

Action Rules Discovery, a new simplified strategy

Zbigniew W. Raś^{1,2} and Agnieszka Dardzińska³

¹ UNC-Charlotte, Department of Computer Science, Charlotte, N.C. 28223, USA

² Polish Academy of Sciences, Institute of Computer Science, Ordona 21, 01-237
Warsaw, Poland; e-mail: ras@uncc.edu

³ Bialystok Technical Univ., Dept. of Mathematics, ul. Wiejska 45A, 15-351
Bialystok, Poland; e-mail: adardzin@uncc.edu

Abstract. A new strategy for discovering action rules (or interventions) is presented in this paper. The current methods [14], [12], [8] require to discover classification rules before any action rule can be constructed from them. Several definitions of action rules [8], [13], [9], [3] have been proposed. They differ in the generality of their classification parts but they are always constructed from certain pairs of classification rules. Our new strategy defines the classification part of an action rule in a unique way. Also, action rules are constructed from single classification rules. We show how to compute their confidence and support. Action rules are used to reclassify objects. In this paper, we propose a method for measuring the level of reclassification freedom for objects in a decision system.

1 Introduction

There are two aspects of interestingness of rules that have been studied in data mining literature, objective and subjective measures [5], [1], [10], [11]. Objective measures are data-driven and domain-independent. Generally, they evaluate the rules based on their quality and similarity between them. Subjective measures, including unexpectedness, novelty and actionability, are user-driven and domain-dependent. A rule is actionable if user can do an action to his/her advantage based on this rule [5]. A new class of rules (called action rules) constructed from certain pairs of association rules, proposed in [8], refers to actionability in a more precise way. A formal definition of an action rule was independently proposed in [2]. Also, interventions introduced in [3] are conceptually very similar to action rules. These rules have been investigated further in [12], [13], [14], [9].

To give an example justifying the need of action rules, let us assume that a number of customers have closed their accounts at one of the banks. We construct, possibly the simplest, description of that group of people and next search for a new description, similar to the one we have, with a goal to identify a new group of customers from which no-one left that bank. If these descriptions have a form of rules, then they can be seen as actionable rules. Now, by comparing these

two descriptions, we may find the cause why these accounts have been closed and formulate an action which if undertaken by the bank, may prevent other customers from closing their accounts. Such actions are stimulated by action rules and they are seen as precise hints for actionability of rules. For example, an action rule may define a group of customers who are seriously thinking to close their bank accounts but if they are invited by the bank for a glass of wine and also get certain offers, most probably they will stay.

The current definition of action rules requires constructing them from certain pairs of classification rules. This assumption makes the process of action rules discovery not only unnecessarily expensive but also gives too much freedom in constructing their classification parts. This paper presents a new strategy for action rules construction and shows how they relate to action rules introduced in [9], [13].

2 Action rules discovery

An information system is used to store information about a group of related objects. Its definition, given here, is due to Pawlak [6]. By an information system we mean a pair $S = (U, A)$, where:

- U is a nonempty, finite set of objects (object identifiers),
- A is a nonempty, finite set of attributes (functions) i.e. $a : U \rightarrow V_a$ for $a \in A$, where V_a is called the domain of a .

We often write (a, v) instead of v , assuming that $v \in V_a$. Information systems can be seen as decision tables. In any decision table together with the set of attributes a partition of that set into conditions and decisions is given. Additionally, we assume that the set of conditions is partitioned into stable and flexible conditions [8]. Attribute $a \in A$ is called stable for the set U , if its values assigned to objects from U can not change in time. Otherwise, it is called flexible. "Date of Birth" is an example of a stable attribute. "Interest rate" on any customer account is an example of a flexible attribute. For simplicity reason, we will consider decision tables with only one decision. We adopt the following definition of a decision table:

By a decision table we mean an information system $S = (U, A_1 \cup A_2 \cup \{d\})$, where $d \notin A_1 \cup A_2$ is a distinguished attribute called decision. The elements of A_1 are called stable conditions, whereas the elements of $A_2 \cup \{d\}$ are called flexible conditions. Our goal is to suggest changes in values of attributes in A_1 for some objects from U so the values of the attribute d for these objects may change as well. A formal expression describing such a property is called an action rule [9], [13].

In order to construct action rules, first we need to extract classification rules describing relationships between attributes from A_1 and the attribute d . By $Dom(r)$ we mean all attributes listed in the *IF* part of a classification rule r .

For example, if $r = [(a_1, 3) \wedge (a_2, 4) \rightarrow (d, 3)]$ is a rule, then $Dom(r) = \{a_1, a_2\}$. By $d(r)$ we denote the decision value of rule r . In our example $d(r) = 3$. If r_1, r_2 are rules and $B \subseteq A_1 \cup A_2$ is a set of attributes, then $r_1/B = r_2/B$ means that the conditional parts of rules r_1, r_2 restricted to attributes B are the same. For example if $r_1 = [(a_1, 3) \rightarrow (d, 3)]$, then $r_1/a_1 = r/a_1$. Assume also that $(a, v \rightarrow w)$ denotes the fact that the value of attribute a needs to be changed from v to w for some objects in U . Similarly, the term $(a, v \rightarrow w)(x)$ means that $a(x) = v$ needs to be changed to $a(x) = w$. Saying another words, the property (a, v) of an object x has to be changed to property (a, w) . Assume now that rules r_1, r_2 are extracted from S and $r_1/A_1 = r_2/A_1$, $d(r_1) = k_1$, $d(r_2) = k_2$ and $k_1 < k_2$ (k_1 is ranked lower than k_2). Also, assume that (b_1, b_2, \dots, b_p) is a list of all attributes in $Dom(r_1) \cap Dom(r_2) \cap A_2$ on which r_1, r_2 differ and $r_1(b_1) = v_1$, $r_1(b_2) = v_2, \dots$, $r_1(b_p) = v_p$, $r_2(b_1) = w_1$, $r_2(b_2) = w_2, \dots$, $r_2(b_p) = w_p$. By (r_1, r_2) -action rule on $x \in U$ we mean an expression: $r = [(b_1, v_1 \rightarrow w_1) \wedge (b_2, v_2 \rightarrow w_2) \wedge \dots \wedge (b_p, v_p \rightarrow w_p)](x) \rightarrow (d, k_1 \rightarrow k_2)(x)$. By the support of action rule r we mean the set of all objects in S satisfying the description $[(b_1, v_1) \wedge (b_2, v_2) \wedge \dots \wedge (b_p, v_p) \wedge (d, k_1)]$. Assume now that by $r(x)$ we mean a new object obtained from x by changing its property (b_i, v_i) to (b_i, w_i) for each $1 \leq i \leq p$. If $d(r(x)) = k_2$, then x supports r .

<i>Stable</i>	<i>Flexible</i>	<i>Stable</i>	<i>Flexible</i>	<i>Stable</i>	<i>Flexible</i>	<i>Decision</i>
<i>A</i>	<i>B</i>	<i>C</i>	<i>E</i>	<i>G</i>	<i>H</i>	<i>D</i>
a_1	b_1	c_1	e_1			d_1
a_1	b_2			g_2	h_2	d_2

Table 1. Two classification rules extracted from S

To construct an extended action rule [13], let us assume that two classification rules, each one referring to a different decision class, are considered. We present them in Table 1 to better clarify the process. Here, "Stable" means stable classification attribute and "Flexible" means flexible one. In a standard representation, these two classification rules have a form:

$$r_1 = [(a_1 \wedge b_1 \wedge c_1 \wedge e_1) \rightarrow d_1], r_2 = [(a_1 \wedge b_2 \wedge g_2 \wedge h_2) \rightarrow d_2].$$

Assume now that object x supports rule r_1 which means that it is classified as d_1 . In order to reclassify x to a higher level class d_2 , we need to change not only the value of B from b_1 to b_2 but also to assume that $G(x) = g_2$ and that the value H for object x has to be changed to h_2 . This is the meaning of the extended (r_1, r_2) -action rule defined by the expression below:

$$r = [(a_1 \wedge (B, b_1 \rightarrow b_2) \wedge g_2 \wedge (H, \rightarrow h_2)](x) \rightarrow (D, d_1 \rightarrow d_2)(x).$$

Assume now that by $Sup(t)$ we mean the number of tuples having property t . By the support of the extended (r_1, r_2) -action rule (given above) we mean:

$Sup[a_1 \wedge b_1 \wedge g_2 \wedge d_1]$. By the confidence of the extended (r_1, r_2) -action rule r (given above) we mean (see [12], [13]):

$$[Sup[a_1 \wedge b_1 \wedge g_2 \wedge d_1]/Sup[a_1 \wedge b_1 \wedge g_2]] \cdot [Sup[a_1 \wedge b_2 \wedge c_1 \wedge d_2]/Sup[a_1 \wedge b_2 \wedge c_1]].$$

Now, we explain the steps needed to compute the confidence of an extended (r_1, r_2) -action rule r . In the first conjunct of the definition of an action rule we are looking for objects described by values of stable attributes only taken from the rule r_2 . Its second conjunct refers to the percentage of objects $r(x)$ which also have a property d_2 , where x supports the first conjunct. Values of stable attributes listed in r_1 do not have to be considered at all. To give another example of an action rule and an extended action rule, assume that $S = (U, A_1 \cup A_2 \cup \{d\})$ is a decision system represented by Table 2. Assume that $A_1 = \{c, b\}$, $A_2 = \{a\}$.

X	c	a	b	d
x_1	2	1	1	L
x_2	1	2	2	L
x_3	2	2	1	H
x_4	1	1	1	L

Table 2. Decision System S

For instance, rules $r_1 = [(a, 1) \wedge (b, 1) \rightarrow (d, L)]$, $r_2 = [(c, 2) \wedge (a, 2) \rightarrow (d, H)]$ can be extracted from S , where $Dom(r_1) = \{x_1, x_4\}$. Extended (r_1, r_2) -action rule $[(a, 1 \rightarrow 2) \wedge (c, 2)](x) \rightarrow [(d, L \rightarrow H)](x)$ is only supported by object x_1 . The corresponding (r_1, r_2) -action rule $[(a, 1 \rightarrow 2)](x) \rightarrow [(d, L \rightarrow H)](x)$ is supported by x_1 and x_4 . The confidence of an extended action rule is higher than the confidence of the corresponding action rule because the object making the confidence of that action rule lower has been removed from its set of support.

3 Action rules discovery, the new approach

Let us assume again that $S = (U, A_1 \cup A_2 \cup \{d\})$ is a decision system, where $d \notin A_1 \cup A_2$ is a distinguished attribute called the decision. The elements of A_1 are called stable conditions, whereas the elements of $A_2 \cup \{d\}$ are called flexible. Assume that $d_1 \in V_d$ and $x \in U$. We say that x is a d_1 -object if $d(x) = d_1$. We also assume that $\{a_1, a_2, \dots, a_p\} \subseteq A_1$, $\{b_1, b_2, \dots, b_q\} \subseteq A_2$, $a_{i,j}$ denotes a value of attribute a_i , $b_{i,j}$ denotes a value of attribute b_i , for any i, j and that

$$r = [[a_{1,1} \wedge a_{2,1} \wedge \dots \wedge a_{p,1} \wedge b_{1,1} \wedge b_{2,1} \wedge \dots \wedge b_{q,1}] \longrightarrow d_1]$$

is a classification rule extracted from S supporting some d_1 -objects in S . By $sup(r)$ and $conf(r)$, we mean the support and the confidence of r , respectively.

Class d_1 is a preferable class and our goal is to reclassify d_2 -objects into d_1 class, where $d_2 \in V_d$.

By an action rule $r_{[d_2 \rightarrow d_1]}$ associated with r and the above reclassification task $(d, d_2 \rightarrow d_1)$ we mean the following expression:

$$r_{[d_2 \rightarrow d_1]} = [[a_{1,1} \wedge a_{2,1} \wedge \dots \wedge a_{p,1} \wedge (b_1, \rightarrow b_{1,1}) \wedge (b_2, \rightarrow b_{2,1}) \wedge \dots \wedge (b_q, \rightarrow b_{q,1})] \rightarrow (d, d_2 \rightarrow d_1)]$$

In a similar way, by an action rule $r_{[\rightarrow d_1]}$ associated with r and the reclassification task $(d, \rightarrow d_1)$ we mean the following expression:

$$r_{[\rightarrow d_1]} = [[a_{1,1} \wedge a_{2,1} \wedge \dots \wedge a_{p,1} \wedge (b_1, \rightarrow b_{1,1}) \wedge (b_2, \rightarrow b_{2,1}) \wedge \dots \wedge (b_q, \rightarrow b_{q,1})] \rightarrow (d, \rightarrow d_1)]$$

The term $[a_{1,1} \wedge a_{2,1} \wedge \dots \wedge a_{p,1}]$, built from values of stable attributes, is called the header of $r_{[d_2 \rightarrow d_1]}$ and its values can not be changed.

The support set of the action rule $r_{[d_2 \rightarrow d_1]}$ is defined as $Sup(r_{[d_2 \rightarrow d_1]}) = \{x \in U : (a_1(x) = a_{1,1}) \wedge (a_2(x) = a_{2,1}) \wedge \dots \wedge (a_p(x) = a_{p,1}) \wedge (d(x) = d_2)\}$.

Clearly, if $conf(r) \neq 1$, then some objects in S satisfying the description $[a_{1,1} \wedge a_{2,1} \wedge \dots \wedge a_{p,1} \wedge b_{1,1} \wedge b_{2,1} \wedge \dots \wedge b_{q,1}]$ are classified as d_2 . According to the rule $r_{[d_2 \rightarrow d_1]}$ they should be classified as d_1 which means that the confidence of $r_{[d_2 \rightarrow d_1]}$ will get decreased.

If $Sup(r_{[d_2 \rightarrow d_1]}) = \emptyset$, then $r_{[d_2 \rightarrow d_1]}$ can not be used for reclassification of objects. Similarly, $r_{[\rightarrow d_1]}$ can not be used for reclassification, if $Sup(r_{[\rightarrow d_1]}) = \emptyset$, for each d_2 where $d_2 \neq d_1$. From the point of view of actionability, such rules are not interesting.

Let $Sup(r_{[\rightarrow d_1]}) = \bigcup \{Sup(r_{[d_2 \rightarrow d_1]}) : (d_2 \in V_d) \wedge (d_2 \neq d_1)\}$ and $Sup(R_{[\rightarrow d_1]}) = \bigcup \{Sup(r_{[\rightarrow d_1]}) : r \in R(d_1)\}$, where $R(d_1)$ is the set of all classification rules extracted from S which are defining d_1 . So, $Sup(R_S) = \bigcup \{Sup(R_{[\rightarrow d_1]}) : d_1 \in V_d\}$ contains all objects in S which potentially can be reclassified.

Assume now that $U(d_1) = \{x \in U : d(x) \neq d_1\}$. Objects in the set $B(d_1) = [U(d_1) - Sup(R_{[\rightarrow d_1]})]$ can not be reclassified to the class d_1 and they are called d_1 -resistant.

Let $B(-d_1) = \bigcap \{B(d_i) : (d_i \in V_d) \wedge (d_i \neq d_1)\}$. Clearly $B(-d_1)$ represents the set of d_1 -objects which can not be reclassified. They are called d_1 -stable. Similarly, the set $B_d = \bigcup \{B(-d_i) : d_i \in V_d\}$ represents objects in U which can not be reclassified to any decision class. All these objects are called d -stable. In order to show how to find them, the notion of a confidence of an action rule is needed.

Let $r_{[d_2 \rightarrow d_1]}$, $r'_{[d_2 \rightarrow d_3]}$ are two action rules extracted from S . We say that these rules are p-equivalent (\simeq), if the condition given below holds for every $b_i \in A_1 \cup A_2$:

$$\text{if } r/b_i, r'/b_i \text{ are both defined, then } r/b_i = r'/b_i.$$

Now, we explain how to calculate the confidence of $r_{[d_2 \rightarrow d_1]}$. Let us take d_2 -object $x \in Sup(r_{[d_2 \rightarrow d_1]})$. We say that x positively supports $r_{[d_2 \rightarrow d_1]}$ if there is no classification rule r' extracted from S and describing $d_3 \in V_d$, $d_3 \neq d_1$, which is p-equivalent to r , such that $x \in Sup(r'_{[d_2 \rightarrow d_3]})$. The corresponding subset of $Sup(r_{[d_2 \rightarrow d_1]})$ is denoted by $Sup^+(r_{[d_2 \rightarrow d_1]})$. Otherwise, we say that x negatively supports $r_{[d_2 \rightarrow d_1]}$. The corresponding subset of $Sup(r_{[d_2 \rightarrow d_1]})$ is denoted by $Sup^-(r_{[d_2 \rightarrow d_1]})$.

By the confidence of $r_{[d_2 \rightarrow d_1]}$ in S we mean:

$$Conf(r_{[d_2 \rightarrow d_1]}) = [card[Sup^+(r_{[d_2 \rightarrow d_1]})]/card[Sup(r_{[d_2 \rightarrow d_1]})]] \cdot conf(r).$$

Now, if we assume that $Sup^+(r_{[\rightarrow d_1]}) = \bigcup\{Sup^+(r_{[d_2 \rightarrow d_1]}) : (d_2 \in V_d) \wedge (d_2 \neq d_1)\}$, then by the confidence of $r_{[\rightarrow d_1]}$ in S we mean:

$$Conf(r_{[\rightarrow d_1]}) = [card[Sup^+(r_{[\rightarrow d_1]})]/card[Sup(r_{[\rightarrow d_1]})]] \cdot conf(r).$$

Now, we go back to the problem of finding all d -stable objects in S . First of all, we observe that $B^+(d_1) = [U(d_1) - Sup^+(R_{[\rightarrow d_1]})]$ contains all d_1 -resistant objects. Let $B^+(-d_1) = \bigcap\{B^+(d_i) : (d_i \in V_d) \wedge (d_i \neq d_1)\}$. Clearly $B^+(-d_1)$ is the set of all d_1 -stable objects in S . The same $B_d^+ = \bigcup\{B^+(-d_i) : d_i \in V_d\}$ contains all d -stable objects in S .

It can be easily proved that the definition of support and confidence of action rules given in Section 3 is equivalent to the definition of support and confidence given in Section 2.

4 Class-movability of objects

In this section we introduce the notion of a class-movability index (m-index) assigned to an object and next we partition objects in U with respect to that index. The m-index associated with an object, shows a rank of the object. Objects of higher rank are seen as objects more preferably movable between decision classes than objects of lower rank. Let (V_d, \preceq) is an ordered set of values of the decision attribute d , where $V_d = \{d_1, d_2, \dots, d_k\}$. Before defining \preceq , we introduce function F_S , called *decision attribute ranking* associated with S . It assigns an integer number to each d_i , $1 \leq i \leq k$. If $F_S(d_i) \leq F_S(d_j)$, then $d_i \preceq d_j$.

Now, for each $j \in \{1, 2, \dots, k\}$, we start with a collection of sets $P_j(i) = Sup^+(r_{[d_j \rightarrow d_i]})$, each containing all positively supported d_j -objects in S . It means $P_j(i)$ contains all objects in U which can be reclassified (moved) from the decision class d_j to the class d_i . Let $P_j(N) = \bigcap\{P_j(i) : i \in N\}$, for any $N \subseteq \{1, 2, \dots, k\}$. Clearly, $P_j(N)$ contains all objects in U which can be moved from the decision class d_j to any of the classes d_i , where $i \in N$.

Before we give the definition of m-index assigned to an object, we introduce the notion of m-index assigned to N_j , where $N_j \subseteq \{1, 2, \dots, k\}$. Let $N_j^+ = \{i \in N : F_S(d_i) - F_S(d_j) > 0\}$, for any $N \subseteq \{1, 2, \dots, k\}$. The set N_j^+ represents all

attribute values in $\{d_i : i \in N\}$ which have higher decision attribute ranking than d_j .

By class-movability index (m-index) assigned to N_j , we mean: $ind(N_j) = \sum\{F_S(d_i) - F_S(d_j) : i \in N_j^+\}$.

By class-movability index (m-index) assigned to d_j -object x , we mean: $ind_S(x) = \max\{ind(N_j) : N_j \subseteq \{1, 2, \dots, k\} \wedge x \in P_j(N)\}$.

In general, $ind_S(x)$ shows the overall amount of improvement open to x in terms of a number of available re-classifications and the improvements linked with every re-classification.

5 An example

Let us assume that the decision system $S = (U, \{A_1 \cup A_2 \cup \{d\}\})$, where $U = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\}$, is represented by Table 3. A number of different methods can be used to extract rules in which the THEN part consists of the decision attribute d and the IF part consists of attributes belonging to $A_1 \cup A_2$. In our example, the set $A_1 = \{a, b, c\}$ contains stable attributes and $A_2 = \{e, f, g\}$ contains flexible attributes. System *LEERS* [4] is used in this paper for rules extraction.

Table 3. Decision System

U	a	b	c	e	f	g	g
x_1	a_1	b_1	c_1	e_1	f_2	g_1	d_1
x_2	a_2	b_1	c_2	e_2	f_2	g_2	d_3
x_3	a_3	b_1	c_1	e_2	f_2	g_3	d_2
x_4	a_1	b_1	c_2	e_2	f_2	g_1	d_2
x_5	a_1	b_2	c_1	e_3	f_2	g_1	d_2
x_6	a_2	b_1	c_1	e_2	f_3	g_1	d_2
x_7	a_2	b_3	c_2	e_2	f_2	g_2	d_2
x_8	a_2	b_1	c_1	e_3	f_2	g_3	d_2

We are interested in reclassifying d_2 -objects either to class d_1 or d_3 . Four certain classification rules describing d_1, d_3 can be extracted by LERS from the decision system S . They are given below:

$$r1 = [b_1 \wedge c_1 \wedge f_2 \wedge g_1] \rightarrow d_1, r2 = [a_2 \wedge b_1 \wedge e_2 \wedge f_2] \rightarrow d_3, r3 = e_1 \rightarrow d_1, r4 = [b_1 \wedge g_2] \rightarrow d_3.$$

It can be checked that $R_{[d, \rightarrow d_1]} = \{r1, r3\}$ and $R_{[d, \rightarrow d_3]} = \{r2, r4\}$. Action rules associated with $r1, r2, r3, r4$ and the reclassification task either $(d, d_2 \rightarrow d_1)$ or $(d, d_2 \rightarrow d_3)$ are:

$$r1_{[d_2 \rightarrow d_1]} = [b_1 \wedge c_1 \wedge (f, \rightarrow f_2) \wedge (g, \rightarrow g_1)] \rightarrow (d, d_2 \rightarrow d_1),$$

$$r2_{[d_2 \rightarrow d_3]} = [a_2 \wedge b_1 \wedge (e, \rightarrow e_2) \wedge (f, \rightarrow f_2)] \rightarrow (d, d_2 \rightarrow d_3),$$

$$\begin{aligned} r3_{[d_2 \rightarrow d_1]} &= [(e, \rightarrow e_1)] \rightarrow (d, d_2 \rightarrow d_1), \\ r4_{[d_2 \rightarrow d_3]} &= [b_1 \wedge (g, \rightarrow g_2)] \rightarrow (d, d_2 \rightarrow d_3). \end{aligned}$$

It can be easily shown that $Sup(r1_{[d_2 \rightarrow d_1]}) = \{x_3, x_6, x_8\}$, $Sup(r2_{[d_2 \rightarrow d_3]}) = \{x_6, x_8\}$, $Sup(r3_{[d_2 \rightarrow d_1]}) = \{x_3, x_4, x_5, x_6, x_7, x_8\}$, $Sup(r4_{[d_2 \rightarrow d_3]}) = \{x_3, x_4, x_6, x_8\}$. The same, $Sup(r1_{[\rightarrow d_1]}) = \{x_3, x_4, x_5, x_6, x_7, x_8\}$, $Sup(r2_{[\rightarrow d_3]}) = \{x_3, x_4, x_6, x_8\}$. So, all d_2 -objects can be potentially reclassified to the class d_1 . On the other hand, d_2 -objects only from the set $\{x_3, x_4, x_6, x_8\}$ can be potentially reclassified to the same class d_1 . It means that d_2 -objects $\{x_5, x_7\}$ are d_1 -resistant.

Now, let us notice that action rules $r1_{[d_2 \rightarrow d_1]}$, $r2_{[d_2 \rightarrow d_3]}$ are p-equivalent. Since $Sup(r1_{[d_2 \rightarrow d_1]}) = \{x_3, x_6, x_8\}$ and $Sup(r2_{[d_2 \rightarrow d_3]}) = \{x_6, x_8\}$, then only x_3 positively supports $r1_{[d_2 \rightarrow d_1]}$ and objects in $\{x_6, x_8\}$ negatively support both rules $r1_{[d_2 \rightarrow d_1]}$, $r2_{[d_2 \rightarrow d_3]}$. So, the confidence:

$$Conf(r1_{[d_2 \rightarrow d_1]}) = [card[Sup^+(r1_{[d_2 \rightarrow d_1]})]/card[Sup(r1_{[d_2 \rightarrow d_1]})]] \cdot conf(r1) = [1/3] \cdot 1 = 1/3.$$

$$Conf(r2_{[d_2 \rightarrow d_3]}) = [card[Sup^+(r2_{[d_2 \rightarrow d_3]})]/card[Sup(r2_{[d_2 \rightarrow d_3]})]] \cdot conf(r2) = [0/2] \cdot 1 = 0.$$

6 Conclusion

The action rules discovery process, presented in this paper, differs significantly from previous strategies because we use only single classification rules, instead of their pairs, to construct action rules. This difference is quite essential because the number of classification rules discovered from a database is usually large. The previous strategies (see [9], [12], [13], [14]) require to consider all pairs of classification rules, defining different decision values, which makes their time complexity quite high. Also, the proposed new approach to action rules discovery allows to define their confidence in a more natural way than in previous papers (see [12]). The results presented in [3], [8], [12], [13], [14] need to be generalized to match this new definition of an action rule. But, this generalization process should be rather straightforward.

Our initial implementation of the proposed new method for action rules discovery outperforms the previous strategy [12], [13], [14] in terms of time complexity. Clearly, the confidence and support of action rules is strictly dependent on the support and confidence of classification rules used to construct them.

7 Acknowledgements

This research was partially supported by the National Science Foundation under grant IIS-0414815.

References

1. Adomavicius, G., Tuzhilin, A. (1997), *Discovery of actionable patterns in databases: the action hierarchy approach*, in Proceedings of KDD97 Conference, Newport Beach, CA, AAAI Press
2. Geffner, H., Wainer, J. (1998), *Modeling action, knowledge and control*, ECAI 98, 13th European Conference on AI, (Ed. H. Prade), John Wiley & Sons, 532-536
3. Greco, S., Matarazzo, B., Pappalardo, N., Slowiński, R. (2005), *Measuring expected effects of interventions based on decision rules*, Journal of Experimental and Theoretical Artificial Intelligence, Taylor Francis, Vol. 17, No. 1-2
4. Chmielewski, M.R., Grzymala-Busse, J.W., Peterson, N.W., Than, S. (1993), *The rule induction system LERS - a version for personal computers*, Foundations of Computing and Decision Sciences, Vol. 18, No. 3-4, Institute of Computing Science, Technical University of Poznan, Poland, 181-212
5. Liu, B., Hsu, W., Chen, S. (1997), *Using general impressions to analyze discovered classification rules*, Proceedings of KDD97 Conference, Newport Beach, CA, AAAI Press
6. Pawlak, Z.(1991), *Information systems - theoretical foundations*, Information Systems Journal, Vol. 6, 205-218
7. Raś, Z.W., Gupta, S. (2002), *Global action rules in distributed knowledge systems*, Fundamenta Informaticae Journal, IOS Press, Vol. 51, No. 1-2, 175-184
8. Raś, Z., Wieczorkowska, A. (2000), *Action Rules: how to increase profit of a company*, in Principles of Data Mining and Knowledge Discovery, (Eds. D.A. Zighed, J. Komorowski, J. Zytkow), Proceedings of PKDD'00, Lyon, France, LNAI, No. 1910, Springer-Verlag, 587-592
9. Raś, Z.W., Tzacheva, A., Tsay, L.-S. (2005), *Action rules*, Encyclopedia of Data Warehousing and Mining, (Ed. J. Wang), Idea Group Inc., 1-5
10. Silberschatz, A., Tuzhilin, A., (1995), *On subjective measures of interestingness in knowledge discovery*, Proceedings of KDD'95 Conference, AAAI Press
11. Silberschatz, A., Tuzhilin, A., (1996), *What makes patterns interesting in knowledge discovery systems*, IEEE Transactions on Knowledge and Data Engineering Vol. 5, No. 6
12. Tsay, L.-S., Raś, Z.W. (2005), *Action Rules Discovery System DEAR, Method and Experiments*, Journal of Experimental and Theoretical Artificial Intelligence, Taylor & Francis, Vol. 17, No. 1-2, 119-128
13. Tsay, L.-S., Raś, Z.W., Dardzinska, A., *Mining E-Action Rules*, in *Mining Complex Data*, Proceedings of 2005 IEEE ICDM Workshop in Houston, Texas, Published by Math. Dept., Saint Mary's Univ., Nova Scotia, Canada, 2005, 85-90
14. Tzacheva, A., Raś, Z.W. (2005), *Action rules mining*, International Journal of Intelligent Systems, Wiley, Vol. 20, No. 7, 719-736