristics of each class and classify many instances accurately [9-10]. If a user has no background knowledge of the domain represented by the target database, then general rules are used. General rules are not appropriate when domain experts use the system because they usually know it.

Consider an example from mushroom database in UCI Repository [4]. This database includes about 8,000 instances and each instance is classified into edible or poisonous. This database is very easy as a benchmark of supervised learning algorithms because the following two rules can classify most instances correctly:

odor $\in$ {almond, anise, none} $\rightarrow$ edible: 98%

odor $\in$ {fishy, foul, spicy} $\rightarrow$ poisonous: 100%

To extract exceptional rules, new criteria of rules [8] are needed. An exceptional rule is a specialization of general rule but concludes a different class with high accuracy [10]. General rules cover relatively many instances and may be trivial for human experts, but exceptional rules may give new knowledge to the experts. The conclusion (class) of exceptional rule holds a large region of an instance space and the rule represents only a part of the region.

1.2 Existing Works

Since the introduction of the (Boolean) Association Rules problem in [11], there has been considerable works on designing algorithms for mining such rules [5,12,13]. These works were subsequently extended to finding association rules when there is taxonomy on the items in [14]. Related work also includes [15], where quantitative rules of the form $x = q_x \Rightarrow y = q_y$ are discovered. However, the antecedent and consequent are constrained to be a single attribute value pair. There are suggestions about extending this to rules where the antecedent is of the from $l \leq x \leq u$. This is done by partitioning the quantitative attributes into intervals. However, the intervals are not combined.

To find the rules comprising (A = a) as the antecedent, where a is a specific value of the attribute A, one pass over the data is made and each record is hashed by values of A. Each hash cell keeps a running summary of values of other attributes for the records with the same A value. The summary for (A = a) is used to derive rules implied by (A = a) at the end of the pass. To find rules for different attributes, the algorithm is run once on each attribute. Thus if users are interested in finding all rules, they must find these summaries for all combinations of attributes, which is exponentially large.

One approach to filtering rules by constraining patterns of rules explicitly is shown in [13]. This approach is applied to association rule mining in which an user restricts what kind of items can appear in rules' bodies and heads. This approach is used in many applications but tends to discover only rules that the user expects beforehand. In addition, even if the constraints on rules or items are given, a discovery system may output too many rules to be checked by the user.

Another approach is to evaluate interestingness of rules with pre-defined criteria based on statistical characteristics in the target database and to accept rules with respect to the criteria [6]. This research work belongs to this second approach. The proposed method extracts interesting rules with respect to the criterion effectively. Here, interestingness is evaluated from statistical point of view. This research work evaluates the interestingness of rules by comparing the accuracy of rules. This approach requires minimum number of user specified parameters and users do not require any prior knowledge about the target problem.

Wu, X. and Zhang, S. [1] presents a weighting model for synthesizing high frequency rules from different data sources. The limitation of this model is that the system requires different data sources of equal size and it synthesizes high frequency rules but not extracts interesting rules. Silberschatz and Tuzhilin [16] discussed unexpectedness of rules and defined it by how the rule contradicts with user's knowledge. A problem of their approach is that a discovery system has to know what its user knows.

Suzuki [6] proposed a discovery system PEDRE that tries to discover pairs of general rules and their exception. PEDRE evaluates an exceptional rule by comparing with only general rules. Such exceptional rule is too specific to predict its concluding class and is not appropriate as an exceptional rule [4]. Nobuhiro Yugami et al. [4] proposed a discovery algorithm DIG to extract interesting rules based on their exceptionality over general rules. But the time requirement of DIG is very large and the number of interesting rules discovered is not large enough.

Apriori is an influential algorithm for mining frequent item sets for Boolean association rules [12]. The name of the algorithm is based on the fact that the algorithm uses prior knowledge of frequent item set properties. Apriori employs an iterative approach known as a level-wise search, where k-item sets are used to explore (k + 1)-item sets.

The Level-wise algorithms make one pass over the database for each level [13]. There are thus K or K+1 passes over the database, where K is the size of the largest frequent set. Sometimes, if there are only few candidates in the last iterations, candidates can be generated and tested for several levels at once.

The partition algorithm [12] reduces the database activity. It computes all frequent sets in two passes over the database. The algorithm works also in the level-wise manner. However, the idea is to be handled in main memory without further

database activity. The first database pass consists of identifying in each part the collection of all locally frequent sets. For the second pass, the union of the collections of locally frequent sets is used as the candidate set [13]. The first passes guaranteed to locate a superset of the collection of frequent item sets; the second pass is needed to merely compute the frequencies of the sets.

The above mentioned descriptions clearly point out limitations of the existing works. For example, the limitation of the weighting model proposed by Wu, X. and Zhang, S. is that the system requires different data sources of equal size and it synthesizes high frequency rules but not extracts interesting rules [1]. A problem of Silberschatz and Tuzhilin is that a discovery system has to know what its user knows [16]. PEDRE evaluates an exceptional rule by comparing with only general rules [6]. Such exceptional rule is too specific to predict its concluding class and is not appropriate as an exceptional rule [4]. The problem of DIG [4] is found that the time requirement of this method is very large and the number of interesting rules discovered is not large enough.

1.3 Objectives of the Thesis

The proposed research work will be conducted with a view to achieving the following goals:

- To find out a new criterion of interestingness to identify a rule corresponding to an isolated exceptional region by comparing its accuracy with plural general rules.

- To show a relative measure that can give an unbiased estimate of relative interestingness of a rule with respect to the extracted rules.

- To evaluate a rule that permits plural values for each attribute appearing in its body.

1.4 Thesis Organization

Chapter two discusses about data mining and different features of data mining. These features include necessity of data mining, several steps of the data mining process and the architecture of data mining system.

Association rules and its relevance with data mining are outlined shortly in *Chapter three*. Support and Confidence –the most salient features of association rules are also analyzed briefly in this chapter. This chapter also introduces the rule mining process and various ways of association rule mining processes.

The proposed constructive approach to mine interesting rules (CAMIR) is conversed briefly in *chapter four*. The algorithm shown in this episode clearly describes the way of generating interesting rules. Necessary terms for evaluating interestingness are argued in this chapter also. In order to evaluate interestingness, one needs the concepts of basic probability theory, classification rules and other criteria that are outlined shortly in this chapter.

Chapter five focuses the experimental studies about CAMIR. AT first different benchmark datasets that were used to test the performance of the CAMIR are described in brief. Later this chapter shows how the proposed method outperforms DIG [4] using the experimental results obtained after applying to those datasets.

Chapter six confers conclusion and recommendation with future expansion.

Chapter 2

Data Mining

- Introduction
- Categories of Data Mining
- Necessity of Data Mining
- Popularity of Data Mining
- The Steps of the Data Mining
- Architecture of Data Mining System

2.1 Introduction

Data Mining is an emerging field, connecting the three worlds of Databases, Artificial Intelligence and Statistics. The computer age has enabled people to gather large volumes of data. Every large organization amasses data on its clients or members, and these databases tend to be enormous. The usefulness of this data is negligible if "meaningful information" or "knowledge" cannot be extracted from it. Data Mining answers this need.

The field of information retrieval has developed out of a need to answer specific queries formulated by the user. So too, statistical methods may be used to verify a given hypothesis that a user may wish to check. However, this is not enough. One needs to be able to ask the questions: "What interesting information is contained in or may be learned from our database?" These questions formulate the key objectives of data mining. Currently, there is so much data that a user often does not even know what the database contains. Furthermore, confining research to a priori hypotheses is limited, as learning is restricted to what is previously suspected. In short, today's databases contain vast quantities of information that users are unable to reach. This *hidden information* is what user reveals when implementing data mining techniques. For this reason data mining is also known as "Knowledge Discovery".

Knowledge discovery implies lack of involvement on the part of the user and automatic guidance in the search. The more independent and therefore unrestricted the search, the more successful it may be in finding surprising and unexpected results. The objective here is not intelligent searching based on a given query, but rather searching and returning information that the user did not ask for and may not even expect exists within the database. In other words, in data mining the computer "informs" the user of what is interesting within the data. Often, some constraints are imposed on the search in order to reduce the computational blowup. These constrains also may help guide the search to areas of interest to the user. Tools with constraints lie on the spectrum between information retrieval and "pure" data mining.

2.2 Categories of Data Mining

The majority of data mining tools may be categorized into three types: predictive modeling, database segmentation and link analysis. Predictive modeling is a form of

what is known in Artificial Intelligence as *supervised learning*. Each transaction in the database is assigned a class label characterizing its type. The aim is to learn the basis for the classification and predict the classification of new, unseen records. A classic application for this type of tool is for a bank to learn whether they should allocate a credit card to a new customer. A learning algorithm may learn the characteristics of customers who do not pay their credit and then predict the probability that a new customer will succeed in paying his credit. Tools for predictive modeling include decision trees and neural networks.

Database segmentation is the process of dividing a database into sets of similar transactions. This method is a type of *unsupervised learning* and is commonly referred to as clustering. Clustering helps the user in two tasks. Firstly, the clusters provide a summary of the contents of the database. This is critical in text mining applications. Upon receiving a huge corpus of documents, each output cluster contains documents connected to a distinct topic. Secondly, new information may be learnt from the contents of each cluster. By understanding why the clustering algorithm grouped a certain set of transactions together, one can learn new and previously unseen patterns, especially if the grouping is not as expected.

Link analysis is employed to discover relationships between transactions or sets of transactions in the database. In contrast to predictive modeling and database segmentation, it does not look at the database as a whole rather focuses on individual or sets of transactions. There are different types of link analysis. Sequential time analysis searches for connections between events (a transaction is an event) based on both the times that they occur and their contents. For example, a sequential analysis of a database of items bought in a supermarket may discover that more milk products are bought in the morning and more meat products towards evening.

Associations' discovery is another form of link analysis and aims to find the relationship between different items within the records themselves. For example, in a supermarket database, one may find that 75% of people buying Coca-Cola also buy Sprite. The associations' problem is important for databases containing quantitative data as well as nominal or categorical data. This problem is specifically referred to as "mining quantitative associations". Association rule discovery is described in depth in the next section.

There is a new category of tools that is known as *Automatic Hypothesis Generation*. Statistical theory deals with hypothesis testing and has developed many

tests for many different classes of hypotheses. Data mining should use this theory in order to ensure the validity of information discovered. Tools that efficiently check all hypotheses of a certain type (say, linear correlations using regression tests) would be invaluable for saving time and discovering connections between variables the user would never have guessed to be connected. This type of tool certainly fulfills data mining's aim of unguided knowledge discovery. A theory of data mining based on hypothesis generation, rather than hypothesis verification as in statistics, would take the users to the next generation of data mining.

2.3 Necessity of Data Mining

Data mining has attracted a great deal of attention in the information industry in recent years due to the wide availability of huge amount of data and the imminent need for turning such data into useful information and knowledge. The information and knowledge can be used for applications ranging from business management, production control, and market analysis, to engineering design and science exploration [17].

Data mining can be viewed as a result of the natural evolution of information technology as shown in Fig. 2.1. An evolutionary path has been witnessed in the database industry in the development of the following functionalities: data collection and database creation, data management (including data storage and retrieval, and database transaction management), and data analysis and understanding (involving data warehouse and data mining) [18].

The abundance of data, coupled with the need for powerful data analysis tools, has been described as rich data but poor information situation. The fast-growing, tremendous amount of data, collected and stored in large and numerous databases, has far exceeded our human ability for comprehension without powerful tools. As a result, data collected in large databases become "data tombs" –data archives that are seldom visited. Consequently, important decisions are often made based not on the information-rich data stored in databases but rather on a decision maker's intuition. This is because the decision maker does not have the tools to extract the valuable knowledge embedded in the vast amount of data.

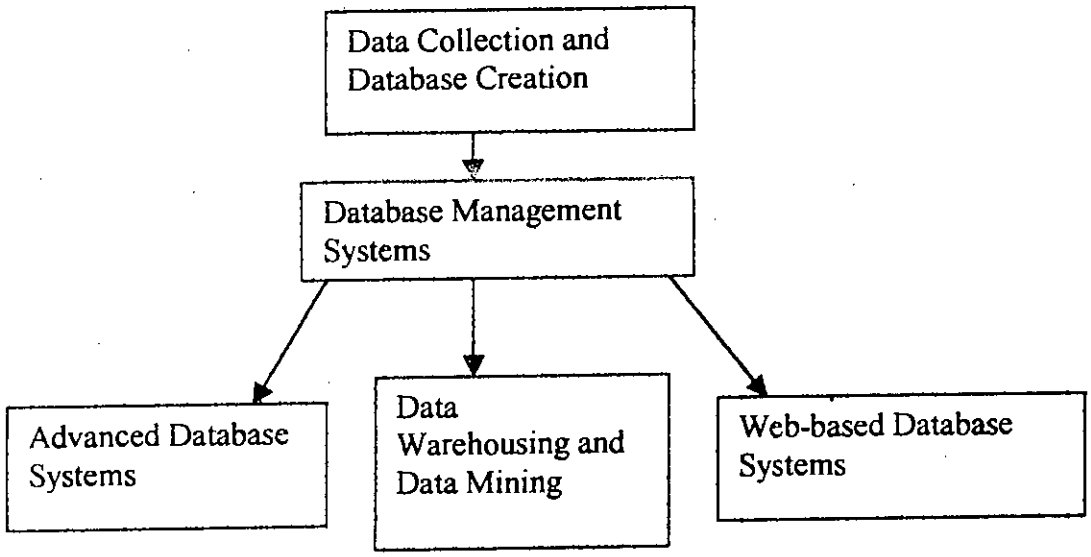


Fig. 2.1 The Evaluation of Database Technology.

In addition, consider current expert system technologies, which typically rely on users or domain experts to manually input knowledge into knowledge bases. Unfortunately, this procedure is prone to biases and errors, and it is extremely time-consuming and costly. Data mining tools perform data analysis and automates the generation of knowledge from knowledgebase for scientific and medical research. The widening gap between data and information calls for a systematic development of data mining tools that will turn data tombs into "golden nuggets" of knowledge.

2.3.1 Making The Most of Resources

Data warehouse, legacy databases, and corporate information systems are ideal information sources to be used for detecting patterns and trends through the application of data mining techniques [17]. Customer-training databases, corresponding systems, inventory control, and other very obvious and explicit data sets are good candidates for data mining applications.

If users are doing forensic accounting, they might be able to use phone-call data collected by the internal telephone switch at most corporations. Communication patterns may also be traced by accessing records of e-mail transactions maintained on users' network servers. If user's application question bears on issues concerning access to building facilities, consider that some security departments have badge reader data that record time and ID numbers of people entering and leaving certain

locations. User's personnel department maintains all of the home address and home data on nonexempt staff.

Expense reports, corporate credit card statements, travel advances, insurance claims, equipment maintenance reports, point-of-sale charges, warranty records, damaged goods receipts, shipping invoices, merchandise sales and returns, weather reports, and just about any other piece of information collected can be used in a data mining application.

In the event that necessary data are not available or do not yet exist, they can usually be generated. Information is reported at a level of detail above what is needed for the analysis. Consider, user had a medical application in which he was trying to trace the origins of a series of apparent adverse reactions to some combination of all medications. He might have coded the names of all medications being taken for each patient case in his study, yet still fail to find a definite link showing that a particular drug was consistently present in the adverse reaction cases. One approach might be to create a new data included in his original data set. By doing this he would add a level of detail to the analysis that might then permit him to discover that it is actually a combination of ingredients that produces the adverse reactions.

It is often found useful to generate new data sources to be used as supplemental information during an analysis. Simple lists may be added as extra data sources and insight to the data mining process. Since data mining is an iterative process, sources can be introduced at any time through the entire engagement. This occurs frequently because as user progress in identifying patterns, he realizes that there are certain dimensions that may be missing. The easiest way to resolve this is to create a new data set that contains the information needed.

2.4 Popularity of Data Mining

The popularity of data mining is increasing day by day. The volume of data is increasing rapidly. Human are facing various complexities to analyze the data. So, introduction of automated data mining techniques makes the task easier. The reasons for the growing popularity of data mining are outlined shortly in the following paragraphs.

2.4.1 Growing Data Volume

The main reason for necessity of automated computer systems for intelligent data analysis is the enormous volume of existing and newly appearing data that require processing. The amount of data accumulated each day by various business, scientific, and governmental organizations around the world is daunting. According to information from GTE research center [17], only scientific organizations store each day about 1 TB (terabyte!) of new information. And it is well known that academic world is by far not the leading supplier of new data. It becomes impossible for human analysts to cope with such overwhelming amounts of data.

2.4.2 Limitations of Human Analysis

There are two other problems that surface when human analysts process data. One is the inadequacy of the human brain when searching for complex multifactor dependencies in data. The other is the lack of objectiveness in such an analysis. A human expert is always a hostage of the previous experience of investigating other systems. Sometimes this helps, sometimes this hurts. However, it is almost impossible to get rid of this fact.

2.4.3 Low Cost of Machine Learning

One additional benefit of using automated data mining systems is that this process has a much lower cost than any trained (and paid) professional statisticians. While data mining does not eliminate human participation in solving the task completely, it significantly simplifies the job. It also allows an analyst who is not a professional in statistics and programming to manage the process of extracting knowledge from data.

2.5 The Steps of the Data Mining

Data mining is the process of Knowledge Discovery in Databases, or KDD. Knowledge discovery is a process that consists of an iterative sequence of the several steps. These steps (shown in Fig. 2.2) are pointed out in this section.

Data cleaning: To remove noise and inconsistent data.
Data integration: Where multiple data sources may be combined.

Data selection: Where data relevant to the analysis task are retrieved from the database

Data transformation: Where data are transformed or consolidated into forms appropriate for mining by performing summery or aggregation operations, for instance.

Data mining: An essential process where intelligent methods are applied in order to extract data pattern.

Pattern Evolution: To identify the truly interesting patterns representing knowledge based on some interesting measure.

Knowledge Presentation: Where visualization and knowledge representation techniques are used to present knowledge to the user.

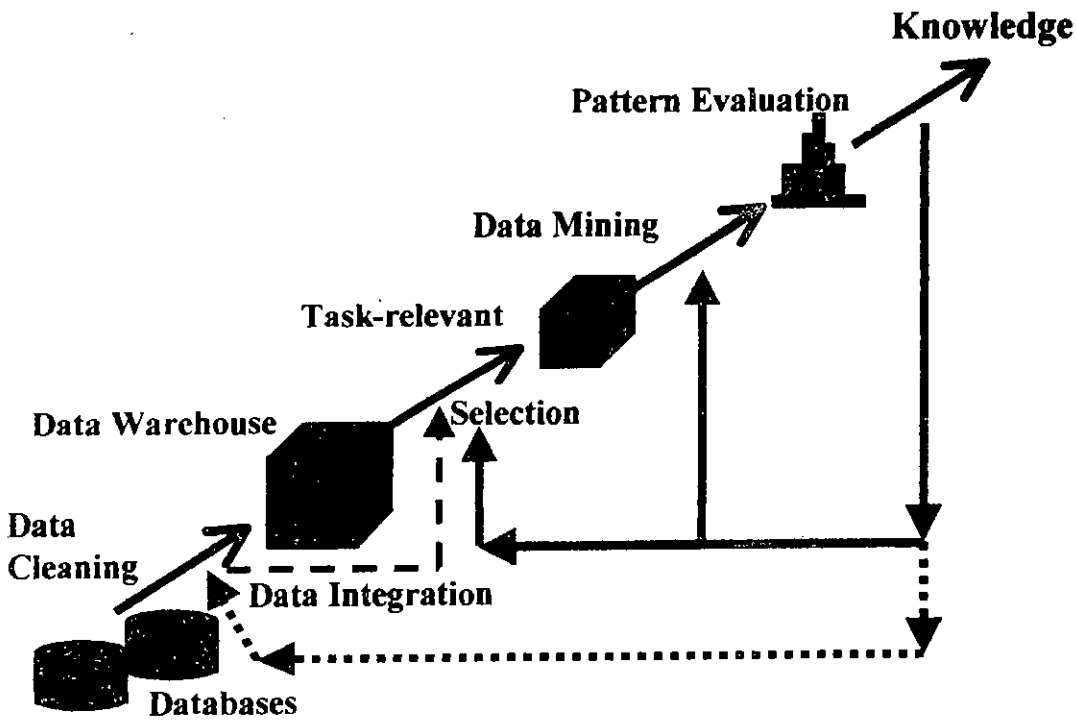


Fig. 2.2 Data Mining as a step in the process of Knowledge Discovery.

2.6 Architecture of Data Mining System

The architecture of a typical data mining system can be described by Fig. 2.3. A typical data mining system may consist of several components. These are database or data warehouse, database or data warehouse server, knowledgebase, data mining engine, pattern evaluation module and graphical user interface. The following subsection describes each component briefly.

Database or Data Warehouse: This is one or a set of databases, data warehouses, spreadsheets, or other kinds of information repositories. These are the repositories of information collected from multiple sources, stored under a unified schema, and which usually resides at a single site.

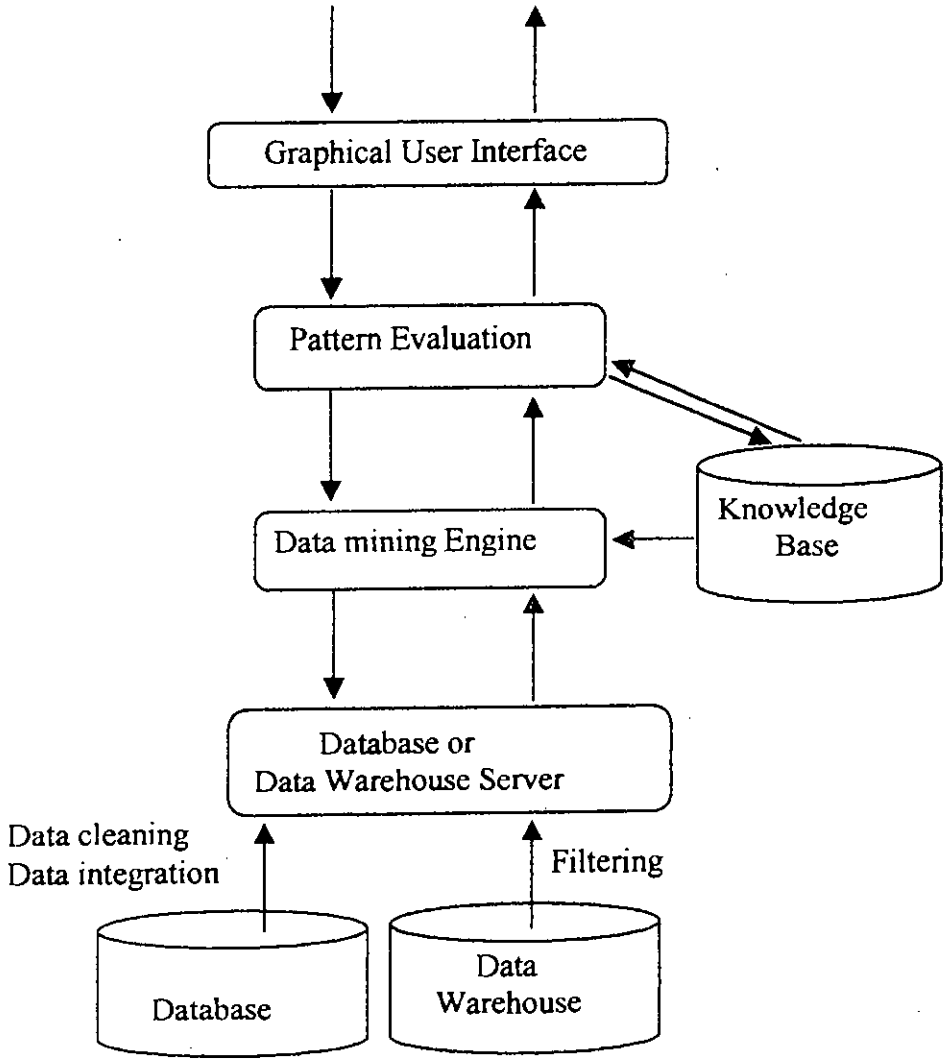


Fig. 2.3 Architecture of a typical data mining system

Database or Data Warehouse Server: The database or data warehouse server is responsible for fetching the relevant data, based on the user's data mining request. Its purpose is to serve data mining users providing necessary data from data repositories.

Knowledge Base: This is the domain knowledge that is used to guide the search, or evaluate the interestingness of resulting patterns. Such knowledge can include concept hierarchies, used to organize attributes or attribute values into different levels of abstraction. Knowledge such as user beliefs, which can be used to assess a pattern's interestingness based on its unexpectedness, may also be included. Other examples of domain knowledge are additional interestingness constraints or thresholds and metadata.

Data Mining Engine: This is an essential component to the data mining system and ideally consists of a set of functional modules for tasks such as characterization, association, classification, cluster analysis and evolution and deviation analysis.

Pattern Evaluation Module: This component typically employs interestingness measures and interacts with the data mining modules so as to focus the search towards interesting patterns. It may use interesting thresholds to filter out discovered patterns. Alternatively, the pattern evaluation module may be integrated with the mining module, depending on the implementation of the data mining method used. For efficient data mining, it is highly recommended to push the evaluation of pattern interestingness as deep as possible into the mining process so as to confine the search to only interesting patterns.

Graphical User Interface: This module communicates between users and the data mining system. It allows the user to interact with the system by specifying a data mining query or task, provides information to help focus the search, and performs exploratory data mining based on the intermediate data mining results. In addition, this component allows the user to browse database and data warehouse schemes or data structures, evaluate mined patterns, and visualize the patterns in different forms.

Chapter 3

Association Rules

- Introduction
- Rules
- Association Rules
- Relations between Data Mining and Association Rules
- Association Rule Mining

3.1 Introduction

This chapter discusses the basic ideas about rules and association rules. "How data mining and association rules are correlated with each other?"-is also elaborated in this chapter. Since this thesis work deals with mining interesting associations rules, so a detail concept about association rules are a vital point of this study.

Two salient features of association rules - support and confidence are elaborated shortly in this chapter. These two terms are used to evaluate interestingness of rules. Various ways of association rule mining are also described in this chapter. Moreover, how to find frequent itemsets from association rules using candidate sets and how to extract association rules having large confidence from these frequent itemsets are conversed in this chapter.

3.2 Rules

For any number of given attributes, if any number of criteria and the decision both are true simultaneously, then it is called rule. It means that for any implication ($p \rightarrow q$), if the antecedent p, and the conclusion q, both are true, then it is a rule. For example, consider the following statement.

" If n is divisible by 3, then square of n is divisible by 9".

Here, antecedent is 'n is divisible by 3' and conclusion is 'square of n is divisible by 9'. If antecedent is true then conclusion is also true. When both antecedent and conclusion are simultaneously true, then it is called a rule.

3.2.1 General Rules

General rules summarize the characteristics of each class and classify many instances accurately [7-8]. Usually, a user has no background knowledge of the domain represented by the target database, and then general rules are used [9]. General rules are not appropriate when domain experts use the system because they usually know it.

For an example, from mushroom database in UCI Repository [18], this database includes about 8,000 instances and each instance is classified into edible or

poisonous. This database is very easy as a benchmark of supervised learning algorithms because the following two rules can classify most instances correctly:

odor $\in$ {almond, anise, none} $\rightarrow$ edible: 98%
odor $\in$ {fishy, foul, spicy} $\rightarrow$ poisonous : 100%

3.2.2 Exceptional Rules

An exceptional rule is a specialization of general rule but concludes a different class with high accuracy [12, 19]. General rules cover relatively many instances and may be trivial for human experts, but exceptional rules may give new knowledge to the experts. Exceptional rules such as

Cap_color {brown, red} ^ stalk_root = bulbous $\Rightarrow$ edible: 100%,

where

Cap_color {brown, red} $\Rightarrow$ edible: 50%
Stalk_root = bulbous $\Rightarrow$ edible: 51%

Each of two conditions, cap_color ={brown, red} and stalk_root = bulbous, is not related to edible by itself but the conjunction of them concludes edible with probability 100%. This type of rule is not a straightforward conclusion from correlations between attributes and classes [14, 20]. It represents exceptions in the given database and may be interesting for human experts.

3.3 Association Rules

An association rule is the attribute-value conditions that occur frequently together in a given set of data [21]. More formally, association rules are of the form $X \Rightarrow Y$, that is, "A_1^.....^A_m $\Rightarrow$ B_1^.....^B_n", where A_i (for i = 1,.....,m) and B_j (for j = 1,.....n) are attribute-value pairs [22]. The association rule $X \Rightarrow Y$ is interpreted as "database tuples that satisfy the conditions in X are also likely to satisfy the condition in Y."

The goal of the association analysis is to detect relationships or associations between specific values of categorical variables in large data sets. This is a common task in many data mining projects as well as in the text mining, a subcategory of data

mining. These powerful exploratory techniques have a wide range of applications in many areas of business practice and also research - from the analysis of consumer preferences or human resource management, to the history of language. These techniques enable analysts and researchers to uncover hidden patterns in large data sets, such as "customers who order product *A* often also order product *B* or *C*" or "employees who said positive things about initiative *X* also frequently complain about issue *Y* but are happy with issue *Z*." The implementation of the so-called a-priori algorithm [6] allows you to process rapidly huge data sets for such associations, based on predefined "threshold" values for detection.

Association rules identify collections of data attributes that are statistically related with the underlying data. An association rule is of the form $X \Rightarrow Y$, where X and Y are disjoint conjunctions of attribute-value pairs [3]. The confidence of the rule is the conditional probability of Y given X, P (Y|X), and the support of the rule is the prior probability of X and Y, P (X and Y). Here probability is taken to be the observed frequency in the data set.

3.3.1 Support

Support is a salient feature of association rules. It is a probability that both of a body and a head are satisfied. Support is expressed as $(X \Rightarrow Y) = P (X \cup Y)$, where $P (X \cup Y)$ indicates that a transaction contains both X and Y, the union of item sets X and Y [10]. For example, if a customer goes to a market to buy a computer then there is a probability to buy only hardware or software or both. Here "support" is the probability of buying hardware and software together.

3.3.2 Confidence

Confidence is also a salient feature of association rules. Confidence is a conditional probability that if a head is satisfied then a body is also satisfied. Confidence is expressed as $(X \Rightarrow Y) = P (Y/X)$, where P (Y/X) indicates conditional probability that a transaction containing X also contains Y. Consider the previous example; here "Confidence" is the conditional probability of buying software with hardware.

Let $J = \{i \ldots m\}$ be a set of items. Let D, the task-relevant data, be a set of database transactions where each transaction T is a set of items such that $T \subseteq J$. Each transaction is associated with an identifier, called TID. Let A be a set of items. A

transaction T is said to contain A if and only if $A \subseteq T$. An association rule is an implication of the form $A \Rightarrow B$, where $A \subset J$, $B \subset J$, and $A \cap B = \varphi$. The rule $A \Rightarrow B$ holds in the transaction set D with support s, where s is the percentage of transactions in D that contain $A \cup B$ (i.e., both A and B). This is taken to be the probability, $P(A \cup B)$. The rule $A \Rightarrow B$ has confidence c in the transaction set D if c is the percentage of transactions in D containing A that also contain B. This is taken to be the conditional probability, P (B|A). That is,

$$\text{Support } (A \Rightarrow B) = P(A \cup B) \quad (3.1)$$

$$\text{Confidence } (A \Rightarrow B) = P(B/A). \quad (3.2)$$

Rules that satisfy both a minimum support threshold (min_sup) and a minimum confidence threshold (min_conf) are called strong. By convention, one will write support and confidence values so as to occur between 0% and 100%, rather than 0 to 1.0.

A set of items is referred to as an item set. An item set that contains k items is a k-item set. The set {computer, financial_management_software} is a 2-itemset. The occurrence frequency of an item set is the number of transactions that contain the item set. This is also known, simply, as the frequency, support count, or count of the item set. An item set satisfies minimum support if the occurrence frequency of the item set is greater than or equal to the product of min_sup and the total number of transactions in D. The number of transactions required for the item set to satisfy minimum support is therefore referred to as the minimum support count. If an item set satisfies minimum support, then it is a frequent item set. From these frequent item sets association rules are mined that fulfill the criteria of minimum confidence.

3.3.3 How Association Rules Work

The usefulness of this technique to address unique data mining problems is best illustrated in a simple example. Suppose you are collecting data at the checkout cash registers at a large bookstore. Each customer transaction is logged in a database, and consists of the titles of the books purchased by the respective customer, perhaps additional magazine titles and other gift items that were purchased, and so on. Hence, each record in the database will represent one customer (transaction), and may consist

of a single book purchased by that customer, or it may consist of many (perhaps hundreds of) different items that were purchased, arranged in an arbitrary order depending on the order in which the different items (books, magazines, and so on) came down the conveyor belt at the cash register.

The purpose of the association analysis is to find associations between the items that were purchased, i.e., to derive association rules that identify the items and co-occurrences of different items that appear with the greatest (co-) frequencies. For example, you want to learn which books are likely to be purchased by a customer who you know already purchased (or is about to purchase) a particular book. This type of information could then quickly be used to suggest to the customer those additional titles. You may already be "familiar" with the results of these types of analyses, if you are a customer of various on-line (Web-based) retail businesses; many times when making a purchase on-line, the vendor will suggest similar items (to the ones purchased by you) at the time of "check-out", based on some rules such as "customers who buy book title *A* are also likely to purchase book title *B*," and so on.

3.3.4 Unique Data Analysis Requirements

Crosstabulation tables in particular multiple response tables can be used to analyze unique data. However, when the number of different items in the data is very large and not known ahead of time, and when the "factorial degree" of important association rules is not known ahead of time, then these tabulation facilities may be very cumbersome to use, or simply not applicable. Consider once more the simple "bookstore-example" discussed earlier. First, the number of book titles is practically unlimited. In other words, if a table is made where each book title represents one dimension, and the purchase of that book (yes/no) is the classes or categories for each dimension, then the complete crosstabulation table is huge and sparse (consisting mostly of empty cells).

Alternatively, all possible two-way tables could be constructed from all items available in the store; this would allow detecting two-way associations (association rules) between items. However, the number of tables that would have to be constructed would again be huge, most of the two-way tables would be sparse, and worse, if there were any three-way association rules "hiding" in the data. The a-priori algorithm implemented in *Association Rules* will not only automatically detect the

relationships ("cross-tabulation tables") that are important (i.e., cross-tabulation tables that are not sparse, not containing mostly zero's), but also determine the factorial degree of the tables that contain the important association rules.

In summary, *Association Rules* allow finding rules of the kind *If X then (likely) Y*, where *X* and *Y* can be single values, items, words, etc., or conjunctions of values, items, words, etc. (e.g., *if (Car=Porsche and Gender=Male and Age<20) then (Risk=High and Insurance=High)*). The program can be used to analyze simple categorical variables, dichotomous variables, and/or multiple response variables. The algorithm will determine association rules without requiring the user to specify the number of distinct categories present in the data, or any prior knowledge regarding the maximum factorial degree or complexity of the important associations. In a sense, the algorithm will construct cross-tabulation tables without the need to specify the number of dimensions for the tables, or the number of categories for each dimension. Thus, this technique is particularly well suited for data and text mining of huge databases.

3.3.5 Association Rules in Market Basket Analysis

Market basket analysis deals with trends of customers demand, availability of those demanding items, buying and selling trends etc to have a well idea about marketing strategies, profit and loss of the business, such and such. Suppose, as manager of an All Electronics branch, you would like to learn more about the buying habits of your customers. Specifically, you wonder, "Which groups or sets of items are customers likely to purchase on a given trip to the store?" To answer your question, market basket analysis may be performed on the retail data of customer transactions at your store. The results may be used to plan marketing or advertising strategies, as well as catalog design. For instance, market basket analysis may help managers design different store layouts. In one strategy, Mining items that are frequently purchased together can be placed in close proximity in order to further encourage the sale of such items together. If customers who purchase computers also tend to buy financial management software at the same time, then placing the hardware display close to the software display may help to increase the sales of both items.

In an alternative strategy, placing hardware and software at opposite ends of the store may entice customers who purchase such items to pick up other items along

the way. For instance, after deciding on an expensive computer, a customer may observe security systems for sale while heading towards the software display to purchase financial management software and may decide to purchase a home security system as well. Market basket analysis can also help retailers to plan which items to put on sale at reduced prices. If customers tend to purchase computers and printers together, then having a sale on printers may encourage the sale of printers as well as computers.

If one thinks of the universe as the set of items available at the store, then each item has a Boolean variable representing the presence or absence of that item. A Boolean vector of values assigned to these variables can then represent each basket. The Boolean vectors can be analyzed for buying patterns that reflect items that are frequently associated or purchased together. These patterns can be represented in the form of association rules. For example, the information that customers who purchase computers also tend to buy financial management software at the same time is represented in association rule below:

$$\text{computer} \Rightarrow \text{financial\_management\_software} \quad [\text{support} = 2\%, \text{confidence} = 60\%] \qquad (3.3)$$

Support and confidence are two measures of rules' interestingness that were described earlier in Section 3.1. They respectively reflect the usefulness and certainty of discovered rules. A support of 2% for association rules means that 2% of all the transactions under analysis show that computer and financial management software are purchased together. A confidence of 60% means that 60% of the customers who purchased a computer also bought the software. Typically, association rules are considered interesting if they satisfy both a minimum support threshold and a minimum confidence threshold. Users or domain experts can set such thresholds.

3.3.6 Categories of Association Rules

Different types of association rules introduce different types of association rule mining. Association rules can be classified in various ways, based on the following criteria:

Based on the types of values handled in the rule: If a rule concerns associations between the presence or absence of items, it is a Boolean association rule. For example, Rule (3.3) is a Boolean association rule obtained from market basket analysis.

If a rule describes associations between quantitative items or attributes, then it is a quantitative association rule. In these rules, quantitative values for items or attributes are partitioned into intervals. The following rule is an example of a quantitative association rule, where X is a variable representing a customer:

$$age(X, \text{“30...39”}) \wedge income(X, \text{“42K...48K”}) \Rightarrow buys(X, \text{high resolution TV}) \quad (3.4)$$

Notc that the quantitative attributes, age and income, have been discretized.

Based on the dimensions of data involved in the rule: If the items or at tributes in an association rule reference only one dimension, then it is a single-dimensional association rule. Note that Rule (3.3) could be rewritten as

$$buys(X, \text{“computer”}) \Rightarrow buys(X, \text{“financial\_management\_software”}) \quad (3.5)$$

Rule (3.3) is a single-dimensional association rule since it refers to only one dimension, buys. If a rule references two or more dimensions, such as the dimensions buys, time_of_transaction, and customer_category, then it is a multidimensional association rule. Rule (3.4) is considered a multidimensional association rule since it involves three dimensions: age, income, and buys.

Based on the levels of abstractions involved in the rule set: Some methods for association rule mining can find rules at differing levels of abstraction. For example, suppose that a set of association rules mined includes the following rules:

$$age(X, \text{“30...39”}) \Rightarrow buys(X, \text{“laptop computer”}) \quad (3.6)$$

$$age(X, \text{“30...39”}) \Rightarrow buys(X, \text{“computer”}) \quad (3.7)$$

In Rules (3.6) and (3.7), the items bought are referenced at different levels of abstraction. (e.g., "computer" is a higher-level abstraction of "laptop computer".) one will refer to the rule set mined as consisting of multilevel association rules. If, instead, the rules within a given set do not reference items or attributes at different levels of abstraction, then the set contains single-level association rules.

Based on various extensions to association mining: Association mining can be extended to correlation analysis, where the absence or presence of correlated items can be identified. It can also be extended to mining maxpatterns (i.e., maximal frequent patterns) and frequent closed itemsets. A maxpattern is a frequent pattern, p, such that any proper superpattern of p is not frequent. A frequent closed itemset is a frequent closed itemset where an itemset c is closed if there exists no proper superset of c, c', such that every transaction containing c also contains c'. Maxpatterns and frequent closed itemsets can be used to substantially reduce the number of frequent itemsets generated in mining.

3.4 Relations between Data Mining and Association Rules

Data Mining is regarded as one of the existing steps in Knowledge Discovery in Databases (KDD) Process, which is the "non-trivial process of extracting patterns from data that are useful, novel and comprehensive" [23]. This process treats all parts in whole analysis task, from understanding goals and collecting data to result verification and validation. Basically, the KDD process can be viewed in three parts: 1) goal definition and data preparing, where existing data, related to the goal is collected, cleaned and enriched as much as possible; 2) Analysis, where data is transformed and applied to one or more data mining algorithms; 3) Results interpretation, validation and deployment, where useful, novelty process output is used as a benefit to the organization.

Although data mining requires high computational efforts, the preceding steps of preparation for data mining accounts for between 75% to 85% of the overall process effort [3-4]. On the other hand, the data-mining step is the part of the process where knowledge is extracted. Since this task is not human-performable, due to the quantity and complexity involved, special attention should be taken in mining algorithm choice and parameterization.

There exist many distinct Data Mining techniques, each one with many algorithms. These techniques include different class problems such as: a) Association Rules, where events are verified in order to determine if their behaviour is linked (in business point of view); b) Classification, where events in business are labeled within some classes and then a description is searched for them, so new events can be pre-classified; c) Sequences, where events are studied to check if their occurrences are subsequents, or if there exists sequence patterns in business events. These class problems incorporatc most of the interesting techniques from the business point of view, except for the lack of clustering techniques. In Clustering, elements or events are analyzed to check if they behave similarly, or if unlabeled group patterns exist. In [7] techniques are further investigated and are called *Data Mining Tasks* and include new techniques, such as clustering, summarization and deviation detection.

The Association Rules' most commonly used example is the relationship between products in Market Baskets [3]. It is stated in an antecedent and consequent set: A $\Rightarrow$ B, where A is an antecedent and B a consequent; or A, B $\Rightarrow$C, where there are two antecedents and one consequence [7] that states association rules as "an interesting combinatorial problem", but it is very hard to predetermine if combinations occur and to what level, two products, three products, etc. These peculiar features allicd to its potential result, pushed attention to association rules. Although easily stated, Association Rules demand some choices to be made. The first is the parameters threshold. In these rules there are originally two parameters to be adjusted: Support and Confidence. The first one determines how often rules must occur within all existing events, which gives an overall rule significance. Confidence is used to verify the internal strength of the rule. This means that an element's occurrence must not be much higher from others that appear in the same rule, e.g. if one element appears in all transactions then it will be associated to all products and will appear in all rules, and this might not interesting in the business point of view. Adjusting parameters is not an easy task, since their values work as a pruning measure: the higher the support and confidence the fewer the rules generated, but there are no perfect/predetermined values. Choosing parameters is an empirical procedure, and based on first extraction of rules, they can be readjusted.

Another choice to be made is the use of quantitative continuous values in contrast with qualitative discrete/boolean ones. It is different to say that elements A and B are associated (e.g., A$\Rightarrow$B) than to say that A with amount of 30 and B with

amount of 50 are associated (A (50) $\Rightarrow$ B (30)). In the second case, an event with A and 20 altogether with B and 10 would not fit the rule and, therefore, would be discarded. Treating quantitative values has been addressed in [8].

The last choice to be made is the number of consequences and antecedents. This alters the mining procedure because having found a pair of elements (antecedent and consequent), it must be decided if the process continues to find more antecedents or consequences still satisfying the parameters' threshold. As the events are not necessarily linked, rules are not transitive, e.g. A $\Rightarrow$ B and B $\Rightarrow$ C do not necessarily mean A $\Rightarrow$ C.

3.5 Association Rule Mining

Association rule mining is the process of extracting interesting rules from a large set of association rules. The traditional association rule-mining problem can be described as follows.

Say a database of transactions, a minimal confidence threshold and a minimal support threshold are given, user have to find all association rules whose confidence and support are above the corresponding thresholds [3].

Association rule mining searches for interesting relationships among items in a given data set [24]. The following subsection briefly describes some methods for mining association rule. Association rule mining is a two-step process:

** Find all frequent item sets: By definition, each of these item sets will occur at least as frequently as a pre-determined minimum support count.

** Generate strong association rules from the frequent item sets: By definition, these rules must satisfy minimum support and minimum confidence.

Additional interestingness measures can be applied, if desired. The second step is the easier of the two. The overall performance of mining association rules is determined by the first step.

3.5.1 Finding Frequent Itemsets from Association Rules Using Candidate Generation

The process of finding frequent itemsets from association rules using candidate generation can be described using the following steps:

i) The join step: To find L_k, joining L_{k-1} with itself generates a set of candidate k-itemsets. This set of candidates is denoted C_k. Let l_1 and l_2 be itemsets in L_{k-1}. The notation $l_i[j]$ refers to the jth item in li (e.g., l_1 [k- 2] refers to the second to the last item in l_1). The join, $L_{k-1} \infty L_{k-1}$ is performed, where members of Lk-1 are joinable if their first (k- 2) items are in common. That is, members l_1 and l_2 of L_{k-1} are joined if $(l_1[1] = l_2[1]) \wedge (l_1[2] = l_2[2]) \wedge ... \wedge (l_1[k-2] = l_2[k-2]) \wedge (l_1[k-1] < l_2[k-1])$. The condition $l_1[k-11 < l_2[k- 1]$ simply ensures that no duplicates are generated. The resulting item set formed by joining l_1 and l_2 is $l_1[1]l_1[2] \ldots l_1[k-1]l_2[k-1]$.

ii) The prune step: C_k is a superset of L_k, that is, its members may or may not be frequent, but all of the frequent k-item sets are included in C_k. A scan of the database to determine the count of each candidate in C_k would result in the determination of L_k (i.e., all candidates having a count no less than the minimum support count are frequent by definition, and therefore belong to L_k). C_k, however, can be huge, and so this could involve heavy computation. To reduce the size of C_k, the Apriori property is used as follows. Any (k-1)-item set that is not frequent cannot be a subset of a frequent k-item set. Hence, if any (k-1) subset of a candidate k-item set is not in L_{k-1}, then the candidate cannot be frequent either and so can be removed from C_k. This subset testing can be done quickly by maintaining a hash tree of all frequent item sets.

Let's look at a concrete example of Apriori, based on the AllElectronics transaction database, D, of Table 3.1. There are nine transactions in this database, that is, ID = 9. Fig. 3.1 is used to illustrate the algorithm for finding frequent itemsets in D.

Table 3.1: Transactional data

TID	List of item IDs
T_{100}	I_1, I_2, I_3
T_{200}	I_2, I_4
T_{300}	I_2, I_3
T_{400}	I_1, I_2, I_4
T_{500}	I_1, I_3
T_{600}	I_2, I_3
T_{700}	I_1, I_3
T_{800}	I_1, I_2, I_3, I_5
T_{900}	I_1, I_2, I_3

1. In the first iteration of the algorithm, each item is a member of the set of candidate 1-itemsets, C. The algorithm simply scans all of the transactions in order to count the number of occurrences of each item.

2. Suppose that the minimum transaction support count required is 2 (i.e., min_sup = 2/9 = 22%). The set of frequent 1-itemsets, L_1 can then be determined. It consists of the candidate 1-itemsets satisfying minimum support.

3. To discover the set of frequent 2-itemsets, L_2, the algorithm uses $L_1 \infty L_2$ to generate a candidate set of 2-itemsets; C_2 .C_2 consists of $\binom{|L_1|}{2}$ 2- itemsets.

4. Next, the transactions in Dare scanned and the support count of each candidate itemset in C_2 is accumulated as shown in the middle table in the second row of Fig. 3.1.

5. The set of frequent 2-itemsets, L_2, is then determined, consisting of those candidate 2-itemsets in C_2 having minimum support.

6. The generation of the set of candidate 3-itemsets, C_3 is shown in the following sub steps. First, let $C_3 = L_2 \infty L_2 = \{\{I_1,I_2,I_3\}, \{I_1,I_2,I_5\}, \{I_1,I_3,I_5\}, \{I_2,I_3,I_4\}, \{I_2,I_3,I_5\}, \{I_2,I_4,I_5\}\}$. Based on the Apriori property that all subsets of a frequent item set must also be frequent, one can determine that the four latter candidates cannot possibly be frequent. Note that when given a candidate k-item set, one only need to check if its (k-1)-subsets are frequent since the Apriori algorithm uses a level-wise search strategy.

i) Join: $C_3 = L_3 \infty L_3 = \{\{I_1,I_2\}, \{\{I_1,I_3\}, \{\{I_1,I_5\}, \{I_2,I_3\}, \{I_2,I_4\}, \{I_2,I_5\}\} \infty \{\{I_1, I_2\},\{\{I_1,I_3\}, \{\{I_1,I_5\},\{ I_2,I_3\}, \{I_2,I_4\}, \{I_2,I_5\}\} = \{\{I_1,I_2,I_3\}, \{I_1,I_2,I_5\}, \{I_1,I_3,I_5\}, \{I_2,I_3,I_4\},\{I_2,I_3,I_4\}, \{I_2,I_4,I_5\}\}$.

ii) Prune using the Apriori property: All nonempty subsets of a frequent itemset must also be frequent. Do any of the candidates have a subset that is not frequent?

- The 2-item subsets of $\{I_1,I_2,I_3\}$ are $\{I_1,I_2\}$, $\{I_1,I_3\}$, and $\{I_2,I_3\}$. All 2-item subsets of $\{I_1,I_2,I_3\}$ are members of L_2. Therefore, keep $\{I_1,I_2,I_3\}$ in C_3.

- The 2-item subsets of $\{I_1,I_2,I_5\}$ are $\{I_1,I_2\}$, $\{I_1,I_5\}$, and $\{I_2,I_5\}$. All 2-item subsets of $\{I_1,I_2,I_5\}$ are members of L_2. Therefore, keep $\{I_1,I_2,I_5\}$ in C_3.

- The 2-item subsets of $\{I_1,I_3,I_5\}$ are $\{I_1,I_3\}$, $\{I_1,I_5\}$, and $\{I_3,I_5\}$. $\{I_3,I_5\}$ is not a member of L_2 and so it is not frequent. Therefore, remove $\{I_1,I_3,I_5\}$ from C_3.
- The 2-item subsets of $\{I_2,I_3,I_4\}$ are $\{I_2,I_3\}$, $\{I_2,I_4\}$, and $\{I_3,I_4\}$. $\{I_3,I_4\}$ is not a member of L_2 and so it is not frequent. Therefore, remove $\{I_2,I_3,I_4\}$ from C_3.
- The 2-item subsets of $\{I_2,I_3,I_5\}$ are $\{I_2,I_3\}$, $\{I_2,I_5\}$, and $\{I_3,I_5\}$. $\{I_3,I_5\}$ is not a member of L_2 and so it is not frequent. Therefore, remove $\{I_2,I_3,I_5\}$ from C_3.

- The 2-item subsets of $\{I_2,I_4,I_5\}$ are $\{I_2,I_4\}$, $\{I_2,I_5\}$ and $\{I_4,I_5\}$. $\{I_4,I_5\}$ is not a member of L_2 and so it is not frequent. Therefore, remove $\{I_2,I_4,I_5\}$ from C_3.

Therefore, $C_3 = \{\{I_1,I_2,I_3\}, \{I_1,I_2,I_5\}\}$ after pruning.

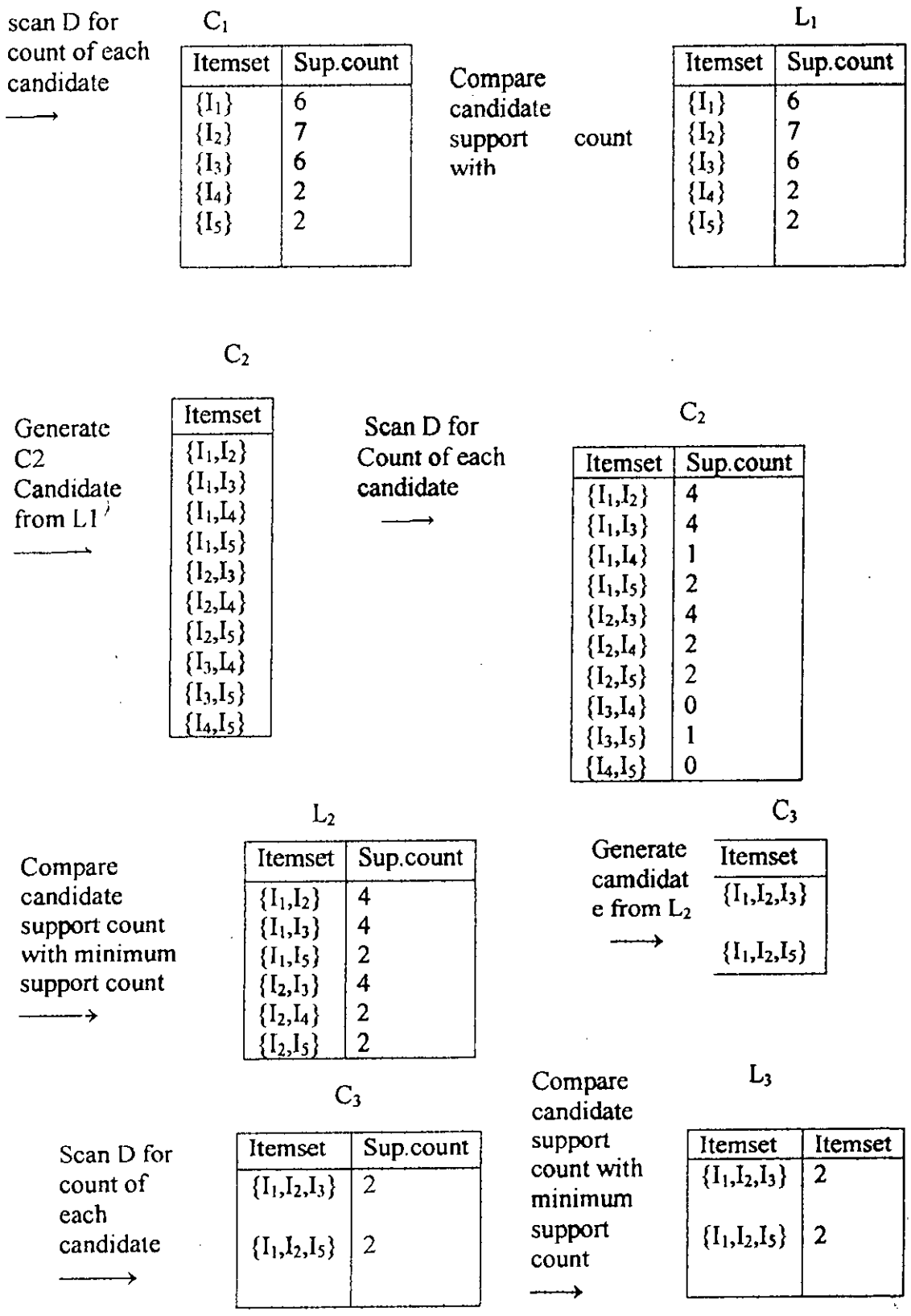


Fig 3.1 Generation of candidate item sets frequent item sets, where the minimum support count is 2

7. The transactions in D are scanned in order to determine L_3, consisting of those candidate 3-itemsets in C3 having minimum support.

8. The algorithm uses $L_3 \infty L_3$ to generate a candidate set of 4-itemsets, C_4. Although the join results in $\{\{I_1,I_2,I_3,I_5\}\}$, this item set is pruned since its subset $\{\{I_2,I_3,I_5\}\}$ is not frequent. Thus, $C_4 = \Phi$, and the algorithm terminates, having found all of the frequent item sets.

The Apriori algorithm [12] generates the candidates and then uses the Apriori property to eliminate those having a subset that is not frequent. Once all the candidates have been generated, the database is scanned. For each transaction, a subset function is used to find all subsets of the transaction that are candidates and the count for each of these candidates is accumulated. Finally, all those candidates satisfying minimum support form the set of frequent item sets. A procedure can then be called to generate association rules from the frequent item sets.

3.5.2 Generating Association Rules from Frequent Item Sets

Once the frequent itemsets from transactions in a database have been found, it is straightforward to generate strong association rules from them (where strong association rules satisfy both minimum support and minimum confidence). This can be done using the following equation for confidence, where the conditional probability is expressed in terms of itemset support count.

$$\text{confidence}(A \Rightarrow B) = P(B|A) = \text{support}(A \cup B)/\text{support\_count}(A), \quad (3.8)$$

where support_count ($A \cup B$) is the number of transactions containing the itemsets $A \cup B$, and support_count (A) is the number of transactions containing the itemset A. Based on this equation, association rules can be generated as follows.

- For each frequent itemset l, generate all nonempty subsets of l.
- For every nonempty subsets of l, output the rule " s = (1- s)" if support_count(l)/support_count(s) ≥ minconf, where min_conf is the minimum confidence threshold.

Since the rules are generated from frequent itemsets, each one automatically satisfies minimum support. Frequent itemsets can be stored ahead of time in hash tables along with their counts so that they can be accessed quickly.

Let's try an example based on the transactional data for AllElectronics. Suppose the data contain the frequent itemset $l = \{I_1, I_2, I_5\}$. What are the association rules that can be generated from l? The nonempty subsets of l are $\{I_1, I_2\}$, $\{I_1, I_5\}$, $\{I_2, I_5\}$, $\{I_1\}$, $\{I_2\}$, and $\{I_5\}$. The resulting association rules are as shown below, each listed with its confidence:

$I_1 \wedge I_2 \Rightarrow I_5$, confidence= 2/4 = 50%

$I_1 \wedge I_5 \Rightarrow I_2$, confidence= 2/2 = 100%

$I_2 \wedge I_5 \Rightarrow I_1$, confidence= 2/2 = 100%

$I_1 \Rightarrow I_2 \wedge I_5$, confidence= 2/6 = 33%

$I_2 \Rightarrow I_1 \wedge I_5$, confidence= 2/7 = 29%

$I_5 \Rightarrow I_1 \wedge I_2$, confidence= 2/2 = 100%

The minimum confidence threshold is, say 70%, then only the second, third, and last rules above are output, since these are the only ones generated that are strong.

Chapter 4

Constructive Approach to Mine Interesting Rules

- Introduction
- Probability Theorems
- Classification Rules
- Interestingness without Grouping Attribute Values
- Interestingness with Grouping Attribute Values
- Constructive Approach to Mine Interesting Rules (CAMIR)

4.1 Introduction

This chapter deals with the proposed constructive approach to mine interesting rules (CAMIR). There are some necessary terms that are required for describing CAMIR. These are probability theorems, classification rules, and interestingness of rules with and without grouping attribute values. These terms are discussed in brief at first in this chapter. Then the detail description of CAMIR with its flow chart and pseudo code is presented.

4.2 Probability Theorems

The objectives of this section is to define probability and the symbols used in probability theory, to define the seven fundamental laws and know how to use them in solving problems in probability and understand the three counting rules and know how to use them to calculate amounts, sets, and subsets of objects.

4.2.1 Definitions and Symbols

Probability can be defined as the chance that something will happen. Other terms (along with chance) that can be considered synonyms to probability are likelihood and tendency. Likelihood can be substituted for the word chance in the above sentence, but tendency is used in a slightly different context. "The tendency to be the same" or "the tendency to be different" is both probability measures.

Probability is measured in terms of a ratio; that is, one value divided into part of that value. Therefore, probability values range between 0.00 and 1.00. The total of all possible chances in a situation is equal to 1.00. Probability is also very frequently measured in percent, but percent is just the ratio multiplied by 100 (a ratio of 0.015 is equal to 1.5%). The basic rules of probability apply to all probability and statistical distributions, even the normal. However, the basic probability laws apply more obviously to the discrete situation than to the continuous and the basic discrete probability distribution are more easily derived from the basic probability rules.

There are only three types of symbols needed in probability theory. *P(A)* or P_A are the symbols used for the probability of an event. The lower case *s* is used to denote success, a desirable event. The lower case *n* is used to denote the total number

of possibilities. This is identical in meaning to the sample size n used in the normal curve. The probability equation, then, is

$$P(A) = P_A = s/n \tag{4.1}$$

where $P_A = P(A)$ = the probability of an event.

s = the number of successes (or failures, depending on what is being evaluated).

n = the total possible number of cases.

RULE 1. The probability of an event lies between 0 and 1. Zero probability (0.00) is the certainty that event A will not occur. One (1.00) is the certainty that event A will occur. The formula is

$$P(A) = P_A = 0.00 \text{ to } 1.00 \tag{4.2}$$

RULE 2. *The sum of the probabilities* of a situation is equal to 1.00. The formula is

$$P_1 + P_2 + P_3 + \ldots + P_n = 1.00 \tag{4.3}$$

RULE 3. The *complementary law* states that if P_A is the probability that an event will occur, then $1 - P_A$ is the probability that the event will not occur. The probability of it occurring plus the probability of it not occurring always equals 1.00.

RULE 4. In the *additive law* of probability, the probability of either A or B is the sum of the probability of A and the probability of B. In this law the two events must be *mutually exclusive*, that is, the occurrence of one event makes the other impossible (cannot have both a head and a tail on one toss of a coin). The equation is

$$P(A \text{ or } B) = P_A + P_B \tag{4.4}$$

The "or" is very distinctive in this case and always means plus, or add (except in the case of the combination law where it can also mean minus). Where the additive law is applicable, the "or" will always occur in the statement of the problem (or be implied in the logic).

RULE 5. The *multiplicative law* of probability states that the mutual probability of two independent events is equal to the product of the probabilities of each event. If two events are independent, the occurrence of one does not affect the probability of the other occurring (the occurrence of a head on the toss of one coin has no effect on the probability of a head or a tail occurring on a toss of a second coin). The equation is

$$P\ (A\ and\ B) = P\ (A) \times P\ (B) \tag{4.5}$$

The "and" in this case always refers to multiplying, never to addition or subtraction. Where the multiplicative law is applicable, the "and" will always occur in the statement of the problem (or be implied in the logic).

RULE 6. The *combination law* is used to find the probability of occurrence of either one or both of two events. The equation is

$$P\ (A\ or\ B\ or\ both) = P\ (A) + P\ (B) - [P\ (A) \times P\ (B)] \tag{4.6}$$

Note that both the additive and then the multiplicative laws are combined in this rule. This formula can be generalized to three or more events. However, when three or more events are used, the term subtracted must include all events in the product. If it does not, the multiplicative law cannot be used in this subtracted term and the amount subtracted must be determined by logic alone.

RULE 7. The *conditional law* of probability applies to dependent events. Two events are dependent if the occurrences of one affect the probability of the second

occurring – it does not necessarily affect the occurrence of the second event, just the probability that it will occur. The equation is

$$P\ (A\ and\ B) = P\ (A) \times P\ (B|A) \tag{4.7}$$

P (B|A) means the probability of *B* occurring given that *A* has already occurred (used only in the special case where the previous occurrence of *A* affects *B's* Probability). *B|A* is usually referred to as *B* given *A*.

4.2.2 Counting Rules

It is frequently important in probability to determine the number of sets and subsets of objects. There are three main methods to do this, each having application to particular types of sets.

4.2.2.1 Simple Multiplication

When it is desired to know the total number of possible sets, the rule of simple multiplication usually applies. If event *A* can happen in any of n_1 ways and event *B* can happen in any of n_2 ways, the total number of ways that both can occur is n_1 times n_2.

4.2.2.2 Permutations

If the possible sets of objects are to be ordered (arranged in specific ways), then these sets are called permutations. A permutation is defined as an ordered arrangement of *n* objects taken *i* at a time. Suppose that three letters of the alphabet (A, C, and T) are chosen, and it is desired to arrange them in definite order. In other words, the order of appearance of the letters is important (this is the case, for instance, when letters of the alphabet are arranged to form words). Permutations of these three letters, then, are ACT, ATC, TAC, TCA, CAT, and CTA. There are six possible permutations of these three letters (even though only two make recognizable words). The equation is

$$P_i^n = \frac{n!}{(n-i)!} \tag{4.8}$$

4.2.2.3 Combinations

If the way the objects are ordered is unimportant, the set is called a combination. Using the same three letters of the alphabet used in the section on permutations (A, C, and T), there is only one combination. Since the same three letters are used, and the order of arrangement does not matter, then (as far as combinations go) ACT = TAC = CAT, etc. There are always more permutations than combinations in the same set of objects. (The letters A, C, and T have six permutations, but only one combination, except for the null set of n things taken 0 at a time when the permutations = the combinations = 1.) The equation for a combination is

$$C_i^n = \frac{n!}{i!(n-i)!} \tag{4.9}$$

4.2.3 Bayes' Rules

Recall the two forms of the rule:

$$P\ (A\ \wedge\ B\ )\ =\ P\ (\ A\ /\ B\ )\ P\ (\ B\ ))\tag{4.10}$$

$$P\ (A\ \wedge\ B\ )\ =\ P\ (\ B\ /\ A\ )\ P\ (\ A\ ))\tag{4.11}$$

Equating the two right-hand sides and dividing by P (A)

$$P\ (B/A)\ =\ P\ (A/B)\ \ P\ (B)/P\ \ (A)\tag{4.12}$$

This equation is known as Bayes' rule. This simple equation underlies all modern AI systems for probabilistic reference. The more general case of multi-valued variables can be written as follows:

$$P\ (Y/X) = P\ (X/Y)\ P\ (Y)/P\ (X)\tag{4.13}$$

where again this is to be taken as representing a set of equations relating corresponding elements of the table.

4.3 Classification Rules

A classification rules is an if-then rule whose head (conclusion) is a class label and whose body is a conjunction of conditions of attribute values. In this thesis following type of classification rules is used:

$$Rl: a_{i_1} \in D_{i_1} \wedge \ldots \wedge a_{i_L} \in D_{i_L} \rightarrow c \tag{4.14}$$

where c is a certain class and D_{ik} is a subset of possible values of attribute a_{ik}. Grouping attribute values and allowing all values in one-group increases the number of possible rules and degrade efficiency of discovery systems [4]. However, it is quite useful to improve readability of extracted rules because one rule with value grouping represents plural rules without grouping.

Classification and prediction are two forms of data analysis that can be used to extract models describing important data classes or to predict future data trends. Whereas classification predicts categorical labels, prediction models continuous-valued functions. For example, a classification model may be built to categorize blank loan applications as either safe or risky, while a prediction model may be built to predict the expenditures of potential customers on computer equipment given their income and occupation. Researchers in machine learning, expert systems, statistics, and neurobiology have proposed many classification and prediction methods. Most algorithms are memory resident, typically as summing a small data size. Recent database mining research has built on such work, developing scalable classification and prediction techniques capable of handling large disk-resident data. These techniques often consider parallels and distributed processing.

The goal of a classifier is to learn a body of knowledge M from an input dataset I. The derived knowledge M can then be used to predict (i.e., classify) new tuples. Let us consider a set of n independent variables $X_1;\ldots\ldots;X_n$ such that each X_k takes on values from a domain Dx_k . In addition, there is another variable Xc, called the class variable, whose domain is $D_C = \{c_1;\ldots\ldots; c_m\}$, with m being the number of classes.

The task of a classifier is:

(i) Given a training dataset I consisting of a set of (n+1)-tuples: (t_1; : : : ; t_n; c), where $t_k \in Dx_k$ (k = 1; : : : ; n) and c $\in$ D_C;

(ii) Construct a mapping M: (Dx_1 ;.....; Dx_n) $\rightarrow$ D_C.

M can now be used to predict the class of new tuples. Thus, given a n-tuple t′ = (t′1; t′2; : : : ; t′n), such that t′i $\in$ Dx_i ; i = 1; : : : ; n, the predicted class c′ will be: c′ =M(t′).

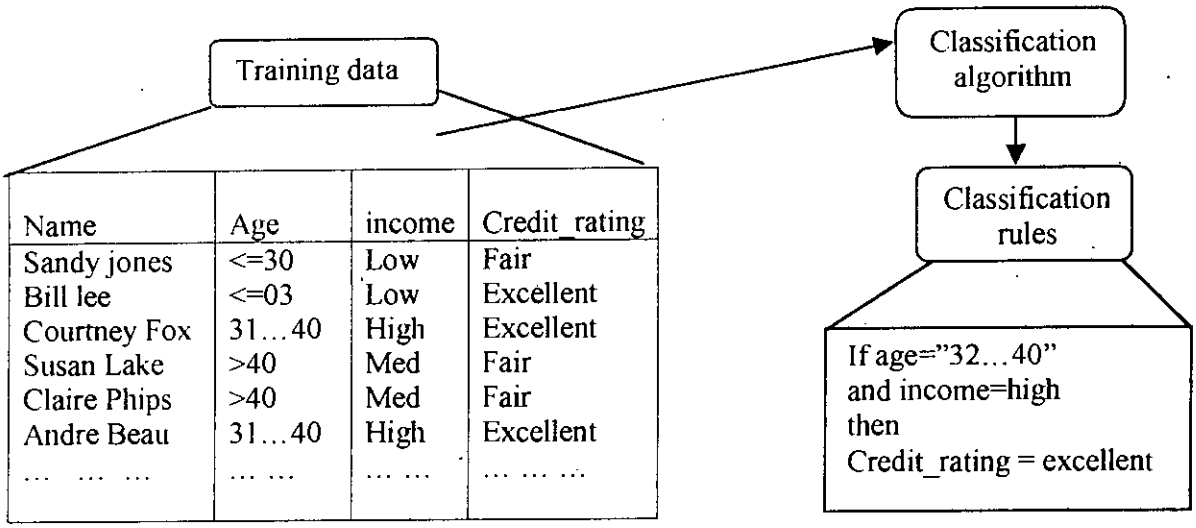
Let "X = v" be a term where X is an independent variable and v one of its values, v $\in$ D_X. A term "X = v" covers a tuple t when the attribute X in t has value v (when discrete variable is considered). Let a pattern be a conjunction of terms. A pattern ρ covers a tuple t when all the terms in ρ cover t. A rule r is a statement of the form: "if ρ then φ" where ρ is a pattern and φ is a class distribution. r covers a tuple t when ρ covers t. The support of r, supp(r), is the number of tuples in I covered by ρ. φ is the class distribution over the tuples covered by r. φ is represented as a vector of m counters, i.e., one counter for each class, of the form: [n1; n2; : : : ; nm], where each n_i is the number of tuples in the training set that are (1) covered by r and (2) whose class attribute is c_i. Thus, supp(r) = Σ_i n_i. The confidence of a rule "if ρ then φ" is a measure of goodness of the rule; it is function of ρ and is often based on the entropy concept.

4.3.1 Classification

Data classification is a two-step process as shown in Fig. 4.1. In the first step, a model is built describing a predetermined set of data classes of concepts. The model can be applied to the classification of new data tuples constructed by analyzing database. Each tuple is assumed to belong to a predefined class, as determined by one of the attributes, called the class labels attribute. In the context of classification, data tuples are also referred to as samples, examples of object.

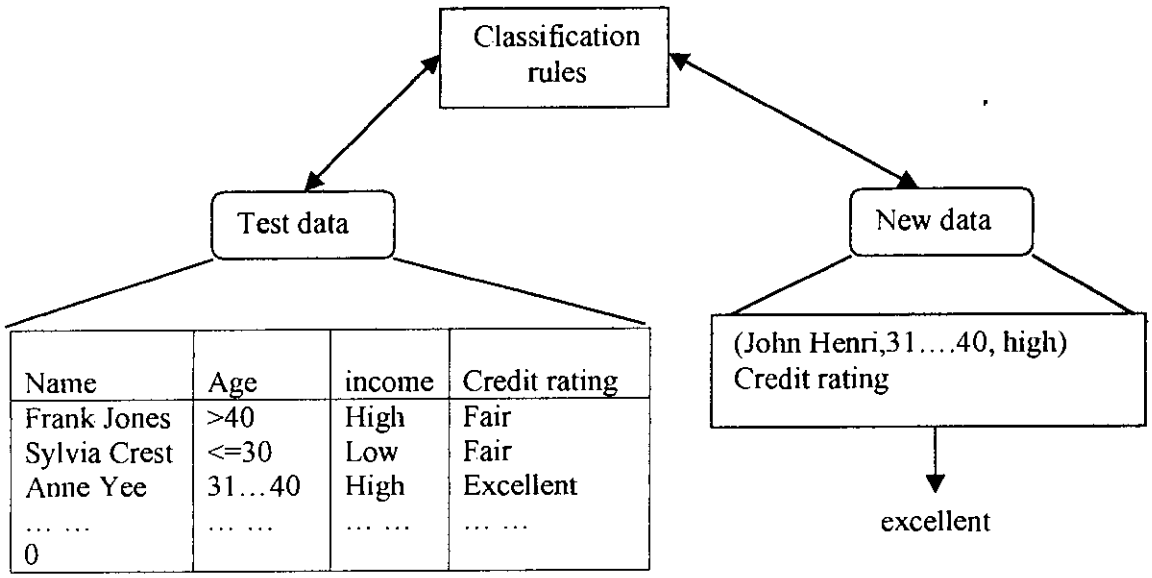
The data tuples analyzed to build the model collectively form the training data set. The individual tuples making up the training set are referred to as training samples and are randomly selected from the sample population. Since the class label of each training sample is provided, this step is also known as supervised learning (i.e., the learning of the model is "supervised" in that it is told to which class each training sample belongs). It contrasts with unsupervised learning (or clustering), in

which the class label of each training sample is not known, and the number or set of classes to be learned may not be known in advance.



Name	Age	income	Credit_rating
Sandy jones	<=30	Low	Fair
Bill lee	<=03	Low	Excellent
Courtney Fox	31...40	High	Excellent
Susan Lake	>40	Med	Fair
Claire Phips	>40	Med	Fair
Andre Beau	31...40	High	Excellent
... ...	... ...	... ...	... ...

Fig. 4.1 (a) *Learning for the data classification process:* a classification algorithm analyzes Training data. Here, the class label attribute is *credit_rating* and the learned model or classifier is represented in the form of classification rules.



Name	Age	income	Credit rating
Frank Jones	>40	High	Fair
Sylvia Crest	<=30	Low	Fair
Anne Yee	31...40	High	Excellent
... ...	... ...	... ...	... ...
0			

Fig. 4.1 (b) *Classification:* Test data are used to estimate the accuracy of the classification rules. If the accuracy is considered acceptable, the rules can be applied to the classification of new data tuples.

Typically, the learned model is represented in the form of classification rules, decision trees, of mathematical formulae. For example, given a database of customer. Data are used to estimate the accuracy of the classification rules. If the accuracy is considered acceptable, the rules can be applied to the classification of new data tuples. Credit information, classification rules can be learned to identify customers as having either excellent of fair credit ratings (Fig. 4.1 (a)). The rule can be used to categorize future data samples, as well as provide a better understanding of the database contents.

In thc second step (Fig. 4.1 (b)), the model is used for classification. First, the predictive accuracy of the model (or classifier) is estimated. The holdout method is a simple technique that uses a test set of class-labeled samples. These samples are randomly selected and are independent of the training samples. The accuracy of a model on a given test set is the percentage of test set samples that are correctly classified by the model. For each test sample, the known class label is compared with thc learned model's class prediction for that sample. Note that if accuracy of the model were estimated based on the training data set, this estimate could be optimistic since the learned model tends to overfeed the data (that is, it may have incorporated some particular anomalies of the training data that are not present in the overall sample population). Therefore a test is used.

If the accuracy of the model is considered acceptable, the model can be used to classify future data tupels of objects for which the class label is not known (such data are also referred to in machine learning literature as "unknown" of "previously unseen" data). For example, the classification rules learned in Fig. 4.1 (a) from the analysis of data from existing customers can be used to predict the credit rating of new customers. Classification has numerous applications including credit approval, medical diagnosis, performance prediction, and selective marketing.

Consider a database of customers on the AllElectronics mailing list. The mailing list is used to send out promotional literature describing new products and upcoming price discounts. The database describes attributes of the customers, such as their name, age, income, occupation, and credit rating. The customers can be classified as to whether or not they have purchased a computer at AllElectronics. Suppose that new customers are added to the database and that you would like to notify these customers of an upcoming computer sale. To send out promotional literature to every new customer can be quite costly. A more cost-efficient method would be to target only

those new customers who are likely to purchase a new computer. A classification model can be constructed and used for this purpose.

Suppose instead that you would like to predict and number of major purchase that a customer will make AllElectornics during a fiscal year. Since the predicted value here is ordered, a prediction model can be constructed for this purpose.

4.3.2 Issues Regarding Classification

This section describes issues regarding preprocessing the data for classification and prediction. Criteria for the comparison and evaluation of classification methods are also described in this section.

4.3.2.1 Preparing the Data for Classification

The following preprocessing steps may be applied to the data in order to help improve the accuracy, efficiency, and scalability of the classification or prediction process.

4.3.2.2 Data cleaning

This refers to the preprocessing of data in order to remove or reduce noise (by applying smoothing techniques, for example) and the treatment of missing values (e.g., by replacing a missing value with the most commonly occurring value for that attribute, or with the most probable value based on statistics). Although most classification algorithms have some mechanisms for handling noisy or missing data, this step can help reduce confusion during learning.

4.3.2.3 Relevance analysis

Many of the attributes in the data may be irrelevant to the classification or prediction task. For example, data recording the day of the week on which a bank loan application was filed is unlikely to be relevant to the success of the application. Furthermore, other attributes may be redundant. Hence, relevance analysis may be performed on the data with the aim of removing any irrelevant or redundant attributes from the learning process. In machine learning, this step is known as feature selection.

Ideally, the time spent on relevance analysis, when added to the time spent on learning from the resulting "reduced" feature subset, should be less than the time that

would have been spent on learning from the original set of features. Hence, such analysis can help improve classification efficiency and scalability.

4.3.3 Classification Modeling

In classification a mapping is learnt from a vector of measurements x to a categorical variable Y. The variable to be predicted is typically called the class variable (for obvious regions), and for convenience of mutation one will use the variable C, taking values in the set $\{c_1,\ldots\ldots,c_m\}$ to denote this class variables for the rest of this section (instead of using Y). The observed or measured variables $X_1,\ldots\ldots,X_p$ are variously referred to the rest as the features, attributes, explanatory variables, input variables, and so on–the generic term input variable will be used throughout this chapter. X will refer to as a p-dimensional Vector (this is comprised of p variables), where each component can be real-valued, ordinal, categorical, and so fourth. $X_j(i)$ is the jth component of the ith input vector, where $1\leq i\leq n$ and $1\leq j\leq p$. In the introductory discussion it will be implicitly assumed that the so-called "0-1" will be used as loss function, where a correct prediction incurs a loss of 0 and incorrect class prediction incurs a loss of 1 irrespective of the true class and the predicted class.

Two different but related general views of classification: the decision boundary (or discriminative) viewpoint, and the probabilistic viewpoint will be discussed in the following subsections.

4.3.3.1 Discriminative Classification and Decision Boundaries

In the discriminative framework a classification model f (x; θ) takes as input the measurements in the vector x and produces as output a symbol from the set $\{c_1\ldots\ c_m\}$. Consider the nature of the mapping function f for a simple problem with just two real-valued input variable X1 and X2. The mapping in effect produces a piecewise constant surface over the (X1, X2) plane; that is, only in certain regions does the surface take the value c1. The union of all such regions where a c1 is predicted is known as the decision region for class c1; that is, if an input x (i) falls in this region, its class will be predicted as c1 (and the complement of this region is the decision region for all other classes).

Knowing where these decision regions are located in the (X1, X2) plane is equivalent to knowing where the decision boundaries of decision surfaces are between the regions. Thus the problem of learning a classification function f can be thought as

being equivalent to learning decision boundaries between the classes. In this context, the mathematical forms can be used to describe decision boundaries, for example, straight lines of planes (linear boundaries), curved boundaries such as low-order polynomials, and any other more exotic functions.

In most real classification problems the classes are not perfectly separable in the X space. That is, it is possible for members of more than one class to occur at some (perhaps all) values of X-though the probability that members of each class occur at any given value x will be different. (It is the fact that these probabilities differ that permits us to make a classification. Broadly speaking, a point x is assigned to the most probable class at x). The fact that the classes "overlap" leads to another way of looking at classification problems. Instead of focusing on decision surfaces, a function $f(x; \theta)$ that maximizes some measure of separation between the classes can be looked for. Such functions are termed as discriminant functions.

4.3.3.2 Probabilistic Models for Classification

Let $P(c_k)$ be the probability that a randomly chosen object or individual i come from class c_k. Then $\sum k\ P(c_k) = 1$,assuming that the classes are mutually exclusive and exhaustive. This may not always be the case – for example, if a person had more than one disease (classes are not mutually exclusive), the problem might be modeled as a set of multiple two – class classification problems ("disease 1 or not," "disease 2 or not," and so on). Or there might be a disease that is not in the classification model (the set of classes is not exhaustive), in which case one could add an extra class c_{k+1} to the model to account for "all other disease". Despite these potential practical complications, unless stated otherwise mutually exclusive and exhaustive assumption will be used.

Imagine that there are two classes, males and females, and that $P(c_k)$, $k=1,2$, represents the probability that a person receives the appropriate chromosomes to develop as male or female. $P(c_k)$ is thus probabilities that individual i belong to class c_k if the user has no other information (no measurement X (i)) at all. $P(c_k)$ are sometime referred to as the class "prior probabilities," since they represent the probabilities of class membership before observing the vector X. Note that estimating the $P(c_k)$ from data is often relatively easy: if a random sample of the entire population has been drawn, the maximum likelihood estimate of $P(c_k)$ is just the

frequency with which ck occurs in the training dataset. Of course, if other sampling schemes have been adopted, things may be more complicated. For example, in some medical situations it is common to sample equal members from each class deliberately, so that the priors have to be estimated by some other means.

Object or individuals belonging to class k are assumed to have measurement vectors x distributed according to some distribution or density function P ($x \mid c_k, \theta_k$), where θ_k is unknown parameter governing the characteristics of class c_k. For example, for multivariate real-valued data, the assumed model structure for the X for each class might be multivariate normal, and the parameters θk would represent the mean (location) and variable (scale) characteristic for each class. If the means are far enough apart, and the variances small enough, one can hope that the classes are relatively well separated (on error) rate. The general problem arises when neither the functional form nor the parameters of the distributions have been estimated. Bayes theorem is applied to yield the posterior probabilities.

Figure 4.2 shows a simple artificial example with a single predictor variable X (the horizontal axis) and two classes. The upper two plots show how the data are distributed within class 1 and class 2 respectively. The plots shows the joint probability of the class and the variable X, P (x, c_k), k= 1,2. Each has a uniform distribution over a different range of X; class c_1 tends to have lower x values then class c_2. There is a region along the X axis (between values x_1 and x_2) where both class populations overlap.

The bottom plot shows the posterior class probability for class c_1, P ($c_1|x$) as calculated via Bayes rule given the class distributions shown in the upper two plots. For values of $x \leq x_1$, the probability is 1 (since only class c_1 can produce data in that region), and for values of $x \geq x_2$ the probability is 0 (since only class c_2 can produce data in that region). The region of overlap (between x_1 and x_2) has a posterior probability 1/3 for class c_1 (by Bayes rule) since class c_2 is roughly twice as like as class c_1 in this region. Thus, class c_2 is the base-optimal decision for any $X \geq X_1$ (noting that in the regions where P (X, C1) or P (X, C2) are both 0, the posterior probability is undefined). However, note that between X_1 and X_2 there is some fundamental ambiguity about which class may be present given an X value in this region; that is, although c2 is the more likely class there is a 1/3 chance of C_1 occurring.

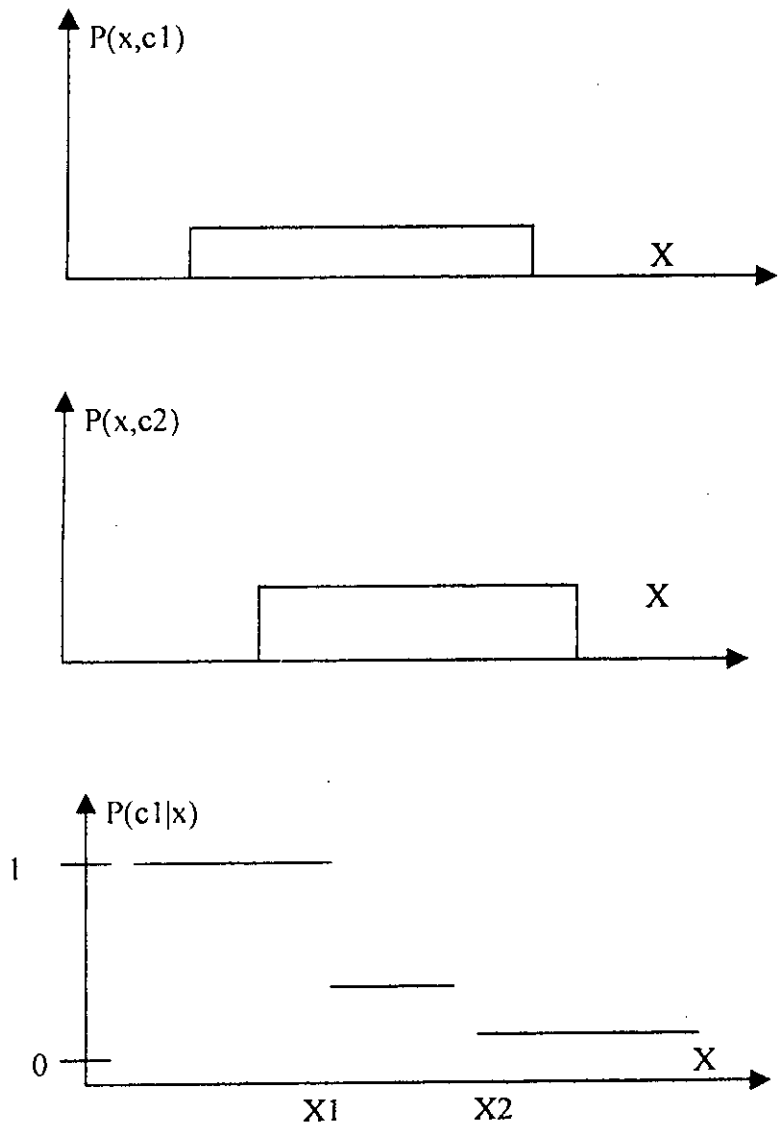


Fig. 4.2 A simple example illustrating posterior class probabilities for a two class one-dimensional classification problem.

In fact, since there is a 1/3 chance of making an incorrect decision in this region, and let guess from visual inspection that there is a 20% chance of an X value falling in this region, this leads to a rough estimate of a Byes error of about 20/3≈6.67% for this particular problem.

Now consider a situation in which x are bi variants, and in which the members of one class are entirely surrounded by members of other class. Here neither of the two X variables alone will lead to classification rules with zero error rates, but

(provided an appropriate model was used) a rule based on both variables together could have zero error rates. Analogous situations, though seldom quite so extreme, often occur in practice: new variables add information, so that one can reduce the Bayes error rate by adding extra variables. This prompts this question: why should not many measurements in classification problems simply used, until the error rate is sufficiently low? The answer lies in the bias-variance principle. While the Bayes error rate can only stay the same or decrease if more variables are added to the model, in fact the optimal classifier or the Bayes error rate is not known. Classification rule from a finite set of training data should be estimated, if the member of variables for a fixed number of training points is increased, the training data are representing the underlying distributions less and less accurately. The Bayes error rate may be decreasing, but there is a proper approximation to it. At some point, as the number of variables increases, the paucity of the approximation overwhelms the reduction in Bayes error rate.

The solution is to choose the variables with care; variables are needed that, when taken together, separate the classes well. Finding appropriate variables (or a small number of features-combinations of variables) is the key of effective classification. This is perhaps especially marked for complex and potentially very high dimensional data such as images, where it is generally acknowledged that finding the appropriate features can have a much greater impact on classification accuracy than the variability that may arise by choosing different classifications models. One data-driven approach in this context is to use a score function such as cross-validated error rate to guide a search through combinations of features- of course, for some classifiers this may be very computationally intensive, since the classifier may need to be retained for each subset examined and total number of such subsets is combinatorial in P (the number of variables).

4.4 Interestingness without Grouping Attribute Values

Interestingness of rules without grouping attribute values, i.e. the rules in which D_{ik} involves exactly one value v_{ik} for each k. then a rule without grouping is

$$R2: a_{i_1} = v_{i_1} \wedge \wedge \ldots \wedge a_{i_L} = v_{i_L} \rightarrow c \tag{4.15}$$

For simplicity, "primitive rules" will be used to stand for the rules without grouping attribute values. The basic idea of interestingness is that a rule is interesting if its accuracy is higher than predicted from more general rules. The larger the difference is, the more interesting the rule is. Let b_k be the conjunction of L-1 of L conditions excluding $a_{i_k} = v_{i_k}$

$$b_k = \left(a_{i_1} = v_{i_1} \wedge \ldots \wedge a_{i_{k-1}} = v_{i_{k-1}} \wedge a_{i_{k+1}} = v_{i_{k+1}} \wedge \ldots \wedge a_{i_L} = v_{i_L}\right) \tag{4.16}$$

Assuming independence of $a_{i_k} = v_{i_k}$ and b_k in a whole instance space and in class c, one can predict accuracy of the rule R2 from accuracy of a pair of more general rules,

$a_{i_k} = v_{i_k} \rightarrow c$ and $b_k \rightarrow c$.

$$P(c \mid a_{i_k} = v_{i_k} \wedge b_k) = P(a_{i_k} = v_{i_k} \wedge b_k \mid c)\, P(c) / P(a_{i_k} = v_{i_k} \wedge b_k) \tag{4.17}$$

That means,

$$P(c \mid a_{i_k} = v_{i_k} \wedge b_k) = P(c \mid a_{i_k} = v_{i_k})\, P(c \mid b_k) / P(c) \tag{4.18}$$

If the real accuracy of R2 is comparable to or lower than this exception, then R2 is a trivial conclusion from the rule pair $a_{i_k} = v_{i_k} \rightarrow c$ and $b_k \rightarrow c$, and R2 is not interesting at all. An interesting rule is require ed that is more accurate than expected from more general rules shown above for any k.

$$P_{1 \le k \le L}(c \mid a_{i_1} = v_{i_1} \wedge \ldots \wedge a_{i_1} = v_{i_1}) > P(c \mid a_{i_k} = v_{i_k})\, P(c \mid b_k) / P(c) \tag{4.19}$$

It is also required that accuracy of R2 is higher than that of $b_k \rightarrow c$. All instances covered by R2 are also covered by $b_k \rightarrow c$ and if the user already knows the rule $b_k \rightarrow c$ and if its accuracy is higher than R2, then R2 is useless to classify instances into c. Then the user will require

$$P_{1\le k\le L}(c|a_{i_1}=v_{i_1}\wedge\ldots\wedge a_{i_L}=v_{i_L})>P(c|b_k) \tag{4.20}$$

The above constraints give lower bound of accuracy and the interestingness of a primitive rule R2, $I_{rule}(R2)$, can be defined as a margin of its accuracy to satisfy the constraints as follows:

$$I_{rule}(R2)=\frac{acc(R2)-\max_{1\le k\le L}(\max(P(c/b_k),\frac{P(c/a_{ik}=v_{ik})P(c/b_k)}{P(c)}))}{1-P(c)} \tag{4.21}$$

where

$$acc(R2)=p(c|a_{i_1}=v_{i_1}\wedge\ldots\ldots\wedge a_{i_L}=v_{i_L}), \tag{4.22}$$

$$b_k=(a_{i_1}=v_{i_1}\wedge\ldots\ldots\wedge a_{i_{k-1}}=v_{i_{k-1}}\wedge a_{i_{k+1}}=v_{i_{k+1}}\wedge\ldots\ldots\wedge a_{i_L}=v_{i_L}). \tag{4.23}$$

The denominator, 1-P(c), is a normalization factor and I_{rule} (R2) becomes 1 when acc (R2) =1 and both of $a_{ik}=v_{ik}$ and b_k are independent of c.

$$\text{i.e. } p(c|a_{i_k}=v_{i_k})=p(c|b_k)=p(c) \tag{4.24}$$

4.5 Interestingness with Grouping Attribute Values

I_{rule} can be used to evaluate classification rules with grouping attribute values by replacing $a_{ik}=vik$ with $a_{ik}\in Dik$, but this sometimes gives large scores to inappropriate rules. Let us show an example in mushroom database why I_{rule} cannot be directly applied to evaluate rules with grouping. The next rule is an example with high estimation with respect to I_{rule}.

R3: cap_color $\in$ {brown, red} ^ stalk_root $\in$ {bulbos, rooted} → edible: 100% (4.25)

The probabilities required to calculate I_{rule} (R3) are

P (edible | cap_color $\in$ {brown, red}) = 0.50,
P (edible | stalk_root $\in$ {bulbous, rooted}) = 0.53,

P (edible) = 0.52.

And I_{rule} (R3) becomes 0.98. It looks like an exceptional rule because each condition in its body has no correlation with edible but only edible mushrooms satisfy the conjunction of them. However, by exploring relationship between edible and each attribute value, one can find the following probability:

P (edible|stalk_root) = 1.00

This means that the condition on cap_color is redundant for mushrooms with stalk_root = rooted and R3 is not interesting at all for classifying them. Instead, the following rule will be preferred that prohibits stalk_root = rooted and increase the probability of edible for all of covered instances compared with the rules whose bodies are one of two conditions.

R4: cap_color ∈ {brown, red} ^ stalk_root ∈ {bulbous} →edible: 100% (4.26)

In this rule, each attribute value correlates with edible as follows and none of them has strong relationship between edible.

P(edible/cap_color = brown) = 0.55,
P(edible/cap_color = red) = 0.42,
P(edible/stalk_root = bulbos) = 0.51.

The problem of I_{rule} is that it evaluates a rule based on only each condition in a body but doesn't concern each attribute value allowed in each condition. To remove this problem, interestingness of an attribute value will be introduced and evaluate how allowing the value contributes to the interestingness of the rule. Formally, the interestingness of an attribute value can be defined as $a_{i_k} = v$ in a rule $a_{i_1} \in D_{i_1} \wedge \ldots \wedge a_{i_L} \in D_{i_L} \rightarrow c$ as follows:

$$I_{value}(a_{ik} = v.a_{i_1} \in D_{i_1} \wedge \ldots \wedge a_{i_L} \in D_{i_L} \rightarrow c) = \max_{v_{i_h} \in D_{i_h}, (h \neq k)} I_{rule}(a_{i_1} = v_{i_1} \wedge \ldots \wedge a_{i_k} = v \wedge \ldots \wedge a_{i_L} = v_{i_L} \rightarrow c) \quad (4.27)$$

For example, the following four primitive rules can be used to evaluate attribute values in R3.

Cap_color = brown ^ stalk_root = bulbous → edible (Irule = 0.93),
Cap_color = brown ^ stalk_root = rooted → edible (Irule = 0.00),
Cap_color = red ^ stalk_root = bulbous → edible (Irule = 1.02),
Cap_color = red ^ stalk_root = rooted → edible (Irule = 0.00).

Stalk_rooted = bulbous appears in the first and in the third rule and its interestingness is 1.02. Instead, stalk_root =rooted appears in the second and in the last rule and I_{value} (stalk_root = rooted, R3) = 0.

Each instance covered by the original rule is covered by exactly one primitive rule and high score of I_{value} (a = v, R) means that there is at least one instance with a = v for which R works better than expected from more general rules. In opposite, I value (a =v, R) becomes low when more general rule can classify the instances with a = v with comparable or higher accuracy.

When extracting a rule whose interestingness is greater than or equal to a certain lower bound, LB, one require not only $I_{rule}(R) \geq$ LB but also I_{value} (a = v, R) >LB for each pair of an attribute and its value allowed in the body of the rule. To satisfy the above requirement, the interestingness of rules will be modified as follows:

$$IG_{rule}(R_1) = \min(I_{rule}(R_1), \min_{1 \leq k \leq L, v \in D_{i_k}} I_{value}(a_{i_k} = v, R_1)) \quad (4.28)$$

Clearly, $IG_{rule}(R)$ coincides with I_{rule} for a primitive rule R that allows exactly one value for each attribute appeared in this body. In the example of mushroom, P(edible |stalk_root = rooted) =1 leads I_{value}(stalk_root = rooted, R3) = 0. Then $IG_{rule}(R3)$ becomes 0 and our new criterion of interestingness, IG_{rule}, judges R3 is not

interesting at all. In contrast, all attribute values in R4 get large scores of I_{value} and IG_{rule} judges R4 is quite interesting.

4.6 Constructive Approach to Mine Interesting Rules (CAMIR)

CAMIR first enumerates primitive rules with large interestingness and then generates rules with value grouping by merging interesting primitive rules. It assumes all attributes are nominal and there is no missing attribute value. Inputs of the algorithm are the number of attributes in bodies of rules L. The lower bounds of interestingness and its supports are LB_i and LB_{sup}, respectively. LB_{acc} is the lower bound of accuracy. To simplify the discussion, the number of attributes appeared in rules bodies are fixed. CAMIR was iterated to extract a set of rules with different number of attributes in their bodies.

CAMIR can be described by Fig. 4.3. The major steps of CAMIR can be explained as follows.

Step 1) Read training set, training body length L, lower bound of support, accuracy, interestingness, LB_{sup}, LB_{acc}, LB_{ig}, and an objective function, f_{obj}.

Step 2) Initialize the set Rules as null set where interesting rules will be stored.

Step 3) Count class distribution for each combination of values $(v_1,\ldots\ldots\ldots\ldots v_L)$ of each combination of L attributes $(a_1, a_2,\ldots\ldots\ldots\ldots a_L)$.

Step 4) For each class c and for each combination of values$(v_1,\ldots\ldots\ldots\ldots v_L)$, generate a primitive rule $a_1=v_1\wedge\ldots\ldots\ldots\wedge a_L= v \rightarrow c$. After generating primitive rules, evaluate the rule, whether its interestingness is greater than or equal to the lower bound of interestingness.

Step 5) If the rule is interesting then pass it to S, the set of interesting primitive rules. Otherwise take another value combination of the current attribute combination and go to step 4. Continue this process until last value combination for the current attribute combination. In the same way continue for each class and for each attribute combination.

Step 6) Merge first primitive rule with the next primitive rule and check whether the interestingness of the merged rule exceeds that of all other primitive rules. If so, then pass it to a new set S'. Otherwise take the next rule and continue this process for other rules in S.

Step 7) Choose best rule R_{best} of which f_{obj} is maximum from the set S' and pass it to the set Rules.

Step 8) Repeat step 6 and step 7 for each rule is S.

Step 9) Rank rules in Rules with respect to f_{obj} and Return Rules.

It is observed from the above description that CAMIR uses minimum number of user specified parameters and users do not require any prior knowledge about the target problem. Here, the computation of interestingness of rules is simple and generalized. Moreover, Memory requirement is not large enough to execute the algorithm. The Pseudo-code of CAMIR is shown in Fig. 4.4.

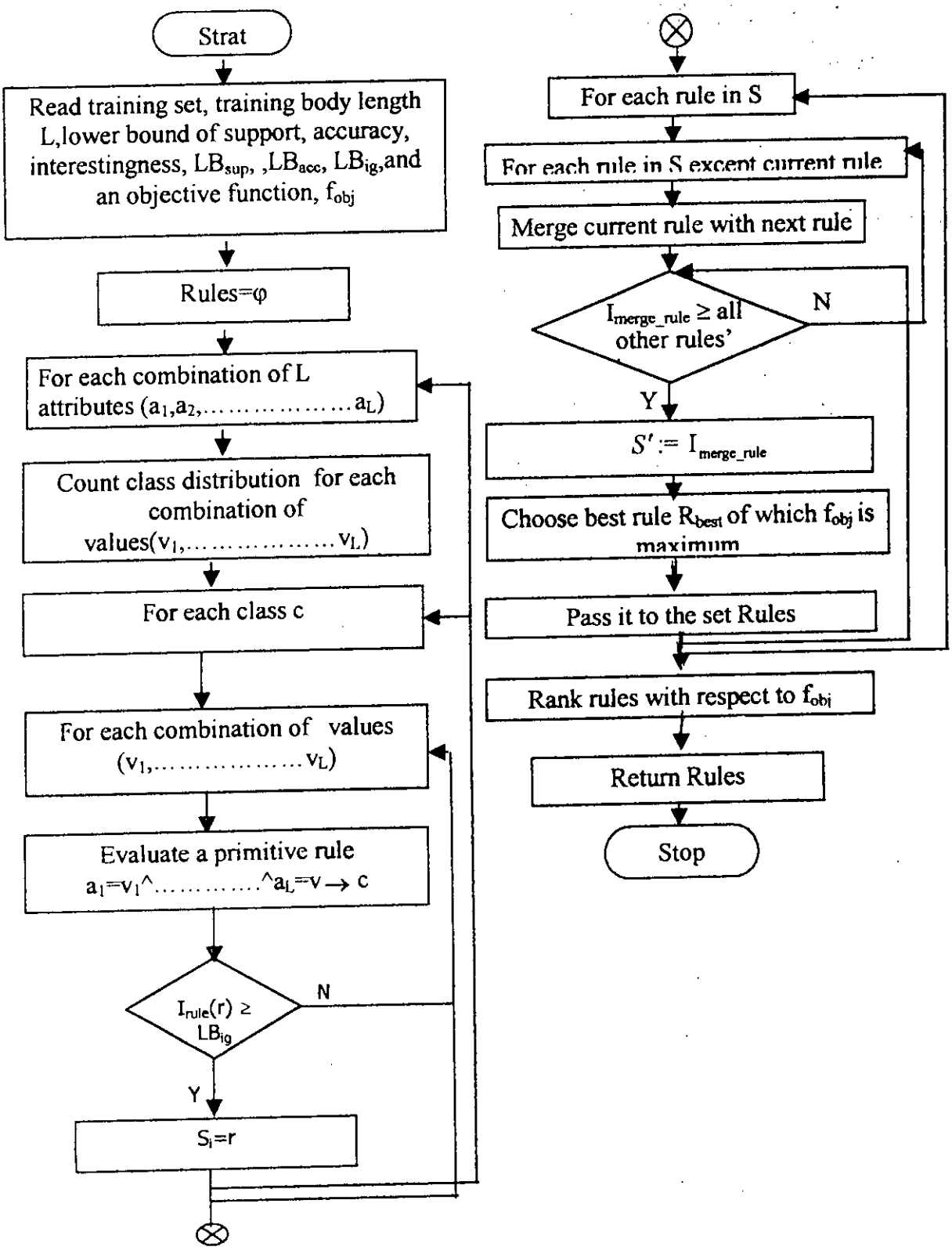


Fig. 4.3 Flowchart of CAMIR

Proccdure CAMIR (Training L, LB_{ig}, LB_{sup}, LB_{acc}, f_{nbj})

Inputs:

Training set, training body length, L, lower bound of support, accuracy and interestingness, LB_{sup}, LB_{acc}, LB_{ig}, an objective function, f_{obj}.

Output:

Set of interesting rules. R_{best}

Rules: = φ

For each combination of attributes $(a_{i_1}, a_{i_2}, \ldots\ldots\ldots\ldots, a_{i_t})$

Count class distribution for each combination of values $(v_{i_1}, v_{i_2}, \ldots\ldots\ldots\ldots, v_{i_t})$

For each class c

R_{best}: = *undefined*

For each combination of values $(v_{i_1}, v_{i_2}, \ldots\ldots\ldots\ldots, v_{i_t})$

Evaluate a primitive rule $a_{i_1} = v_{i_1} \wedge \ldots\ldots\ldots\ldots \wedge a_{i_t} = v_{i_t} \rightarrow c$

$S := \{r | r \text{ is primitive } \wedge I_{rule}(r) \geq LB_{ig}\}$

For each $s' \subseteq s$

Merge with the next rule.

If the interestingness exceed all other's interestingness then

If support $(R_{Merge}) \geq LB_{sup} \wedge ACC(R_{Merged}) \geq LB_{acc} \wedge IG_{rule}(R_{merged}) \geq LB_{ig}$ *then*

Rules: = *Rules* $\cup$ $\{R_{merge}\}$

Merge with the third rule and continue.

Rank rules in Rules with respect to f_{obj}.

Return Rules.

Fig. 4.4 Pseudo code of CAMIR

4.6.1 Time complexity

The most time consuming process in CAMIR is counting a class distribution of instances in each value pattern to evaluate primitive rules. Because CAMIR has to count distributions for all combinations of L attributes, the time complexity of this process is $O((^{M}C_{L}), N, L)$, where M is the number of attributes and N is the number of instances. This complexity depends on N linearly and on L exponentially. However, simple rules with few number of attributes are sometimes more important for knowledge discovery than complex rules with large number of attributes because one can easily understand the meaning of them.

Chapter 5

Experimental Studies

- Introduction
- Description of Datasets
- Experimental Setup
- Experimental Result
- Discussion

5.1 Introduction

The objective of this chapter is to describe the performance of CAMIR after applying on a number of benchmark data sets. These datasets were collected from UCI machine learning repository [18]. They are mushroom, letter recognition, breast cancer, statlog (heart), hepatitis and liver disorders. These problems have been the subject of many studies in machine learning. They contain real world data and are available by anonymous ftp at ics.uci.edu (128.195.1.1) in directory/pub/machine-learning-databases.

5.2 Description of Datasets

This subsection describes various datasets used in the thesis work to test the performance of CAMIR. Table 5.1 shows the number of instances, attributes and classes of these datasets. The following subsections will describe the real life application of these data sets with their attribute and class information.

5.2.1 The Mushroom Dataset

This dataset contains 125 inputs, 2 outputs, and 8124 examples. This data set is special in several respects: it is the one with the most inputs, the one with the most examples, the easiest one, and it is the only one that is not real in the sense that its examples are not actual observations made in the real world, but instead are hypothetical observations based on descriptions of species in a book. The examples correspond to 23 species of gilled mushrooms in the Agaricus and Lepiota Family. In the book, each species is identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. This latter class was combined with the poisonous one. 52% of the examples are edible; entropy 1.0 bit per example.

5.2.2 The Letter Recognition Dataset

Since all the above-mentioned problems are not too large classification problems, so letter recognition was chosen as a large problem for assessing the performance of our algorithm for a large problem. The data set representing this problem contained 20,000 examples. This was a 26-class problem. The objective is to identify each of a large number of black-and-white rectangular pixel displays as one of the 26 capital

letters in the English alphabet. The character images were based on 20 different fonts and each letter within these 20 fonts was randomly distorted to produce a file of 20,000 unique stimuli. Each stimulus was converted into 16 primitive numerical attributes (statistical moments and edge counts) that were then scaled to fit into a range of integer values from 0 through 15.

5.2.3 The Breast Cancer Dataset

The breast cancer data set was originally obtained from W.H. Wolberg at the university of Wisconsis Hospitals, Madison. The purpose of the data set is to classify a tumar as either benign or malignant based on cell descriptions gathered by microscopic examination. Input attributes are for instance the clump thickness, the uniformity of cell size and cell shape, the amount of marginal adhesion, and the frequency of bare nuclei. The dataset contains 9 attributes and 699 examples of which 458 are benign examples and 241 are malignant examples.

5.2.4 The Statlog (Heart) Dataset

The Stalog databases are a subset of the datasets used in the European Statlog project, which involves comparing the performances of machine learning, statistical, and neural network algorithms on data sets from real-world industrial areas including medicine, finance, image analysis, and engineering design. There were a number of databases used in this project, of which Heart Disease is selected for this experiment. This dataset is a heart disease database similar to a database already present in the repository (Heart Disease databases) but in a slightly different form. This database contains 13 attributes (which have been extracted from a larger set of 75). Variable to be predicted are absence (1) or presence (2) of heart disease. There were 270 observations in this dataset.

5.2.5 The Hepatitis Dataset

The Hepatitis data set was donated by G.Gong (Carnegie-Mellon University) via Bojan Cestnik, Jozef Stefan Institute. The purpose of the data set is to classify Hepatitis as either die or alive.

Input attributes are age, sex, steroid, antivirals, fatigue, malaise, anorexia, liver big, liver firm, spleenpalpable, spiders, ascites, varices, bilirubin, alk phosphate, sgot,

albumin, protime, histology. The dataset contains 20 attributes and 155 examples of which 32 are die and 123 are live.

Table 5.1: Characteristics of machine–learning benchmark datasets [18].

Dataset	Number of Instances	Attributes	Classes
Mushroom	8124	22	2
Letter Recognition	20000	16	26
Breast Cancer	699	9	2
Statlog (Heart)	270	13	2
Hepatitis	155	20	2
Liver disorders	345	7	2

5.2.6 The Liver Disorders Dataset

The Liver disorders data set was created by BUPA Medical Research Ltd. Input attributes are mcv, alkphos, sgpt, sgot, gammagt, drinks,s elector. The dataset contains 7 attributes and 2 classes and 345 instances.

5.3 Experimental Setup

In this study, LB_{acc} was set at 0.9 and L was set at 2,3 and 4. That means CAMIR extracts rules with 90% or higher accuracy that involves 2 to 4 attributes in the bodies. Minimum support, LB_{sup} was set at 1% for all data sets. The lower bound of interestingness, LB_{ig} was fixed as 0.8. As an objective function, f_{obj}, support and interestingness of rules were applied. All experiments were executed on Dell workstation with Processor-550 MHz Pentium 4. 256 MB of RAM was used in that setup and no shared memory was required in this experiment.

5.4 Experimental Results

Table 5.2 - 5.6 show a summary of experimental results, the number of extracted rules and CPU time for execution of CAMIR and DIG. In these tables, the CPU time is the amount of time required to find interesting rules. L is the number of attributes to

combine; LB_{ig} is the lower bound to satisfy the interestingness of the rules. To make fair comparison with CAMIR, DIG was run on same experimental setup as CAMIR algorithm. In these tables, average interestingness shows the average value of interestingness of the extracted rules and best one contains the maximum value of interestingness.

Table 5.2: Performance comparison of CAMIR and DIG [4] for the Mushroom dataset.

Algorithm	L	Number of extracted rules	Average Interestingness of rules	Best Interestingness of rules	System time (sec)
CAMIR	2	128	1.123508	2.363643	1080
	3	0	0	0	6950
	4	0	0	0	9060
DIG	2	90	1.196831	1.836721	1200
	3	0	0	0	7200
	4	0	0	0	10200

Table 5.2 shows that CAMIR works faster than DIG to extract interesting rules. For 2 (two) attributes it returns 38 more interesting rules with respect to the criterion. Since, the dataset contains a large number of instances (8124), the time requirement for both CAMIR and DIG were very large. For 3 (three) attributes CAMIR returns no interesting rules for this dataset. Time is exponentially increasing with respect to L.

Table 5.3: Performance comparison of CAMIR and DIG for the Letter Recognition dataset.

Algorithm	L	Number of extracted rules	Average Interestingness of rules	Best Interestingness of rules	System time (sec)
CAMIR	2	95	1.213452	2.207656	3780
	3	7	1.284562	2.459084	17100
	4	0	0	0	31280
DIG	2	79	1.196831	1.836725	3780
	3	5	1.321954	2.034753	17640
	4	0	0	0	33390

CAMIR was tested on Letter Recognition dataset since it is a large dataset having 20000 instances and 26 classes. This dataset was selected in order to evaluate our algorithm whether it can perform well on large dataset or not. From Table 5.3 it can be observed that interesting rules were extracted only for $L \leq 3$ and for 2 (two) attributes CAMIR returns 95 interesting rules while DIG extracts 79 interesting rules at same situation. For 3 (three) attributes CAMIR returns 7 interesting rules while DIG extracts only 5 interesting rules. Both of these two methods return no interesting rules for 4 attributes. In all combinations of attributes CAMIR uses less time than DIG.

Table 5.4: Performance comparison of CAMIR and DIG for the Breast Cancer dataset.

Algorithm	L	Number of extracted rules	Average Interestingness of rules	Best Interestingness of rules	System time (sec)
CAMIR	2	12	1.345213	1.983472	<1
	3	0	0	0	25
	4	0	0	0	67
DIG	2	0	0	0	<1
	3	0	0	0	47
	4	0	0	0	81

From Table 5.4 it can be observed that interesting rules were extracted only for $L \leq 3$ and for 2 (two) attributes CAMIR returns 12 interesting rules while DIG extracts no interesting rules at same situation. This test takes less than 1 (one) second to execute since it is applied on 699 instances only. Breast cancer data set return fewer interesting rules than mushroom because it has fewer number of instances than mushroom. And time requirement is also less here in this test.

Table 5.5: Performance comparison of CAMIR and DIG for the Statlog (heart) dataset.

Algorithm	L	Number of extracted rules	Average Interestingness of rules	Best Interestingness of rules	System time (sec)
CAMIR	2	302	1.321781	2.046721	49
	3	38	1.567832	2.654327	90
	4	0	0	1.0	172
DIG	2	10	1.196831	1.836724	62
	3	0	0	0	90
	4	0	0	0	185

Table 5.5 shows that CAMIR returns 192 more interesting rules than DIG for 2 (two) attributes using lesser time. For 3 (three) attributes CAMIR also returns 90 interesting rules while DIG returns only 38 rules. But here time requirement is same as DIG. For 4 (attributes) both CAMIR and DIG return no interesting rules but CAMIR uses less time in comparison to the time required by DIG.

Table 5.6: Performance comparison of CAMIR and DIG for the Hepatitis dataset.

Algorithm	L	Number of extracted rules	Average Interestingness of rules	Best Interestingness of rules	System time (sec)
CAMIR	2	241	1.389211	1.987652	498
	3	52	1.689326	2.930213	732
	4	0	0	0	1864
DIG	2	7	1.392431	1.930191	608
	3	0	0	0	868
	4	0	0	0	2030

Table 5.6 shows that in test performed on Hepatitis dataset, CAMIR returns 241 interesting rules for 2 (two) while DIG returns only 7 rules and for 3 (three) attributes CAMIR returns 52 interesting rules while DIG returns no rules. And time complexity of CAMIR is also lower than DIG for all combinations. Time is also exponentially increasing with respect to L

Table 5.7: Performance comparison of CAMIR and DIG for the Liver disorders dataset.

Algorithm	L	Number of extracted rules	Average Interestingness of rules	Best Interestingness of rules	CPU time (sec)
CAMIR	2	370	1.327891	2.139021	30
	3	0	0	0	60
	4	0	0	0	89
DIG	2	223	1.392831	1.891062	35
	3	0	0	0	90
	4	0	0	0	136

Liver disorders returns interesting rule for $L \leq 3$ as shown in Table 5.7 and for 2 (two) attributes it also outperforms DIG with respect to number of interesting rules and time requirement. For 3 (Three) attributes, though both CAMIR and DIG returns no interesting rules but time complexity of CAMIR is less than that of DIG.

5.5 Discussion

The performance of CAMIR is better than DIG for all of the datasets that wereused for test. There are two major differences that might lead to better performance by CAMIR in comparison with DIG.

The first reason is that CAMIR extracts more interesting rules than DIG with respect to the same criterion. It is not clear whether those rules are really interesting, but they are at least interesting from statistical point of view.

The second reason is that CAMIR uses less time than DIG to extract interesting rules. Though for both of the algorithms time requirement is high enough but in comparison to DIG for returning same number of interesting rules CAMIR requires less time for all datasets.

The other salient features of CAMIR that were found from the experiments can be mentioned as follows:

In these tables, it can be observed that the number of rules with large score of IG_{rule} decreases with L and no rule with L=4 satisfies $IG_{rule} \geq 0.8$ in all datasets. In all databases, CAMIR worked sufficiently fast when $L \leq 3$ and the time complexity of the method depends on L exponentially. CAMIR may require huge time when $L \geq$

5. However, this is not a critical disadvantage because rules with large interestingness are usually extracted with small L as shown in the above tables. For all datasets time is exponentially increasing with respect to L (number of attributes).

The algorithms presented in this thesis paper have been implemented on several data repositories, including mushroom, letter recognition, breast cancer, stat log (heart), liver disorders, and hepatitis. Among these datasets mushroom and letter recognition have large number of instances and the rest of the datasets have less number of instances. Since almost all datasets shown well performance, so it can be said that CAMIR perform well for both small and large datasets.

The experimental results show that CAMIR can extract interesting rules with respect to the criterion effectively and it outperforms DIG in terms of execution time and number of interesting rules for almost every dataset.

CAMIR evaluates interestingness with respect to exceptionality of rules. Domain experts feel interesting using exceptional rules rather than general rules. An exceptional rule is a specialization of general rule but concludes a different class with high accuracy [10]. General rules cover relatively many instances and may be trivial for human experts, but exceptional rules may give new knowledge to the experts. So, domain experts will be benefited from our research work.

Chapter 6

Conclusions

- **Conclusions**
- **Future Works**

6.1 Conclusions

In this research work, rules are extracted using interestingness of the grouping attribute value. This algorithm works on what kind of classification rules should be extracted by knowledge discovery systems and proposed a new criterion of interestingness of rules that evaluates a rule by comparing its accuracy with those of more general rules. In addition, this approach pointed out the necessity of evaluation of each attribute value allowed in a body of a rule to evaluate the rule correctly. In this work, a constructive approach to mine interesting rules (CAMIR) from a large set of association rules is proposed. This research work extracts interesting rules with grouping attribute values. CAMIR works on what kind of classification rules should be extracted by knowledge discovery systems using a new criterion of interestingness of rules.

Several optimization techniques are devised to speed up the discovery of interesting association rules. The performance study reveals that with association rules, one can discover more comprehensive knowledge; and careful selection of search space beforehand is critical to the effectiveness and efficiency of such association rule mining. It is not clear that rules are really interesting for the users, but these rules are at least quite interesting from statistical point of view. CAMIR evaluates the interestingness of rules by comparing the accuracy of rules. The essence of CAMIR is that it requires minimum number of user specified parameters and users do not require any prior knowledge about the target problem.

6.2 Future Works

In the future, the followings can be adopted to extend this work along the following dimensions:

- Data sets from practical domain sometimes involve instances in which values of some attributes are unknown. For each combination of attributes, CAMIR works with the instances in which values of all the selected attributes are known. For example, CAMIR ignores an instance whose value of a_1 is unknown when extracting a rule with an attribute set $\{a_1,a_2\}$ but uses the instance when extracting a rule with

$\{a_2, a_3\}$, if the values are known. In future, one can try to assume the missing attribute value using statistical approach. For example, in the above mentioned case one can take the most frequent value of attribute a_1 where the value is missed. If it were numerical attribute one can take the average value for the missing attribute value.

- This thesis work dealt with intra transaction association rules. But there is scopes to modify the algorithm for applying on inter transaction association rules. Intra transaction association rules means the associations rules that were generated from the transactions that occur in the same day or same month or same year or same event. Intertransaction association rules means the associations rules that were generated from the transactions that occur in different days or different months or different years or different events. So, further research work can be made on these intertransaction association rules.

- This thesis work investigated only single dimensional association rules. Later multidimensional association rules can be used for the evaluation of the algorithm. In chapter three, we have already discussed about single dimensional and multi dimensional association rules. Rule (3.3) is a single-dimensional association rule since it refers to only one dimension, buys. If a rule references two or more dimensions, then it is a multidimensional association rule. Rule (3.4) is considered a multidimensional association rule since it involves three dimensions: age, income, and buys.

References

[1] X. Wu and S. Zhang, "Synthesizing High-Frequency Rules from Different Data Sources," *In IEEE Transaction on Knowledge and Data Engineering*, vol. 15,no. 2, pp. 353-367, March 2003.

[2] R. Omiecinski, "Alternative Interest Measure for Mining Associations in Databases," *In IEEE Transaction on Knowledge and Data Engineering*, vol. 15,no. 1, pp. 57-69, 2003.

[3]. R. Agrawal, H. Mannila, R. Srikant, H. Toivonen and Verkamo, "Fast Discovery of Association Rules," *In Advances Knowledge Discovery and Data Mining*, AAAI press, pp. 307-328, 1996.

[4]. Nobuhiro Yugami, Yuiko Ohta and Seishi Okamoto, "Fast Discovery of Interesting Rules," *In Proceedings of the 4th Pacific-Asia Conference*, PADKK 2000, pp. 17-28, April 18-20, 2000.

[5]. Heikki Mannila, Hannu Toivonen and Inkeri Verkam, "Efficient algorithms for discovering association rules," *In KDD-94, AAAI Workshop on Knowledge Discovery in Databases*, pages 181-192, Seattle, Washington, July 1994.

[6]. E. Suzuki, "Autonomous Discovery of Reliable Exception Rules," *The Third International Conference on Knowledge Discovery and Data Mining, AAAI Press*, pp.259-262, 1997.

[7]. M Kamber and R. Shainghal, "Evaluating the Interestingness of Characteristic Rules," *In Proceedings of the Second International Conference on Knowledge Discovery and Mining (KDD-96)*, AAAI press, pp.263-266, 1996.

[8]. G. Piatesky-Shapiro, "Discovery, Analysis and Presentation of Strong Rules," *In Knowledge Discovery in Databases, AAAI press*, pp.229-248, 1991.

[9]. J. R. Quinlan, "Programs for Machine Learning," *Morgan Kaufmann Publishers*, 1993.

[10]. I. Guyon, N. Matic and V. Vapnik, "Discovering Informative Patterns and Data Cleaning," *In advances in Knowledge Discovery and Mining, AAAI press*, pp.181-203, 1996.

[11]. Rakesh Agarwall,Tomasz Imielinki and Arun Swami, "Mining association rules between sets of items in large databases," *In Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOID'93)*, pp. 207-215, May 1993.

[12]. Rakesh Agrawal and Remerkrishnan Srinkant, "Fast algorithm for mining association rules in large databases," *In Proceedings of the twentieth International Conference on Very Large Data Bases (VLDB'94)*, pp. 487-499, September 1994.

[13]. A. Savasere, E. Omiecinski and S. Navathe, "An efficient algorithm for mining association rules in large databases," *In Proc. of the VLDB Conference*, Zurich, Switzerland, September 1996.

[14]. Ramakrishnan Srikant and Rakesh Agrawal, "Mining Generalized Association Rules," *In Proc. of the 21st Int'l Conference on Very Large Databases*, Zurich, Switzerland, September 1996.

[15]. G. Piatetsky-Shapiro, "Discovery, analysis and presentation of strong rules," *In G. Piatetsky Shapiro and W. J. Frawley, editors, Knowledge Discovery in Databases*, pages. 229-248, AAAI/MIT Press, Menlo Park, CA. 1991.

[16]. A. Silverschatz and A. Tuzhilin, "What Makes Patterns Interesting in Knowledge Discovery Systems," *IEEE Transactions on Knowledge and Data Engineering*, vol.8, No.6, pp. 970-974, 1996.

[17]. Jiawei Han and Micheline Kamber, "Data Mining: Concepts and Techniques," *Morgan Kaughman*, 2002.

[18]. C. Olake, E. Keogh and C. Merz, "UCI Repository of machine learning databases," *http://www.ics.uci.edu/~mlearn/MLRepository.html(1999).*

[19]. F. Hussain, H. Liu, E. Suzuki and H. Lu, "Exception Rule Mining with a Relative Interestingness Measure," *In Proceedings of the 4th Pacific-Asia Conference, PADKK 2000*, pp. 86-97, April 18-20, 2000.

[20]. P. Smyth and R. M. Goodman, "Rule Induction Using Information Theory," *In Knowledge Discovery in Databases", AAAI press*, pp.160-176, 1991.

[21]. Tao Zhang: "Association Rules", *Trida Ltd.*, CA 94404.

[22]. 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, AAAI press, pp.1022-1027, 1993.

[23]. Md. Sheikh Sadi, Chowdhury Mofizur Rahman, Hafiz Md. Hasan Babu, "An Efficient and Coherent Method to Cluster Web Documents Using Data Mining," *ICECE, 26-28 December*, 2002.

[24]. R. Srikant, Q. Vu and R. Agrawal, "Mining Association Rules with Item Constraints". *The third International Conference on Knowledge Discovery and Data Mining (KDD-97)*, pp.67-73, 1997.