Mining High Order Decision Rules

Y.Y. Yao

Department of Computer Science, University of Regina
Regina, Saskatchewan, Canada S4S 0A2
e-mail: yyao@cs.uregina.ca

Abstract. We introduce the notion of high order decision rules. While a standard decision rule expresses connections between attribute values of the *same* object, a high order decision rule expresses connections of *different* objects in terms of their attribute values. An example of high order decision rules may state that "if an object x is related to another object y with respect to an attribute a, then x is related to y with respect to another attribute b." The problem of mining high order decision rules is formulated as a process of finding connections of objects as expressed in terms of their attribute values. In order to mine high order decision rules, we use relationships between values of attributes. Various types of relationships can be used, such as ordering relations, closeness relations, similarity relations, and neighborhood systems on attribute values. The introduction of semantics information on attribute values leads to information tables with added semantics. Depending on the decision rules to be mined, one can transform the original table into another information table, in which each new entity is a pair of objects. Any standard data mining algorithm can then be used. As an example to illustrate the basic idea, we discuss in detail the mining of ordering rules.

1 Introduction

In machine learning and data mining, one often uses an attribute-value language to represent individual objects and mined knowledge [9–11,13,14,20]. Each object is represented by the values of a set of attributes. The knowledge mined from a dataset is represented in the form of rules. At least two types of rules can be identified [5]. The first type, called type 1 rules in this paper, is exemplified by decision rules. A type 1 rule states that "if the value of an object is v_a on attribute a, then the value of the object is v_b on attribute b". By pooling together many type 1 rules, we can obtain the second type rules, called type 2 rules in this paper. An example of type 2 rules is a functional (or data) dependency rule. A type 2 rule states that "if two objects have the same value on attribute a, then they have the same value on attribute b". The two types of rules represent different levels of knowledge derivable from a dataset. While type 1 rules focus on a *single* object, type 2 rules focus on a *pair* of objects.

One objective of this paper is to provide a systematic study of type 2 rules, which have not received much attention. We also refer to type 2 rules as high order decision rules representing a higher level of knowledge. The phrase "high order rules" expresses our intuitive understanding and interpretation of the type 2 rules. A special kind of high order rules called ordering rules has been studied in [15,21,22]. By further generalizing the results, we introduce various high order rules. This is done by generalizing the trivial equality relation on attribute values used in defining functional (or data) dependency rules. For example, by using a similarity relation, we derive a weaker functional (or data) dependency rule: "if two objects have *similar* values on one set of attributes, then they have *similar* values on another set of attributes". When preference (ordering) relations are used, an ordering rule is obtained: "if an object is ranked ahead of another object according to one set of attributes, then the pair are ranked in the same way with respect to another set of attributes" [15,21].

Another objective of this paper is to study algorithms for mining high order decision rules. For this purpose, we need to express relationships between attribute values. Information tables with added semantics are used [19]. The relationships between attribute values are used to infer relationships between objects. Depending on the high order decision rules to be mined, one can transform the original table into another information table, in which each new entity is a pair of objects. After the transformation, any standard data mining algorithm can be used. The basic idea is illustrated by showing how ordering rules can be mined [15,21].

2 Motivations

Two essential tasks in machine learning and data mining are the representation of objects and the identification of forms and types of knowledge to be mined. An attribute-value language provides a simple and useful tool for dealing with the two tasks.

Objects are represented in terms of their values on a set of attributes. More specifically, information about objects is summarized in an information table defined by [11]:

$$S = (U, At, \{V_a \mid a \in At\}, \{I_a \mid a \in At\}),$$

where

$$\begin{aligned}
&U \text{ is a finite nonempty set of objects,}\\
&At \text{ is a finite nonempty set of attributes,}\\
&V_a \text{ is a nonempty set of values for } a \in At,\\
&I_a : U \longrightarrow V_a \text{ is an information function.}
\end{aligned}$$

Each information function I_a is a total function that maps an object of U to exactly one value in V_a. It can be conveniently given in a tabular form, where the rows correspond to objects of the universe, the columns correspond to a set of attributes, and each cell is the value of an object with respect to an attribute.

Knowledge derivable from an information table is commonly represented in the form of rules, such as decision rules [10,14], association rules [1], functional dependency rules [2], and data dependency rules [11]. Roughly speaking, those rules show the connections between attributes, which are normally characterized by the problem of determining the values of one set of attributes based on the values of another set of attributes. Depending on the meanings and forms of rules, one may classify rules in many ways. A clear classification of rules is useful for the understanding of the basic tasks of machine learning and data mining.

Rules can be classified into two groups in terms of their directions, *one-way* and *two-way* connections, and further classified into two levels in terms of their applicability, *local* and *global* connections [18,19,24]. A one-way connection shows that the values of one set of attributes determine the values of another set of attributes, but does not say anything of the reverse. A two-way connection is a combination of two one-way connections, representing two different directions of connection. A local connection is characterized by a type 1 rule and shows the relationship between *one* specific combination of values on one set of attributes and *one* specific combination of values on another set of attributes. A global connection is characterized by a type 2 or high order rule and shows the relationships between *all* combinations of values on one set of attributes and *all* combinations of values on another set of attributes.

Finding local one-way connection is one of the main tasks of machine learning and data mining [9,11,13,14]. The well known association rules [1], which state that the presence of one set of items implies the presence of another set of items, is a special kind of local one-way connections. Decision rules from decision tree learning algorithms [14] are also examples of local one-way connections. Functional dependency in relational databases [2] is a typical example of global one-way connections. Attribute (data) dependency studied in the theory of rough sets [11] is another example of global one-way connections.

Without additional semantic relationships between attribute values, one can only use the trivial equality relation. In this case, a local one-way connection is expressed by a type 1 rule of the form: for $x \in U$,

$$I_a(x) = v_a \Rightarrow I_b(x) = v_b. \tag{1}$$

A global one-way connection is expressed by a type 2 or high order rule of the form: for $(x, y) \in U \times U$,

$$I_a(x) = I_a(y) \Rightarrow I_b(x) = I_b(y). \tag{2}$$

We have used only a single attribute in expressing the conditions in a rule. In the next section, it will be shown how to construct more complex conditions by using many attributes. A closer examination of local and global connections shows the following difference. A local rule states knowledge about *one* object. A local one-way rule shows that if the object has a specific value on one set of attributes, then it will have specific value on another set of attributes. On the other hand, a global rule states knowledge about a *pair* of objects. A global one-way rule suggests that if a pair of objects have the *same* value on one set of attributes, then they will have the *same* value on another set of attributes. This observation motivates the present study. Since a global rule states the connection of attributes with respect to pairs of objects, or equivalently, the connection between object pairs in terms of attributes, it is referred to as a high order rule.

Majority research in machine learning and data mining has been focused on discovering type 1 rules. There is relatively less work on type 2 or high order rules. In addition, the equality relation is commonly used in expressing rules. Semantically speaking, high order rules explain why two objects are connected based on the relationships between their attribute values. Type 1 rules are useful for classification. High order rules are useful for the study of relationships between objects. Many real world problems are related to high order rule. For example, in multi-attribute decision making, ranking of universities, ranking of consumer products, one is interested in finding out the correlation of the overall ranking and the individual rankings induced by individual attributes [6–8,12,15,21]. Weak or fuzzy functional dependency characterized by similarity is another example [4]. There is a need for a systematic study of high order rules. Moreover, when semantics relationships between attribute values are introduced, one may obtain various classes of high order decision rules.

3 Mining High Order Decision Rules

For the purpose of mining rules, we add a binary relation R_a for each attribute $a \in At$, a language $\mathcal{L}_0$ and a language $\mathcal{L}_1$ to an information table. The binary relation is used to define relationships between values of an attribute, with the equality relation as a special case. While the language $\mathcal{L}_0$ is used to define conditions in type 1 rules, the language $\mathcal{L}_1$ is used to describe conditions in high order rules. The details of language $\mathcal{L}_0$ can be found in [20]. We will only discuss the language $\mathcal{L}_1$.

In the language $\mathcal{L}_1$, an atomic formula is $\mathcal{R}_a$ for an attribute $a \in At$. If ϕ and ψ are formulas, so are $\neg\phi$, $\phi \wedge \psi$, $\phi \vee \psi$, $\phi \rightarrow \psi$, and $\phi \equiv \psi$. The semantics of the language $\mathcal{L}_1$ can be defined in the Tarski's style through the notions of a model and satisfiability. The model is an information table S with added binary relations, which provides interpretations for symbols and formulas of $\mathcal{L}_1$. The satisfiability of a formula ϕ by a pair of objects (x, y), written $(x, y) \models_S \phi$ or in short $(x, y) \models \phi$ if S is understood, is given by the conditions:

$$
\begin{array}{ll}
(0) & (x,y) \models \mathcal{R}_a \text{ iff } I_a(x) R_a I_a(y), \\
(1) & (x,y) \models \neg\phi \text{ iff } \text{ not } (x,y) \models \phi, \\
(2) & (x,y) \models \phi \wedge \psi \text{ iff } (x,y) \models \phi \text{ and } (x,y) \models \psi, \\
(3) & (x,y) \models \phi \vee \psi \text{ iff } (x,y) \models \phi \text{ or } (x,y) \models \psi, \\
(4) & (x,y) \models \phi \rightarrow \psi \text{ iff } (x,y) \models \neg\phi \vee \psi, \\
(5) & (x,y) \models \phi \equiv \psi \text{ iff } (x,y) \models \phi \rightarrow \psi \text{ and } (x,y) \models \psi \rightarrow \phi.
\end{array}
$$

For formula ϕ, the set $m_S(\phi)$ defined by:

$$m_S(\phi) = \{(x,y) \in U \times U \mid (x,y) \models \phi\}, \tag{3}$$

is called the meaning set of ϕ in S. If S is understood, we simply write $m(\phi)$. Obviously, the properties hold:

$$
\begin{array}{ll}
\text{(m1).} & m(\mathcal{R}_a) = \{(x,y) \in U \times U \mid I_a(x) R_a I_a(y)\}, \\
\text{(m2).} & m(\neg\phi) = U \times U - m(\phi), \\
\text{(m3).} & m(\phi \wedge \psi) = m(\phi) \cap m(\psi), \\
\text{(m4).} & m(\phi \vee \psi) = m(\phi) \cup m(\psi).
\end{array}
$$

A pair $(x,y) \in m(\phi)$ is said to satisfy the expression ϕ. The formula ϕ can be viewed as the description of the set of object pairs $m(\phi)$, and each object pair in $m(\phi)$ as an instance of the concept given by ϕ.

For two subsets of attributes $A, B \subseteq At$ with $A \cap B = \emptyset$, let ϕ and ψ be formulas constructed from attributes in A and B, respectively. A high order decision rule can be expressed in the form, $\phi \Rightarrow \psi$. In many studies of machine learning and data mining, a rule is usually paraphrased by an if-then statement. This interpretation suggests a kind of cause and effect relationship between ϕ and ψ, although such a cause and effect relationship does exist. We therefore need to closely look at the meaning and interpretation of rules.

An immediately interpretation of rules is through logic implication. That is, the symbol$\Rightarrow$ is interpreted as the logical implication $\rightarrow$ of the language $\mathcal{L}_1$. In practice, such an interpretation may be too restrictive to be useful. Moreover, one may only be interested in the satisfiability of ψ under the condition that the object pair (x,y) satisfies ϕ. Under those situations, probabilistic interpretations may be more appropriate. Many probabilistic interpretations can be found in [23]. We choose to use two measures called accuracy and coverage defined by [16]:

$$
\begin{aligned}
accuracy(\phi \Rightarrow \psi) &= \frac{|m(\phi \wedge \psi)|}{|m(\phi)|}, \\
coverage(\phi \Rightarrow \psi) &= \frac{|m(\phi \wedge \psi)|}{|m(\psi)|},
\end{aligned} \tag{4}
$$

where $|\cdot|$ denotes the cardinality of a set. While the accuracy reflects the correctness of the rule, the coverage reflects the applicability of the rule. If $accuracy(\phi \Rightarrow \psi) = 1$, relationships between objects induced by ϕ would determine the relationship induced by ψ. We thus have a strong association between the two relationships induced by ϕ and ψ. A smaller value of *accuracy* indicates a weaker association. A high order rule with higher coverage suggests that relationships of more pairs of objects can be derived from the rule. The accuracy and coverage are not independent of each other, as both are related to the quantity $|m(\phi \wedge \psi)|$. It is desirable for a rule to be accurate as well as to have a high degree of coverage. In general, one may observe a trade-off between accuracy and coverage. A rule with higher coverage may have a lower accuracy, while a rule with higher accuracy may have a lower coverage.

In theory, if we have a finite set of attributes and each attribute has a finite set of values, high order rules can be derived from an information table by searching all pairs of formulas. The meaning sets $m(\phi)$, $m(\psi)$ and $m(\phi \wedge \psi)$ can be used to eliminate those rules that are not interesting. In what follows, we present a transformation method so that any existing machine learning and data mining algorithm can be directly applied.

For each object pair (x, y), a formula is either satisfied or not satisfied. The relationship between objects induced by an attribute a can be easily represented by the satisfiability of the atomic formula $\mathcal{R}_a$. Thus, if we create a new universe consisting of pair of objects, we can produce a binary information table that preserve the relationships of objects induced by binary relation relations. A binary information table is defined as follows:

$$I_a(x, y) = \begin{cases} 1, & I_a(x) R_a I_a(y), \\ 0, & \text{not } [I_a(x) R_a I_a(y)]. \end{cases} \tag{5}$$

The values 1 and 0 show that the pair satisfy the atomic expression $\mathcal{R}_a$ and does not satisfy the atomic expression, respectively. Statements in the original information table expressed in a language $\mathcal{L}_1$ be translated into equivalent statements expressed in a language $\mathcal{L}_0$ in the binary information table, and vice versa. More specifically, the atomic formula $\mathcal{R}_a$ corresponds to an atomic formula $I_a(x, y) = 1$. The formula $\neg\mathcal{R}_a$ corresponds to the formula $\neg(I_a(x, y) = 1)$, or equivalently, $I_a(x, y) = 0$.

The translation of the original table into a binary information table is a crucial step for mining high order rules. With the translation, high order rules of the original table reduce to standard decision rules of the binary table. Consequently, any standard machine learning and data mining algorithms can be used to mine high order rules.

For clarity and simplicity, we have only considered a very simply language $\mathcal{L}_1$ in which there is only one relation on values of each attribute. In general, one may use many binary relations on attribute values to represent different types of relationships. Accordingly, a more powerful language is needed. The arguments presented so far can be easily applied with slight modification.

4 Mining Ordering Rules: an Illustrative Example

In real world situations, we may face many problems that are not simply classification [3,12]. One such type of problems is the ordering of objects. Two familiar examples of ordering problems are the ranking of universities and the ranking of the consumer products produced by different manufactures. In both examples, we have a set of attributes that are used to describe the objects under consideration, and an overall ranking of objects. Consider the example of ranking consumer products. Attributes may be the price of products, warranty of products, and other information. The values of a particular attribute, say the price, naturally induce an ordering of objects. The overall ranking of products may be produced by the market shares of different manufactures. The orderings of objects by attribute values may not necessarily be the same as the overall ordering of objects.

The problem of mining ordering rules can be stated as follows. There is a set of objects described by a set of attributes. There is an order relation $\succ_a$ on values of each attribute $a \in At$, and there is also an overall ordering of objects $\succ_o$. The overall ordering may be given by experts or obtained from other information, either dependent or independent of the orderings of objects according to their attribute values. We are interested in mining the association between the overall ordering and the individual orderings induced by different attributes. More specifically, we want to derive ordering rules exemplified by the statement that "if an object x is ranked ahead of another object y on an attribute a, then x is ordered ahead of y".

Order relations are special types of relations that induce orderings on the set of objects. An ordering of values of a particular attribute a naturally induces an ordering of objects, namely, for $x, y \in U$:

$$x \succ_a y \Longleftrightarrow I_a(x) \succ_a I_a(y), \tag{6}$$

where $\succ_a$ also denotes an order relation on U induced by the attribute a. An object x is ranked ahead of another object y according to an attribute a if and only if the value of x on a is ranked ahead of the value of y on a. The order relation on objects has exactly the same properties as that of the order relation on attribute values. For this reason, we have used the same symbol to denote both relations. Typically, an order relation should satisfy certain conditions. We consider the two properties:

$$\begin{aligned} \text{Asymmetry}: \ & x \succ y \Longrightarrow \neg(y \succ x), \\ \text{Negative transitivity}: \ & [\neg(x \succ y), \neg(y \succ z)] \Longrightarrow \neg(x \succ z). \end{aligned}$$

An order relation satisfying these properties is called a weak order [17]. An important implication of a weak order is that the relation,

$$x \sim y \Longleftrightarrow [\neg(x \succ y), \neg(y \succ x)], \tag{7}$$

is an equivalence relation. For two objects, if $x \sim y$ we say x and y are indiscernible by $\succ$. The equivalence relation $\sim$ induces a partition $U/\sim$ on U, and an order relation on $U/\sim$ can be defined by:

$$[x]_\sim \succ^* [y]_\sim \Longleftrightarrow x \succ y, \tag{8}$$

where $[x]_\sim$ is the equivalence class containing x. Moreover, $\succ^*$ is a linear order [17]. Any two distinct equivalence classes of $U/\sim$ can be compared. It is therefore possible to arrange the objects into levels, with each level consisting of indiscernible objects defined by $\succ$. For a weak order, $\neg(x \succ y)$ can be written as $y \succeq x$ or $x \preceq y$, which means $y \succ x$ or $y \sim x$. For any two objects x and y, we have either $x \succ y$ or $y \succeq x$, but not both.

In the subsequent discussion, we assume that all order relations are weak orders. For simplicity, we also assume that there is a special attribute, called decision attribute. The ordering of objects by the decision attribute is denoted by $\succ_o$ and is called the overall ordering of objects.

By making use of the physical meaning of order relations, we can re-express ordering rules in a easy to read form. Consider an ordering rule,

$$\succ_a \wedge \neg \succ_b \quad \Rightarrow \quad \succ_c . \tag{9}$$

It can be re-expressed as,

$$x \succ_a y \wedge x \preceq_b y \Rightarrow x \succ_c y. \tag{10}$$

The rule suggests that the ordering of objects by c is determined by the ordering of objects by a and b. For two arbitrary objects x and y, if x is ranked ahead of y by a, and at the same time, x is not ranked ahead of y by b, then x is ranked ahead of y by c.

We illustrate the ideas developed so far by a simple example. Consider the information table of five products [21]:

	Size	Warranty	Price	Weight	Overall
1	middle	3 years	$200	heavy	best
2	large	3 years	$300	very heavy	good
3	small	3 years	$300	light	good
4	small	3 years	$250	very light	better
5	small	2 years	$200	very light	good

$\succ_{\text{Size}}$: small $\succ_{\text{Size}}$ middle $\succ_{\text{Size}}$ large,
$\succ_{\text{Warranty}}$: 3 years $\succ_{\text{Warranty}}$ 2 years,
$\succ_{\text{Price}}$: \$200 $\succ_{\text{Price}}$ \$250 $\succ_{\text{Price}}$ \$300,
$\succ_{\text{Weight}}$: very light $\succ_{\text{Weight}}$ light $\succ_{\text{Weight}}$ heavy $\succ_{\text{Weight}}$ very heavy,
$\succ_{\text{Overall}}$: best $\succ_{\text{Overall}}$ better $\succ_{\text{Overall}}$ good.

The order relations induces the following orderings of products:

$$\begin{aligned}
\succ_{\text{Size}}: &\quad [3,4,5] \succ^*_{\text{Size}} [1] \succ^*_{\text{Size}} [2],\\
\succ_{\text{Warranty}}: &\quad [1,2,3,4] \succ^*_{\text{Warranty}} [5],\\
\succ_{\text{Price}}: &\quad [1,5] \succ^*_{\text{Price}} [4] \succ^*_{\text{Price}} [2,3],\\
\succ_{\text{Weight}}: &\quad [4,5] \succ^*_{\text{Weight}} [3] \succ^*_{\text{Weight}} [1] \succ^*_{\text{Weight}} [2],\\
\succ_{\text{Overall}}: &\quad [1] \succ^*_{\text{Overall}} [4] \succ^*_{\text{Overall}} [2,3,5].
\end{aligned}$$

Examples of formulas and their meaning sets are given by:

$$\begin{aligned}
&m(\succ_{\text{Size}}) = \{(1,2),(3,1),(3,2),(4,1),(4,2),(5,1),(5,2)\},\\
&m(\succ_{\text{Price}}) = \{(1,2),(1,3),(1,4),(4,2),(4,3),(5,2),(5,3),(5,4)\},\\
&m(\succ_{\text{Overall}}) = \{(1,2),(1,3),(1,4),(1,5),(4,2),(4,3),(4,5)\},\\
&m(\succ_{\text{Size}} \wedge \succ_{\text{Overall}}) = \{(1,2),(4,2)\},\\
&m(\succ_{\text{Price}} \wedge \succ_{\text{Overall}}) = \{(1,2),(1,3),(1,4),(4,2),(4,3)\},\\
&m(\succ_{\text{Size}} \wedge \succ_{\text{Price}}) = \{(1,2),(4,2),(5,2)\},\\
&m(\succ_{\text{Size}} \wedge \succ_{\text{Price}} \wedge \succ_{\text{Overall}}) = \{(1,2),(4,2)\}.
\end{aligned}$$

For a rule, $\succ_{\text{Size}} \Rightarrow \succ_{\text{Overall}}$, namely, $x \succ_{\text{Size}} y \Rightarrow x \succ_{\text{Overall}} y$, its accuracy and coverage are:

$$\begin{aligned}
accuracy(\succ_{\text{Size}} \Rightarrow \succ_{\text{Overall}}) &= 2/7,\\
coverage(\succ_{\text{Size}} \Rightarrow \succ_{\text{Overall}}) &= 2/7.
\end{aligned}$$

One may conclude that the Size does not tell us too much information about the overall ranking in terms of both accuracy and coverage. The accuracy and coverage of the rule, $\succ_{\text{Price}} \Rightarrow \succ_{\text{Overall}}$, namely, $x \succ_{\text{Price}} y \Rightarrow x \succ_{\text{Overall}} y$, are:

$$\begin{aligned}
accuracy(\succ_{\text{Price}} \Rightarrow \succ_{\text{Overall}}) &= 5/8,\\
coverage(\succ_{\text{Price}} \Rightarrow \succ_{\text{Overall}}) &= 5/7.
\end{aligned}$$

In terms of both measures, the new rule is better. In other words, the Price tells us more about the overall ranking. By combining both Size and Price, we have another rule, $\succ_{\text{Size}} \wedge \succ_{\text{Price}} \Rightarrow \succ_{\text{Overall}}$, namely, $x \succ_{\text{Size}} y \wedge x \succ_{\text{Price}} y \Rightarrow x \succ_{\text{Overall}} y$, and

$$\begin{aligned}
accuracy(\succ_{\text{Size}} \wedge \succ_{\text{Price}} \Rightarrow \succ_{\text{Overall}}) &= 2/3,\\
coverage(\succ_{\text{Size}} \wedge \succ_{\text{Price}} \Rightarrow \succ_{\text{Overall}}) &= 2/7.
\end{aligned}$$

The new rule increases the accuracy, but decreases the coverage. In fact, the third rule is more specific than the first two rules.

It will be an easy task to transform the information table into a binary information. In the binary information table, the conditional $x \succ_a y$ is replaced by $I_a(x, y) = 1$. The meaning sets, accuracy and coverage of rules can be similarly defined [15,21].

5 Conclusion

A functional dependency rule states that if the values of a pair of objects are the same on one set of attributes, their values are the same on another set of attributes. A weak or fuzzy functional dependency rule states that if the values of a pair of objects are similar on one set of attributes, their values are similar on another set of attributes. An ordering rule suggests that if an object is ranked ahead of another object according to one attribute, then the same ranking of the two objects is obtained by another attribute. All these rules show the connections of objects based on their values on two sets of attributes. They may be considered as special cases of high order rules introduced in this paper, which state that if two objects are related according to one set of attributes, they are related based on another set of attributes. The relatedness of objects are modeled by a binary relation, or a group of binary relations, on the values of each attribute.

Information tables with added semantics are used to represent individual objects and relationships between values of attributes. A language is defined with respect to an information table, in which various concepts can be interpreted in Tarski's style. In particular, each formula of the language is interpreted as a set of object pairs called the meaning set. High order rules representing connections of two formulas can be interpreted in terms of their meaning sets.

The main contribution of this paper is the introduction of the notion of high order decision rules which represent a higher level knowledge than the standard decision rules, and the formulation of the problem of mining high order rules. Furthermore, we suggest that this problem can be reduced to standard machine leaning and data mining problems by a simple transformation method. Consequently, one can directly apply any existing data mining algorithms for mining ordering rules.

References

1. Agrawal, R., Imielinski, T., and Swami, A. Mining association rules between sets of items in large databases, *Proceedings of the ACM SIGMOD International Conference on the Management of Data*, 207-216, 1993.
2. Bell, A. Discovery and maintenance of functional dependencies by independencies, *Proceedings of KDD-95*, 27-32, 1995.
3. Cohen, W.W., Schapire, R.E. and Singer, Y. Learning to order things, *Journal of Artificial Intelligence Research*, **10**, 243-270, 1999.
4. Cubero, J.C., Medina, J.M., Pons, O. and Vila, M.A. Discovering Fuzzy Functional Dependencies in Databases, *Proceedings of EUFIT'96*, 811-815, 1996.
5. Gediga, G. and Düntsch, I. Rough approximation quality revisited, *Artificial Intelligence*, **132**, 219-234, 2001.
6. Greco, S., Matarazzo, B., and Slowinski, R. Rough approximation of a preference relation by dominance relations, *European Journal of Operational Research* **117**, 63-83, 1999.

7. Greco, S., Matarazzo, B., and Slowinski, R. Extension of the rough set approach to multicriteria decision support, *INFOR* **38**, 161-195, 2000.
8. Greco, S., Matarazzo, B., and Slowinski, R. Rough set theory for multicriteria decision analysis, *European Journal of Operational Research* **129**, 1-47, 2001.
9. Michalski, R.S., Carbonell, J.G. and Mitchell, T.M. (Eds.), *Machine Learning*, Tioga, 1983.
10. Mitchell, T.M. *Machine Learning*, WCB/McGraw-Hill, Boston, 1997.
11. Pawlak, Z. *Rough Sets, Theoretical Aspects of Reasoning about Data*, Kluwer Academic Publishers, Boston, 1991.
12. Pawlak, Z., Slowinski, R. Rough set approach to multi-attribute decision analysis, *European Journal of Operational Research*, **72**, 443-359, 1994.
13. Polkowski, L. and Skowron, A. (Eds.), *Rough Sets in Knowledge Discovery 1,2*, Physica-Verlag, Heidelberg, 1998.
14. Quinlan, J.R. *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, 1993.
15. Sai, Y., Yao, Y.Y. and Zhong, N. Data Analysis and Mining in Ordered Information Tables, *Proceedings of 2001 IEEE Conference on Data Mining*, 497-504, 2001.
16. Tsumoto, S. Automated discovery of plausible rules based on rough sets and rough inclusion, *Proceedings of PAKDD'99*, 210-219, 1999.
17. Yao, Y.Y. Measuring retrieval effectiveness based on user preference of documents, *Journal of the American Society for Information Science*, **46**, 133-145, 1995.
18. Yao, Y.Y. Information-Theoretic Measures for Knowledge Discovery and Data Mining, Manuscript, 1999.
19. Yao, Y.Y. Information tables with neighborhood semantics, in: *Data Mining and Knowledge Discovery: Theory, Tools, and Technology II*, Dasarathy, B.V. (Ed.), The International Society for Optical Engineering, Bellingham, Washington, 108-116, 2000.
20. Yao, Y.Y. Modeling data mining with granular computing, *Proceedings of the 25th Annual International Computer Software and Applications Conference*, 638-643, 2001.
21. Yao, Y.Y. and Sai, Y., Mining ordering rules using rough set theory, *Bulletin of International Rough Set Society*, **5**, 99-106, 2001.
22. Yao, Y.Y. and Sai, Y., On mining ordering rules, in: *New Frontiers in Artificial Intelligence, Lecture Notes in Computer Science 2253*, Terano, T., Nishida, T., Namatame, A., Tsumoto, S., Ohsawa, Y. and Washio, T. (Eds.), Springer, Berlin, 316-321, 2001.
23. Yao, Y.Y. and Zhong, N. An analysis of quantitative measures associated with rules, *Proceedings of PAKDD'99*, 479-488, 1999.
24. Yao, Y.Y. and Zhong, N. On association, similarity and dependency of attributes, *Proceedings of PAKDD'00*, 2000.