Rule Mining with Prior Knowledge
- A Belief Networks Approach

Zonglin Zhou[1] Huan Liu[2] Stan Z. Li[3] Chin Seng Chua[1]
[1] School of EEE, Nanyang Technological University
BLK S1, Nanyang Avenue, Singapore 639798
ezlzhou@ntu.edu.sg, Fax: (+65)793-3318
[2] Department of Computer Science, School of Computing
National University of Singapore
Lower Kent Ridge Road, Singapore 119260
[3] Microsoft Research China
Sigma Center, 49 Zhichun Road
Beijing 100080, R.R. China

Abstract

Some existing data mining methods, such as classification trees, neural networks and association rules, have the drawbacks that the user's prior knowledge cannot be easily specified and incorporated into the knowledge discovery process, and the rules mined from databases lack quantitative analyses. In this paper, we propose a belief networks method for rule mining, which takes the advantage of belief networks as the directed acyclic graph language and their function for numerical representation of probabilistic dependencies among the variables in the database, so that it can overcome the drawbacks. Since belief networks provide a natural representation for capturing causal relationship among a set of variables, our proposed method can mine more general correlation rules which can capture the relationship of more than two attribute variables. The potential application of the proposed method is *demonstrated through the detailed case studies on benchmark databases.*

Keywords: *Belief networks, rule mining, machine learning, correlation rule, classification rule.*

1 Introduction

The field of knowledge discovery in databases (KDD, data mining) has been defined in [6, 21] as the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns from databases. In the KDD field, one important issue is how to efficiently incorporate into a knowledge discovery process the user's prior knowledge (beliefs or preferences), in order to conduct a search and speed up the process of knowledge discovery from databases. Another difficult issue is how to accurately represent the rules mined from databases in terms of both qualitative and quantitative analyses. The existing approaches in knowledge discovery from databases seem inadequate in handling the two issues. In the earlier work of data mining, the general goal of data mining is to discover patterns having high predictive accuracy. For this reason, researchers and practitioners adopted the model of automated data mining and focused on the development of discovery tools and methods that would intelligently extract "statistically strong" patterns which satisfy some objective criteria (like support, confidence level). Although the automated data mining model seems to be powerful, it is prone to suffering from practical problems due to the user having little or no control at all of the knowledge discovery process. Typically, after a highly intensive process of knowledge discovery (which may range from hours to days), the user could be confused by the overabundance of rules returned, of which only a small fraction might be what he/she is really interested in. To the contrary, in user-guided data mining, the user has the primary control over the kind of rules to be discovered. Typically, the user can incorporate his/her prior knowledge and preferences into guiding the discovery process to return the kind of rules he/she requires. Such a user-oriented methodology is seen as a good alternative for a data mining tool to be effective and useful for solving real-world problems.

In this paper, we propose a belief network method to take advantage of a user's prior knowledge for rule mining. This method can accurately mine *classification rule* and *correlation rule* from databases.

The rest of this article is organized as follows. Section 2 briefly describes the relationship between belief networks and rule mining. Section 3 presents an algorithm of learning belief networks. The study on two benchmark databases using the learning algorithm is presented in Section 4. Section 5 extends the scope of applying the belief networks approach. Section 6 concludes with some discussion and future research.

2 Belief Networks and Rule Mining

A belief network ([22], also called Bayesian network or probabilistic network) consists of a directed acyclic graph (DAG) of nodes and arcs, referred to as a network structure, and a set of conditional probabilities associated with the graph. A node in DAG represents a variable. In this article, we take the causal interpretation of the arcs and consider an arc as a causal link representing the existence of direct causal influence between the linked variables. These conditional probabilities associated with the graph consist of $P(v_i|\pi_i)$, the conditional probability for each node v_i, given its parent set π_i. When v_i does not have any parent, i.e., v_i is a root, $P(v_i|\pi_i)$ becomes $P(v_i)$, the prior probability of v_i. A conditional independence assumption is made for such networks: $P(v_i|\pi_i, v_j) = P(v_i|\pi_i)$ where v_j is any variable other than v_i itself and its descendants. This assumption allows one to recover the joint probability distribution of all the variables from the given conditional probabilities by the simple equation

$$P(v_1, v_2, ..., v_n) = \prod_{i=1}^{n} P(v_i|\pi_i) \tag{1}$$

For any set X of variables in a belief network, the *ancestral set* $an(X)$ consists of variables in X and ancestors of those variables, i.e., variables from which there are directed paths to at least one variable in X. It is evident that $v_i \in an(X)$ implies $\pi_i \in an(X)$. In fact, the Eq. (1) also holds for all the variables in an *ancestral set*. The fact provides a basis on which *classification rule* and *correlation rule* will be derived later in this paper.

Belief networks are actually a rich language of directed acyclic graphs, which facilitates users to specify their prior knowledge. For instance, if an item or attribute is represented by a node, by adding or deleting a node, it is easy for a user to specify whether an item or attribute should be kept or not. By adding or deleting some arcs, the correlation relationship between an item or attribute and other some items or attributes is easy to be specified as well. Belief networks can handle multiple users' prior knowledge in that the directed acyclic graphs can be easily modified by different users at different times to represent their prior knowledge. The Figure 1 below shows a simple belief network.

A belief network consists of a qualitative part, encoding a domain's variables and the probabilistic influences among them in a directed acyclic graph, and a quantitative part, encoding conditional probabilities over these variables. By learning belief networks from databases, both qualitative and quantitative parts of a belief network can be determined. The user's prior knowledge (beliefs or preferences) about both qualitative and quantitative parts can be set up prior to learning. Once the qualitative and quantitative parts of a belief network are initialized based on user's prior knowledge, the learning from real databases will be invoked. The learning will refine the network. Some rules consistent with the user's expectation will appear in the learned network. Moreover, some learned rules different from user's prior knowledge will also appear in the network. Therefore, we can use the learned belief networks to confirm and/or verify whether our prior knowledge is consistent with the database and discover new rules.

Another main purpose of this paper is to present a novel technique, which is based on learning belief networks, to mining *correlation rule* and *classification rule*. A *correlation rule* is to depict the relationship between an attribute variable and other attribute variables. A *classification rule* is to reflect the relationship between the class variable and attribute variables. So a *correlation rule* usually involves an attribute in its right hand side (RHS), and a *classification rule* always involves a class variable in its RHS. There have been many methods such as classification tree, naive Bayes model, neural networks for classification rule mining (e.g., [18]). There is, however, little work that discusses correlation rule mining. In [20], the authors treated an attribute as a random variable, and calculated the coefficient of correlation between any two attributes by a probability method. However, a correlation rule conducted by their method can only capture the relationship between any two attributes. Since a belief network can capture causal relationship among a set of attribute variables [3], the approach of belief networks can mine out the much more general correlation rules which can capture the relationship among more than two attribute variables.

By learning a belief network from database, correlation and classification rules are represented in terms of the learned network's structure and conditional probabilities among different items or attributes. The rules mined are in the following form:

- If $(A_1\theta\nu_1)$ and $(A_2\theta\nu_2)$ and ... and $(A_n\theta\nu_n)$ then $(A_{n+1}\theta\nu_{n+1})$, for correlation rules.
- If $(A_1\theta\nu_1)$ and $(A_2\theta\nu_2)$ and ... and $(A_n\theta\nu_n)$ then C_j, for classification rules.

Where A_i's are the attributes, ν_i's are constants, θ's are relational operators $(=, \leq, \geq, <, >)$, and C_j is one of the class labels.

The above forms of rules are widely adopted in other literatures (e.g., [18]). Based on the preprocessing to the datasets, in this paper the rules mined are represented as follows for notational simplicity:

- $(A_1 = \nu_1, A_2 = \nu_2, ..., A_n = \nu_n) \rightarrow (\Rightarrow)(A_{n+1} = \nu_{n+1})$, for correlation rules.

- $(A_1 = \nu_1, A_2 = \nu_2, ..., A_n = \nu_n) \to (\Rightarrow)(C = c_j)$, for classification rules.

Where C is the class variable, c_j's are constants.

The conditional probabilities among some items or nodes can be viewed as a kind of correlation coefficient ρ. The number of ρ indicates the strength of the correlation. When $\rho > 0$, the attributes represented by these nodes have correlation. When $\rho = 0$, they have no correlation. The value of ρ can be used to determine whether a correlation rule is "strong" or "weak". Say a user sets Min_ρ to be 0.5, then only the correlation rules with $\rho > 0.5$ are considered "strong". For classification rule mining, a user can also set Min_ρ as a threshold. The class variable can be treated as a child node, only its parents with ρ exceeding the threshold are considered to be "strongly" related to the class variable.

The correlation and classification rules discussed in this paper are different from the association rules widely used in data mining community. The correlation and classification rules here are mined based on the structure and conditional probabilities of the learned network, and no support and confidence level specified as in mining association rules.

In the next section, we introduce a Bayesian method for learning belief networks called "K2", which was originally developed by Cooper and Herskovits [1]. It can automatically extract the most probable belief-network structure, given a database. Once such a structure is obtained, the conditional probabilities can be derived from the database. The resulting belief network will pave the way for rule mining.

3 The K2 Algorithm

The K2 is aimed at constructing the most probable network structure B_s, given a database of cases D, i.e., the one with the highest posterior probability $P(B_s|D)$. A set of structures can be ranked by their posterior probabilities. To rank the posterior probabilities, it is only needed to calculate the joint probablity $P(B_s, D)$. Let $r_i > 1$ be the number of values variable v_i can have and q_i the number of distinct instantiation patterns its parent set π_i has in D. Under a set of assumptions as in [1], then,

$$P(B_s, D) = P(B_s) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}! \tag{2}$$

where N_{ijk} is the number of cases in D in which v_i has the kth of the r_i values, and π_i is the jth of the q_i instantiations, and $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$.

The K2 employs a greedy algorithm to search for a B_s which is (sub)optimal measured by Eq. (2). The K2 requires a preexisting order among all nodes in such a way that only those nodes which precede a particular node can possibly be chosen as its parents during learning. The ordering is based on a user's prior knowledge. For simplicity, we specify this node order by the node subscripts, *i.e.*, node v_j can be chosen as a parent of node v_i only if $j < i$. Let π_i denote the parent set of the node v_i. The K2 begins with $\pi_i = \emptyset, i = 1, 2, ..., n$. For a given $v_i, i = 2, 3, ..., n$, it adds incrementally a parent node $v_j, j < i$ into the current π_i, whose addition most increases the probability of the resulting structure. The process continues until the addition of no single parent into the current π_i can increase the probability. When all B_s structures are assumed to have equal prior probability, the following equation is used to compute the probability at each step:

$$g(i, \pi_i) = \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}! \tag{3}$$

The K2 runs as follows:

for $i = 1$ **to** n **do**

$\pi_i := \emptyset$

for $i = 2$ **to** n **do**

repeat

select $v_j \in \{v_1, ..., v_{i-1}\} \backslash \pi_i$ that maximizes $g(i, \pi_i \bigcup \{v_j\})$

$\Delta := g(i, \pi_i \bigcup \{v_j\}) - g(i, \pi_i)$

if $\Delta > 0$ **then** $\pi_i := \pi_i \bigcup \{v_j\}$

until $\Delta \leq 0$ **or** $\pi_i = \{v_1, ..., v_{i-1}\}$

The K2 holds true under the following four assumptions:

1. The variables in B_s are discrete;

2. There are no cases that have variables with missing values;

3. Cases occur independently, given a belief network model;

4. Before observing D, we are indifferent regarding which numerical probabilities to assign to the belief network with structure B_s.

Currently, the most common way for belief networks to handle continuous variables is to discretize them. Many discretization techniques can be found in [16]. There are also many techniques to handle the variables with missing values [1, 7, 8, 11, 12, 25]. For example, in [25] an approach called *Bound* and *Collapse* is proposed to deal with the missing data. Assumption 3 is reasonable for many real-world databases due to the way in which data is obtained. Assumption 4 is that for the structure B_s, the prior distribution density for the conditional probabilities is uniform. The four assumptions are restrictive. There has been a great deal of recent work building on general results from Bayesian statistics to relax these assumptions [10, 13, 12, 17].

The K2 requires an ordering on all the nodes prior to learning. Such an ordering prevents the formation of causal cycles. Moreover, it restricts the set of all possible parents of a node to the set of all nodes preceding it. Learning will then determine the genuine parents of a node only from the restricted set of possibilities. The ordering may be chosen to reflect causality. Thus, the ordering is important for the constructing of belief networks. If we would like to avoid such a requirement, one approach is to search for likely undirected graphs and use these as starting points in searching for directed graphs [1]. By doing so, however, it will make the belief network model very general in representing the relationship between a node and its parents. Such generality may require that an excessive number of conditional probabilities be available because the number of conditional probabilities for each node is exponential to the number of its parents. This problem becomes more serious for networks of high connectivities.

Once the belief network structure is extracted by the K2, we shall start deriving the conditional probabilities, the quantitative part of the belief network.

Let θ_{ijk} denote the conditional probability $P(v_i = \nu_{ik} | \pi_i = w_{ij})$ - that is, the probability that the node v_i has value ν_{ik}, for some k from 1 to r_i, given that the parents of v_i, represented by π_i, are instantiated as w_{ij}. Call θ_{ijk} a *network conditional probability*. Let ξ denote the above four assumptions. Consider the value of $E[\theta_{ijk} | D, B_s, \xi]$, which is the expected value of θ_{ijk} given database D, the belief-network structure B_s, and the assumptions ξ. Then the following result is derived in [1]:

$$E[\theta_{ijk} | D, B_s, \xi] = \frac{N_{ijk} + 1}{N_{ij} + r_i} \tag{4}$$

There are many methods of learning belief networks such as [9, 14, 19, 23]. The K2 is a representative Bayesian learning approach. It has been implemented by some programs for the induction of belief networks from data (e.g., [26]). To our best knowledge, it is the first time for the K2 to be applied to rule mining. From our work [24], it is found that the K2 has certain inherent noise-resistance capability. Especially, the K2 can handle multi-value variables, so it is suitable to be applied to real-world problems of different domains.

4 Case Studies

In this section, we present results of applying the K2 algorithm to two benchmark databases: credit card screening data from a Japan bank, and mushrooms data from the Audubon Society Field Guide to North American Mushrooms[1].

4.1 Mining Credit Card Screening Data

The domain for this case study concerns the approval of credit card applications using a database provided by a Japan bank. The database contains 10 attributes, a class variable, and 125 records. These attributes include age (A_1) and sex (A_2) of the applicants, work years (A_3) , loan months (A_4), whether or not jobless (A_5), amount of savings (A_6), marriage status (A_7), living area (A_8), months between purchase and payment (A_9), and the kinds of purchase (A_{10}), The class variable is whether or not being granted for credit card (C).

The user's prior knowledge can be specified in belief networks as described in the second paragraph of Section 2. We suppose the following basic prior knowledge. For a more experienced data miner, there could be more to be pre-specified.

- All the attributes and the class variable in the database should be kept in the database; and
- Attributes A_1 and A_2 have no parents, A_3 has possible parents $\{A_1, A_2\}$, A_4 possible parents $\{A_1, A_2, A_3\}$, ..., C possible parents $\{A_1, A_2, ..., A_{10}\}$.

With the above prior knowledge, we set up an initial belief network and start the learning from the credit card screening database. The K2 is used to refine the network. We have the following result from the learning:

[1] As to the two databases, please see *http://www.ics.uci.edu/ mlearn/MLSummary.html*

- Attributes A_1, A_2, A_4, A_9 have no parents,
- A_3 has parent $\{A_1\}$,
- A_5 has parents $\{A_1, A_3\}$,
- A_6 has parent $\{A_4\}$,
- A_7 has parents $\{A_1, A_3, A_5, A_6\}$,
- A_8 has parents $\{A_5, A_7\}$,
- A_{10} has parents $\{A_4, A_6, A_9\}$,
- C has parents $\{A_3, A_5\}$.

The belief network learned from the data is shown in Figure 2. In Figure 2, the attribute A_2 is isolated and independent of all the other attributes.

The relationship between a node and its parents is also called causal dependency. The indirect dependencies between a node and the other nodes except its parents may exist, and it is possibly conditionally independent of them. As described in Section 3, the K2 first determines the parents of each node, then calculates the conditional probability of a node given its parents by Eq. (4). The conditional probabilities serve as the measurement of the strength of causal dependencies between the nodes and their parents. Since the calculation of the conditional probabilities depends on the corresponding database, they are also said to be learned from the database.

We note that the class variable C, whether or not being granted credit card, is only caused by A_3 (work years) and A_5 (whether or not jobless). C is either independent of the other attributes such as A_4 (loan months), A_7 (marriage status), etc., or has so weak causal dependencies with them that those dependencies cannot be captured by K2. Attribute A_5 has causal dependencies with A_1 (age) and A_3 (work years) only, and has no or very weak causal dependencies with the other attributes such as A_4 (loan months), etc. Obviously, the discovered knowledge is different from the prior knowledge. If we assume that the information in a database is complete, we should suspect that our prior knowledge is incorrect, and it should be modified based on our findings from the learned network.

In what follows, we will discuss the learned quantitative part of the belief network.

Table 1 gives the learned conditional probabilities for attribute A_5 and class variable C with regard to their respective parents. These conditional probabilities are learned from the original database of 125 records. The others are omitted due to space constraint. In Table 1, for attribute A_1, 0, 1 and 2 denote the age between 0-40, 40-60, and over 60 years old, respectively; for attribute A_3, 0, 1 and 2 denote 0-5, 5-20, and over 20 work years; for attribute A_5, 0 and 1 denote non-jobless, and jobless; for class variable C, 0 and 1 denote non-granted, and granted for credit card.

When focusing on C - the class variable, and mining the classification rules in terms of the belief network in Figure 2 and Table 1, we can obtain the following some rules if $Min_\rho = 0.5$:

Rule 1) $(A_3 = 0, A_5 = 1) \Rightarrow (C = 0)$,

Rule 2) $(A_3 = 0, A_5 = 0) \rightarrow (C = 0)$,

Rule 3) $(A_3 = 2, A_5 = 0) \Rightarrow (C = 1)$,

Rule 4) $(A_3 = 0, A_5 = 1) \rightarrow (C = 1)$,

Rule 5) $(A_3 = 1, A_5 = 0) \Rightarrow (C = 1)$,

Rule 6) $(A_3 = 1, A_5 = 0) \rightarrow (C = 0)$,

...

Where $A \Rightarrow B$ means that A is “strongly” related to B, $A \rightarrow B$ indicates that A is “weakly” related to

B.

The above rules are only the part of all the possible classification rules implied in the network in Figure 2. They are derived from direct parent-child dependencies. From Figure 2, A_1 is conditionally independent of C given A_3 and A_5. As an example, let's see how to establish the classification rules between C and the other attributes except its parents such as A_1.

Since

$$\begin{aligned} P(C|A_1) &= \frac{P(C, A_1)}{P(A_1)}, \quad (5) \\ an\{C, A_1\} &= \{C, A_1, A_3, A_5\}, \end{aligned}$$

in terms of Eq. (1) for the ancestral set,

$$P(C, A_1) = \sum_{A_3, A_5} P(C, A_1, A_3, A_5) = \sum_{A_3, A_5} P(A_1)P(A_3|A_1)P(A_5|A_1, A_3)P(C|A_3, A_5) \quad (6)$$

From Eq. (5) and (6), we can obtain the conditional probability for C given A_1, then the classification rule between C and A_1 can be acquired accordingly.

From Table 1, we can see that some conditional probabilities are very high and close to 1, so the corresponding parent patterns very "strongly" cause the class variable with corresponding outcomes. As an example, from Rule 1, if work-year is between 0 to 5, and the applicant jobless, then the outcome of classification for the applicant with a high probability (0.8125) is non-granted for credit card. From Table 1, also it can be found that some conditional probabilities are very small and close to 0, which indicates that the corresponding parent patterns very "weakly" cause the class variable with corresponding outcomes. For instance, from Rule 2, if work-year is between 0 to 5, and the applicant non-jobless, then the outcome of classification for the applicant with a small probability (0.4091) is non-granted for credit card, i.e., the applicant is of high probability granted for credit card.

As mentioned earlier, a correlation rule is yielded in terms of whether or not there exists non-zero conditional probability among attributes. Whether it is “strong” or “weak” depends on the threshold chosen. From Table 1, all the parent patterns of attribute A_5 have positive correlation with it. If we set $Min_\rho = 0.5$, the correlation rules with $Min_\rho > 0.5$ are considered “strong”, and the others “weak”.

For example, the following correlation rules are “strong”:

Rule 1) $(A_3 = 1, A_1 = 1) \Rightarrow (A_5 = 0)$,

Rule 2) $(A_3 = 0, A_1 = 1) \Rightarrow (A_5 = 1)$,

Rule 3) $(A_3 = 0, A_1 = 2) \Rightarrow (A_5 = 0)$,

Rule 4) $(A_3 = 2, A_1 = 2) \Rightarrow (A_5 = 0)$,

Rule 5) $(A_3 = 0, A_1 = 0) \Rightarrow (A_5 = 0)$,

Rule 6) $(A_3 = 2, A_1 = 1) \Rightarrow (A_5 = 0)$,

and the following correlation rules are “weak”:

Rule 7) $(A_3 = 0, A_1 = 1) \rightarrow (A_5 = 0)$,

Rule 8) $(A_3 = 2, A_1 = 1) \rightarrow (A_5 = 1)$,

Rule 9) $(A_3 = 0, A_1 = 0) \rightarrow (A_5 = 1)$,

Rule 10) $(A_3 = 1, A_1 = 0) \rightarrow (A_5 = 1)$.

Rule 11) $(A_3 = 0, A_1 = 2) \rightarrow (A_5 = 1)$,

Rule 12) $(A_3 = 2, A_1 = 2) \rightarrow (A_5 = 1)$,

To illustrate the above correlation rules, from Rule 1, if work-year is between 5 to 20, and age between 40 to 60, then the applicant with a high probability (0.9375) is non-jobless. From Rule 6, if work-year is over 20, and age between 40 to 60, then the applicant with a small probability (0.1111) is jobless, i.e., the applicant is of a high probability non-jobless.

In addition to the above correlation rules derived from direct parent-child dependencies, there are many correlation rules which can be acquired from indirect dependencies shown in the network in Figure 2. We give an example as follows to show how to get the correlation rule between A_8 and A_1, A_3.

Since

$$
\begin{aligned}
P(A_8|A_1, A_3) &= \frac{P(A_1, A_3, A_8)}{P(A_1, A_3)}, && (7) \\
P(A_1, A_3) &= P(A_1)P(A_3|A_1), && (8) \\
an\{A_1, A_3, A_8\} &= \{A_1, A_3, A_4, A_5, A_6, A_7, A_8\},
\end{aligned}
$$

in terms of Eq. (1) for the ancestral set,

$$
\begin{aligned}
P(A_1, A_3, A_8) &= \sum_{A_4, A_5, A_6, A_7} P(A_1, A_3, A_4, A_5, A_6, A_7, A_8) \\
&= \sum_{A_4, A_5, A_6, A_7} P(A_1)P(A_4)P(A_3|A_1)P(A_5|A_1, A_3) \\
&\quad \cdot P(A_6|A_4)P(A_7|A_1, A_3, A_5, A_6)P(A_8|A_5, A_7) && (9)
\end{aligned}
$$

The conditional probability for A_8 given A_1, A_3 can be obtained from the Eq. (7), (8) and (9), then the correlation rule between attribute A_8 and attributes A_1, A_3 can be got accordingly. In this paper, we only provide the formulae to obtain the classification and correlation rules from indirect dependencies.

These mined classification and correlation rules make sense. In the light of these rules, a bank manager will get a reference to target potential customers.

4.2 Mining Mushroom Data

The mushroom dataset was drawn from The Audubon Society Field Guide to North American Mushrooms. It involves 22 attributes, a class variable, and 8124 records. These attributes include population (A_1), habitat (A_2), cap-shape (A_3), cap-surface (A_4), cap-color (A_5), bruises? (A_6), odor

(A_7), gill-attachment (A_8), gill-spacing (A_9), gill-size (A_{10}), gill-color (A_{11}), stalk-shape (A_{12}), stalk-root (A_{13}), stalk-surface-above-ring (A_{14}), stalk-surface-below-ring (A_{15}), stalk-color-above-ring (A_{16}), stalk-color-below-ring (A_{17}), veil-type (A_{18}), veil-color (A_{19}), ring-number (A_{20}), ring-type (A_{21}), spore-print-color (A_{22}). The class variable (C) indicates that a mushroom is edible or poisonous. The order for all the attributes is adjusted and different from in the original database.

- It is assumed that we have the prior knowledge before learning that class variable C has all the attributes $\{A_1, A_2, ..., A_{22}\}$ as its possible parents. Each of $A_3, ..., A_{22}$ has possible parents $\{A_1, A_2\}$. Then the learning from the mushroom dataset is invoked with the K2.

- After the learning done, the result is that the class variable C only has attributes $\{A_5, A_{18}, A_{20}\}$ as its parents, and none of $A_3, ..., A_{22}$ has any parent. It means that other attributes either are independent of the class variable or have with it so weak causal relationships that the K2 cannot capture them, and each of $A_3, ..., A_{22}$ is either independent of A_1 and A_2 or has very weak causal relationship with them.

The belief network learned from mushroom data is shown in Figure 3. The isolated attributes are omitted.

Table 2 shows the conditional probabilities for class variable C with regard to its parents A_7, A_{20}, and A_{22}. These conditional probabilities are learned from the mushroom database with 8124 records. The others are omitted due to space constraint. In Table 2, attribute A_7 has values: 0(almond), 1(creosote), 2(foul), 3(anise), 4(musty), 5(none), 6(pungent), 7(spicy), 8(fishy); attribute A_{20} has values: 0(none), 1(one), 2(two); attribute A_{22} has values: 0(buff), 1(chocolate), 2(black), 3(brown), 4(orange), 5(green), 6(purple), 7(white), 8(yellow); the class variable C has values: 0(edible), 1(poisonous). As to the values of other attributes, please refer to the web site given earlier.

If we focus on the class variable C and mine the classification rules, meanwhile, the threshold $Min_\rho = 0.5$, the following rules can be obtained in terms of the belief network in Figure 3 and the Table 2:

Rule 1) $(A_7 = 3, A_{22} = 3, A_{20} = 1) \Rightarrow (C = 0)$,

Rule 2) $(A_7 = 0, A_{22} = 2, A_{20} = 1) \Rightarrow (C = 0)$,

Rule 3) $(A_7 = 5, A_{22} = 2, A_{20} = 1) \Rightarrow (C = 0)$,

Rule 4) $(A_7 = 3, A_{22} = 6, A_{20} = 1) \Rightarrow (C = 0)$,

Rule 5) $(A_7 = 5, A_{22} = 4, A_{20} = 1) \Rightarrow (C = 0)$,

Rule 6) $(A_7 = 6, A_{22} = 2, A_{20} = 1) \rightarrow (C = 0)$,

Rule 7) $(A_7 = 2, A_{22} = 1, A_{20} = 1) \rightarrow (C = 0)$,

Rule 8) $(A_7 = 1, A_{22} = 2, A_{20} = 1) \rightarrow (C = 0)$,

Rule 9) $(A_7 = 8, A_{22} = 7, A_{20} = 1) \rightarrow (C = 0)$,

Rule 10) $(A_7 = 2, A_{22} = 7, A_{20} = 1) \rightarrow (C = 0)$,

...

We briefly explain the above results, Rule 1 suggests that if odor is anise, spore-print-color brown, and ring-number one, then the outcome of classification for a mushroom with a very high probability (0.9950) is edible. Rule 6 says that if odor is pungent, spore-print-color black, and ring-number one, then the outcome of classification for a mushroom with a very small probability (0.0077) is edible, i.e., the mushroom is of a very high probability poisonous. These mined rules can provide a farmer the reference to judge whether a mushroom is edible or poisonous.

4.3 Summary

Since the K2 is a greedy search algorithm, there is no guarantee that a global maximum can be found. Here we only suppose that the results extracted with K2 are relatively reliable. We observed from Sections 4.1 and 4.2 that the following can be performed in learning a belief network by K2: if a user's conjecture is correct, the information in a database is complete, then the result of rule mining will be relatively reliable. If the user's conjecture is wrong, but the information in the database is complete, the result will still be relatively reliable as K2 learns from the database. That is, the learned belief networks can confirm whether a user's prior knowledge is correct or not. This is somehow beyond rule mining because the user's prior knowledge also can be corrected through the learned network. If the user's conjecture is correct, but the information in the database is incomplete, the result will be unreliable. In most situations, it is always deemed that the information in a database is almost complete. Lastly, if the user's conjecture is wrong, and the information in the database is incomplete, the result will be very unreliable. Overall, we give a general summary in Table 3.

Table 3 only serves as a general summary. Since the information in any database is not absolutely complete, and any learning algorithm itself is not perfect either, a learning algorithm will at times give spurious dependencies, have inaccurate parameter estimates, and make wrong predictions.

Many diverse methods for automated learning from data have been developed. Since it is impractical to compare all these methods, we shall only restrict our comparison to classification trees and neural networks that we believe are close to our method. A classification tree is usually restricted to representing the distribution on one variable of interest - the classification variable. With this constraint, however, classification trees can often represent compactly the attributes that influence the distribution of the classification variable. It is simple and efficient to apply a classification tree to perform classification.

However, belief networks can capture the probabilistic relationships among multiple variables, without the need to designate a classification variable. These networks provide a natural representation for capturing causal relationship among a set of variables [3]. In addition, inference algorithms exist for computing the probability of any subset of variables conditioned on the values of any other subset. In the worst case, however, these inference algorithms have a computational time complexity that is exponential in the size of the belief network. Nonetheless, for networks that are not densely connected, there exist exact inference algorithms such as [4, 15].

In addition to classification trees, in data mining community, the approach of neural networks for classification rule mining has been proposed [18]. As presented in this paper, however, the approach of belief networks can mine both classification rules and correlation rules. That is an advantage of approach of belief networks over that of neural networks in rule mining. In addition, the naive Bayes model is widely used for performing classification. It can compete with other classifiers such as backpropagation though its structure is less complex.

In this paper, we apply the K2, a Bayesian approach, to learning the structure of belief networks. The Bayesian approach takes prior knowledge as input and combines it with data to produce one or more belief networks. The most general form of Bayesian approach for learning the structure of belief networks is so called *marginal likelihood*. Let D be a database of cases, given a belief network structure m, the *marginal likelihood* function is given by

$$p(D|m) = \int p(D|m, \theta_m) p_m(\theta_m) d\theta_m, \tag{10}$$

where for the given model m, $p_m(\theta_m)$ is the prior distribution density for its parameter θ_m. $p(D|m)$ or its logarithm, can be used directly as a score function for comparing different models. Given a score function that evaluates the merit of a belief network structure, learning the qualitative part of belief networks from databases reduces to a search for one or more structures that have a high score. From Bayes' theorem, we have

$$p(m|D) = p(m)p(D|m)/p(D) \tag{11}$$

where p(D) is a normalization constant that does not depend upon structure. Thus, to determine the posterior distribution $p(m|D)$ for network structures, we need to compute the marginal likelihood of the data $p(D|m)$ for each possible structure.

In terms of Eq. (11), Eq. (2) is just the score function for the K2. Eq. (2) was derived based on the four assumptions in Section 3. The similar simple formula holds when the assumption 4 is relaxed to that the prior distribution density is *Dirichlet* and satisfies parameter independence. For the continuous variables arising from a Gaussian model with a conjugate prior, the similar formulae can be found as well in the case of complete data. In most other cases, however, the integral in Eq. (10) is not obtainable in closed form. Various approximation to this integral have been proposed [12]. One possible approximation to the log marginal likelihood is the Bayesian Information Criterion (BIC), but this doesn't require the evaluation of the prior distribution density $p_m(\theta_m)$, and is too crude to use except for extremely large samples. All the possible approximations to the log marginal likelihood serve as the score functions for the evaluation of the belief networks with incomplete data or/and more general prior distribution density. For the detailed discussion of the recent progress in the structural learning of belief networks, please refer to [2, 12].

5 From a Belief Language to Belief Networks

In [20], a language for the user's belief specification is proposed. The language is designed to represent user's prior knowledge in three different abstraction levels, in order that a user can iteratively increase the specificity of their beliefs for different needs in a discovery process.

5.1 A Belief Language

Here we will briefly introduce the belief language. Firstly, we bring in the following notations that are defined to represent the general structures of preferences and beliefs for classification and correlation rules.

$\{\}$ - represents a set of beliefs. $(Item_i)$ - represents a single pre-defined item or attribute i, as well as a single pre-defined value, "?" - represents the unknown variable for a user's query. $[ValRange]$ - represents range values for continuous attributes. $(class) = [ClassVal_j]$ - represents a single pre-defined class value in a dataset. $Corr \in \{\downarrow, \uparrow, ?\}$ - represents a correlation operation.

Level-1: Preference - Itemsets

1. incl$\{(Item_i)\}$ - inclusive itemsets

2. excl$\{(Item_i)\}$ - restrictive itemsets

Preference 1 represents an inclusive template. the set of items specified in an inclusive template is deemed to be "interesting" to the user. Preference 2 represents an exclusive template. The set of items specified in the exclusive template is deemed to be "uninteresting" to the user.

Level - 2: Preference - Rule Structure

1. $\{(Item_i)\} \rightarrow \{?\}$ - specifies antecedent(s) only

2. $\{?\} \rightarrow \{(Item_i)\}$ - specifies consequent(s) only

3. $\{?\} \rightarrow (class) = [ClassVal_j]$ - feature selection: classification task

4. $\{(Item_i)\ ?\ [ValRange]\} \rightarrow (Item_j)$ - correlation task

Preference 1 and 2 represent a specific preference for the return of correlation rules (if any) containing the specific itemsets in the antecedent and the consequent, respectively. Preference 3 represents a specific preference for the return of classification rules (if any) containing the specified class value. Preference 4 represents a specific preference for the return of correlation rules (if any) containing the continuous attributes (within a specific range) in the antecedent and the consequent respectively.

Level - 3: Belief - Rule

1. $\{(Item_i)\} \rightarrow \{(Item_j)\}$ - for a correlation task

2. $\{(Item_i)\} \rightarrow (class) = [ClassVal_j]$ - for a classification task

3. $\{(Item_i)\ Corr\ [ValRange]\} \rightarrow (Item_j)$ - for a correlation task

Beliefs 1 and 3 represent the conjectures or prior knowledge concerning an correlation pattern between itemset i and j. Belief 2 represents a conjecture or prior knowledge of the set of item i to be influential to the given classification outcome.

The belief language allows the user to incorporate his/her preferences and beliefs into the data mining process, so as to guide the discovery process and control the kind of knowledge that is returned. Based on the belief language, a pre-processing phase can be included into the discovery process to eliminate irrelevant attributes at the earliest opportunity [5].

5.2 Using Belief Networks

We now will show that the belief language can be represented easily by belief networks. Hereafter, it is assumed that a node (variable) in belief networks be an item in the belief language.

1. For level-1, adding nodes in belief networks corresponds to inclusive itemsets, and deleting nodes does to restrictive itemsets.

2. For level-2, specifying parent node(s) in belief networks is equivalent to Rule1 in belief language in level-2, specifying child node(s) is to Rule2, and specifying the value of a node, which represents a class, is to Rule 3.

   As for the Rule 4 in level-2 belief language, we can represent it by a conditional probability, $f(t, x) = P(A = t|B = x), x \in [ValRange]$. A and B denote nodes, respectively.

3. For level-3, checking the existence of an arc between two nodes in belief networks is equal to Rule 1 in belief language in level-3, and specifying a set of parent nodes for a child node is to Rule 2.

   As to the Rule 3 in level-3 belief language, As the above description, $f(t, x) = P(A = t|B = x), x \in [ValRange]$. If $f(t, x) > 0$, node A has positive correlation with node B. If $f(t, x) = 0$, A has no correlation with B. The number of $f(t, x)$ can be determined by learning belief network from databases.

Learning belief networks from databases can refine the user's prior knowledge. The belief language in [20] can only specify the qualitative part based on user's prior knowledge, but can't refine it by learning from real databases. In addition, the belief language can only reflect user's subjective judgments, and can't efficiently extract objective knowledge from real databases. By learning a belief network from database, the "likelihood" for classification and correlation rules can be depicted accurately in terms of the learned conditional probabilities. The belief language can only give the rough estimation of the "likelihood" based on user's prior knowledge.

6 Concluding Remarks

In this article, we present a belief network approach to the mining of correlation and classification rules from databases with prior knowledge. An algorithm of learning belief networks is presented. The study of two benchmark databases is conducted using the learning algorithm. The study proves that a user's prior knowledge or conjectures can be easily specified with the directed acyclic graph language of belief networks. The classification and correlation rules mined from databases can be represented based on the structure and conditional probabilities of the networks learned from databases. By the new approach, the representation of correlation rules is generalized from an existing alternative so that the generalized correlation rules can capture the relationship among more than two attribute variables. The experiments show that a user's prior knowledge or conjectures can be confirmed or disconfirmed by learning belief networks from databases. This is somehow beyond rule mining. The belief networks method for rule mining takes advantage of machine learning, so the more complete the information in a database, the more accurate the rules mined. We also extend our method to replace a belief language for the specification of a user's preferences. The advantages of our method over the belief language are pointed out. Several relevant issues should be noted. First, whether the information in a database is complete or not heavily affects the accuracy of the rules mined by learning belief networks from databases. The databases with complete information are preferred. Second, the result of learning belief networks depends on the learning algorithm itself as well. An algorithm often has some assumptions. For instance, K2 requires an ordering on the domain variables prior to learning. For a different ordering, some of the conditional independencies may be hidden, it also slightly affects the result of learning. Thus the combination of a few methods of learning belief networks to build an interactive learning shell system will allow several learning methods to complement each other.

Acknowledgment

We thank the anonymous reviewer for insightful comments. The work was partially done while the first author was with the Program for Research in Intelligent Systems (PRIS), School of Computing, National University of Singapore.

References

[1] G. Cooper and E. Herskovitz. A bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

[2] R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer-Verlag, New York, 1999.

[3] S. Crawford and R. Fung. An analysis of two probabilistic model induction techniques. In *Proceedings of the Third International Workshop on AI and Statistics*, 1991.

[4] B. D'Ambrosio. Inference in bayesian networks. *AI Magazine*, 20(2), 1999.

[5] M. Dash and H. Liu. Feature selection for classification. *Journal of Intelligent Data Analysis*, 1(3), 1997.

[6] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview. In U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, 1996.

[7] N. Friedman. Learning belief networks in the presence of missing values and hidden variables. In *Proceedings of the International Conference on Machine Learning*, pages 125–133, 1997.

[8] N. Friedman. The bayesian structural em algorithm. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 129–138. Morgan Kaufmann, 1998.

[9] P. Haddawy. An overview of some recent developments in bayesian problem-solving techniques. *AI Magazine*, 20(2), 1999.

[10] D. Heckerman. A bayesian approach to learning causal networks. Technical Report MSR-TR-95-04, Microsoft Research, 1995.

[11] D. Heckerman. Bayesian networks for data mining. *Data Mining and Knowledge Discovery*, 1(1), 1997.

[12] D. Heckerman. A tutorial on learning with bayesian networks. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 301–354. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.

[13] D. Heckerman and D. Geiger. Likelihoods and parameter priors for bayesian networks. Technical Report MSR-TR-95-54, Microsoft Research, 1995.

[14] D. Heckerman, D. Geiger, and D. Chichering. Learning bayesian network: The combination of knowledge and statistical data. Technical Report MSR-TR-94-09, Microsoft Research, 1994.

[15] M. Henrion. An introduction to algorithms for inference in belief nets. In M. Henrion, R. D. Shachter, L. N. Kanal, and J. Lemmer, editors, *Uncertainty in artificial intelligence 5*. Amsterdam: North-Holland, 1990.

[16] F. Hussain, H. Liu, C. L. Tan, and M. Dash. Discretization: an enabling technique. Technical Report TRC6/99, School of Computing, National University of Singapore, 1999.

[17] P. J. Krause. Learning probabilistic networks. Technical report, Philips Research Laboratories, UK, Manuscript.

[18] H. Lu, R. Setiono, and H. Liu. Effective data mining using neural networks. *IEEE Transaction on Knowledge and Data Engineering*, 8(6), 1996.

[19] R. Neal. Connectionist learning of belief networks. *Artificial Intelligence*, 40:185–234, 1989.

[20] K. S. Ng and H. Liu. A belief-based approach to knowledge discovery in databases. In *Proceedings of PRICAI'98 Workshop on Data Mining*, Nov 1998.

[21] B. Padmanabhan and A. Tuzhilin. A belief-driven method for discovering unexpected patterns. In *Proceeding of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD 98)*, 1998.

[22] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, CA, 1988.

[23] Y. Peng and Z. Zhou. A neural network learning method for belief networks. *International Journal of Intelligent Systems*, 11:893–915, 1996.

[24] Y. Peng, Z. Zhou, and S. Cho. Constructing belief networks from realistic data. *International Journal of Intelligent Systems*, 14:671–695, 1999.

[25] M. Ramoni and P. Sebastiani. Learning bayesian networks from incomplete databases. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 401–408. Morgan Kaufmann, 1995.

[26] P. Sebastiani, M. Ramoni, and A. Crea. Profiling your customers using bayesian networks. *SIGKDD Explorations*, 1(2):91–96, 2000.