Totally Optimal Decision Rules

Talha Amin, Mikhail Moshkov

Computer, Electrical and Mathematical Sciences & Engineering Division, King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia

Abstract

Optimality of decision rules (patterns) can be measured in many ways. One of these is referred to as length. Length signifies the number of terms in a decision rule and is optimally minimized. Another, coverage, represents the width of a rule's applicability and generality. As such, it is desirable to maximize coverage. A totally optimal decision rule is a decision rule that has the minimum possible length and the maximum possible coverage. This paper presents a method for determining the presence of totally optimal decision rules for "complete" decision tables (representations of total functions in which different variables can have domains of differing values). Depending on the cardinalities of the domains, we can either guarantee for each tuple of values of the function that totally optimal rules exist for each row of the table (as in the case of total Boolean functions where the cardinalities are equal to 2) or, for each row, we can find a tuple of values of the function for which totally optimal rules do not exist for this row.

Keywords:
decision rules, patterns, length, coverage

1. Introduction

A decision rule (pattern) is a mapping from a set of conditions to an outcome. These patterns are widely used in applications concerning data mining [12], knowledge representation and discovery, and machine learning [8]. Many problems in these areas can be solved with multiple approaches including systems of decision rules [9], decision trees [22], logical representations [19], k-nearest neighbor [11], neural networks, and support vector machines [10].

Although there are many comparable methods, decision rules tend to be the most meaningful and easily understood by humans [13]. This allows them to excel where knowledge representation is concerned. Even among decision rules some are better than others. For example, short decision rules are easy to

Email addresses: talha.amin@kaust.edu.sa (Talha Amin), mikhail.moshkov@kaust.edu.sa (Mikhail Moshkov)

remember and simple to understand while rules that cover many cases are more general and applicable. These and other such criteria mean that it is important to consider which rules to include in a system.

There are many different ways to design and analyze decision rules, such as brute force, genetic algorithms [25], boolean reasoning [21], derivation from decision trees [23, 18], sequential covering procedures [9, 13], greedy algorithms [17], and dynamic programming [2, 26]. Implementations of many of these methods can be found in programs such as LERS [14], RSES [5], Rosetta [20], Weka [15], TRS Library [24], and DAGGER [1].

According to Hammer, Kogan, Simeone, and Szedmak [16], "it has been observed in empirical studies and practical applications that some patterns are more suitable than others for use in data analysis". The decision rules that we consider are very similar to patterns from logical analysis of data [6, 7, 16], and the same consideration can be applied to them. Hammer et al [16] go on to consider the Pareto optimality of patterns with regards to various criteria, developing algorithms to optimize a given pattern with respect to any one criteria.

In this paper, we consider instead the existence of decision rules that are totally optimal with respect to both of our chosen criteria, thus removing the need to consider Pareto optimality. The criteria we consider are length and coverage. Length is an important criteria because short rules are easier to understand. Coverage is important because the more coverage a rule has, the more general it is and the wider its application. Intuitively, it might seem that considering length and coverage will also always result in totally optimal rules. After all, shorter rules have less restrictions, leading to greater coverage. However, based on results seen in [2], we know that this is often not the case.

The main result of our work defines a system of inequalities for "complete" decision tables (representations of total functions in which different variables can have domains of differing values). When all inequalities in this system are true then, for each total function, we can guarantee the existence of totally optimal decision rules for each row. Otherwise, for each row, we can guarantee the existence of at least one total function such that totally optimal decision rules do not exist for the considered row. In particular, we utilize this system of inequalities to prove that for any Boolean function that is defined on all values of input (total Boolean functions) there exist totally optimal decision rules. The same situation applies to total functions of k-valued logic ($k > 2$) Furthermore, we study datasets from the UCI Machine Learning Repository [4] and show that a considerable amount of datasets are complete tables that can be proven to have totally optimal decision rules.

This paper consists of three sections. Section 2 provides basic definitions and introduces concepts necessary for understanding the rest of the work. Section 3 then uses these concepts to derive two theorems regarding totally optimal decision rules. One of these theorems is then used to prove the existence of totally optimal decision rules for all total Boolean functions. Section 4 presents the findings when working with the UCI Machine Learning Repository. Finally, Section 5 contains conclusions.

2. Decision Tables and Rules

In this section, we consider the main notions and notation related to decision tables and decision rules.

2.1. Decision Tables

A *decision table* is a rectangular table T with $n \geq 1$ columns filled with numbers from the set of nonnegative integers. Columns of the table are labeled with pairwise different *conditional* attributes $f_1, \ldots, f_n$, respectively. Rows of the table are pairwise different, and each row is labeled with a nonnegative integer which is interpreted as a decision (a value of the *decision* attribute d). Rows of the table are interpreted as tuples of values of conditional attributes. Note that in the definition of decision table, instead of natural numbers we can use symbols of arbitrary infinite alphabet.

A decision table is called *empty* if it has no rows. The table T is called *degenerate* if it is empty or all rows of T are labeled with the same decision. We denote by $\dim(T)$ the number of columns (conditional attributes) in T and by $N(T)$ the number of rows in the table T.

For any conditional attribute $f_i \in \{f_1, \ldots, f_n\}$, we denote by $E(T, f_i)$ the set of values of the attribute f_i in the table T. We denote by $E(T)$ the set of conditional attributes for which $|E(T, f_i)| \geq 2$.

Let T be a nonempty decision table. A *subtable* of T is a table obtained from T by the removal of some rows. Let $f_{i_1}, \ldots, f_{i_m} \in \{f_1, \ldots, f_n\}$ and $a_1, \ldots, a_m \in \omega$ where ω is the set of nonnegative integers. We denote by $T(f_{i_1}, a_1) \ldots (f_{i_m}, a_m)$ the subtable of the table T containing the rows from T which at the intersection with the columns $f_{i_1}, \ldots, f_{i_m}$ have numbers $a_1, \ldots, a_m$, respectively.

A *complete* decision table is a representation of a total function in which different variables can have domains of differing values. Given that $B_1, \ldots, B_n$ are nonempty finite subsets of the set of nonnegative integers, $I = B_1 \times \ldots \times B_n$, and $\nu : I \to \omega$, the pair $T = (I, \nu)$ can be used to describe the complete decision table T with conditional attributes $f_1, \ldots, f_n$ that have values from the sets $B_1, \ldots, B_n$, respectively. n-Tuples from I are rows of T, and the value $\nu(b_1, \ldots, b_n)$ is the decision attached to a row $(b_1, \ldots, b_n) \in I$. The table T is nondegenerate if and only if ν is a non-constant function (which has at least two different values).

2.2. Decision Rules

Let T be a decision table with n conditional attributes $f_1, \ldots, f_n$ and $r = (b_1, \ldots, b_n)$ be a row of T. A *decision rule over* T is an expression of the kind

$$f_{i_1} = a_1 \wedge \ldots \wedge f_{i_m} = a_m \to t \tag{1}$$

where $f_{i_1}, \ldots, f_{i_m} \in \{f_1, \ldots, f_n\}$, and $a_1, \ldots, a_m, t$ are nonnegative integers. It is possible that $m = 0$. For the considered rule, we denote $T^0 = T$, and if $m > 0$

we denote $T^j = T(f_{i_1}, a_1) \dots (f_{i_j}, a_j)$ for $j = 1, \dots, m$. We will say that the decision rule (1) *covers* the row $r = (b_1, ..., b_n)$ if $b_{i_1} = a_1, \dots, b_{i_m} = a_m$.

A decision rule (1) over T is called a *decision rule for* T if all the rows of T that are covered by the rule are labeled with the decision t, and either $m = 0$, or $m > 0$ and, for $j = 1, \dots, m$, T^{j-1} is not degenerate, and $f_{i_j} \in E(T^{j-1})$. A decision rule (1) for T is called a *decision rule for* T *and* r if it covers r.

We denote by $DR(T)$ the set of decision rules for T. By $DR(T, r)$ we denote the set of decision rules for T and r.

We now define the notion of *length* for decision rules. The length of a decision rule is equal to the number of attribute-value pairs on the left hand side. For a decision rule ρ of the form (1), the length $L(T, \rho) = m$.

We will also define the notion of *coverage* for decision rules. The coverage of a decision rule is equal to the number of rows that the decision rule covers. The coverage $C(T, \rho)$ of the decision rule ρ (1) is equal to the number of rows in $T(f_{i_1}, a_1)...(f_{i_m}, a_m)$.

3. Totally Optimal Decision Rules

In this section, we will study the existence of decision rules that are totally optimal (simultaneously optimal with respect to length and coverage). This study allows us to partially explain a phenomenon observed experimentally in many cases: the existence of decision rules that have the minimum length and the maximum coverage simultaneously.

We know that the set $DR(T, r)$ is equal to the set of decision rules for a nonempty decision table T and row r from T. We will now define a *totally optimal decision rule* ρ *for* T *and* r as a decision rule ρ for T and r such that $L(T, \rho) = \min\{L(T, \rho') : \rho' \in DR(T, r)\}$ and $C(T, \rho) = \max\{C(T, \rho') : \rho' \in DR(T, r)\}$.

Now we will consider the existence of totally optimal decision rules for complete decision tables.

Let $T = (I, \nu)$ be a complete decision table with conditional attributes $f_1, \dots, f_n$ that have values from the sets $B_1, \dots, B_n$, respectively. Let $c_i = |B_i|$ for $i = 1, \dots, n$, $2 \leq c_1 \leq \dots \leq c_n$, and $f_{j_1} = b_{j_1} \wedge \dots \wedge f_{j_k} = b_{j_k} \to \nu(r)$ be a rule for T and the row $r = (b_1, ..., b_n)$ from T. We denote this rule as ρ. Because T is a complete table, $C(T, \rho) = \prod_{i \in \{1,\dots,n\} \setminus \{j_1,\dots,j_k\}} c_i$. Furthermore, it follows that $C(T, \rho) \leq c_{k+1} \times \dots \times c_n$, and $C(T, \rho) \geq c_1 \times \dots \times c_{n-k}$.

Let us consider the system of inequalities

$$\{c_{m+2} \times \dots \times c_n \leq c_1 \times \dots \times c_{n-m} : m = 1, \dots, n-1\}. \quad (2)$$

If $m = n - 2$ then we have the inequality $c_n \leq c_1 \times c_2$. If $m = n - 1$ then we have the inequality $1 \leq c_1$ which holds since $c_1 \geq 2$.

Theorem 1. *Let $B_1, \dots, B_n$ be nonempty finite subsets of the set ω, $c_i = |B_i|$ for $i = 1, \dots, n$, $2 \leq c_1 \leq \dots \leq c_n$, and $I = B_1 \times \dots \times B_n$.*

If each inequality from the system (2) holds then, for each non-constant function $\nu : I \to \omega$ and for each row r of the complete decision table $T = (I, \nu)$, there exists a totally optimal decision rule for T and r.

Otherwise, for each row r from I there exists a non-constant function $\nu_r : I \to \omega$ such that there is no a totally optimal decision rule for $T = (I, \nu_r)$ and r.

Proof. Let each equality from the system (2) hold, $\nu : I \to \omega$ be a non-constant function, and $r = (b_1, \ldots, b_n)$ be a row of the decision table $T = (I, \nu)$. We will show that there exists a totally optimal rule for T and r. Let us assume the contrary. In this case there exist two rules ρ_1 and ρ_2 for T and r such that $L(T, \rho_2) > L(T, \rho_1)$ and $C(T, \rho_2) > C(T, \rho_1)$. Let $m = L(T, \rho_1)$. Then $L(T, \rho_2) \geq m + 1$, and $m > 0$ since ν is a non-constant function.

Since $c_1 \leq \ldots \leq c_n$, we have $C(T, \rho_1) \geq c_1 \times \ldots \times c_{n-m}$ and $C(T, \rho_2) \leq c_{m+2} \times \ldots \times c_n$. We know that $C(T, \rho_2) > C(T, \rho_1)$. Therefore $c_{m+2} \times \ldots \times c_n > c_1 \times \ldots \times c_{n-m}$, but this is impossible. Hence a totally optimal decision rule for T and r exists.

Let there exist $m \in \{1, \ldots, n-2\}$ such that $c_{m+2} \times \ldots \times c_n > c_1 \times \ldots \times c_{n-m}$. For an arbitrary $r = (b_1, \ldots, b_n) \in I$, we define a function $\nu_r : I \to \omega$. For $x = (x_1, \ldots, x_n) \in I$, $\nu_r(x) = 1$ if $x_1 = b_1, \ldots, x_{m+1} = b_{m+1}$ or $x_{n-m+1} = b_{n-m+1}, \ldots, x_n = b_n$. Otherwise, $\nu_r(x) = 0$. The function ν_r is non-constant: $\nu_r(r) = 1$ and $\nu_r(x) = 0$ if, in particular, $x_1 \neq b_1$ and $x_n \neq b_n$.

Let $T = (I, \nu_r)$, $p_1 = c_1 \times \ldots \times c_{n-m}$ and $p_2 = c_{m+2} \times \ldots \times c_n$. We now show that $p_1 < p_2$ since it is obvious that $m < m + 1$. We denote by ρ_1 the rule

$$f_{n-m+1} = b_{n-m+1} \wedge \ldots \wedge f_n = b_n \to 1,$$

and by ρ_2 we denote the rule

$$f_1 = b_1 \wedge \ldots \wedge f_{m+1} = b_{m+1} \to 1.$$

It is not difficult to show that ρ_1 and ρ_2 are decision rules for T and r such that $L(T, \rho_1) = m$, $C(T, \rho_1) = p_1$, $L(T, \rho_2) = m + 1$, and $C(T, \rho_2) = p_2$. Therefore $L(T, \rho_1) < L(T, \rho_2)$ and $C(T, \rho_1) < C(T, \rho_2)$ meaning that neither is a totally optimal decision rule.

Let ρ be an exact decision rule for T and r, and ρ be equal to

$$f_{i_1} = b_{i_1} \wedge \ldots \wedge f_{i_k} = b_{i_k} \to 1.$$

Let $A = \{i_1, \ldots, i_k\}$, $B = \{n - m + 1, \ldots, n\}$, and $C = \{1, \ldots, m + 1\}$. We now show that $B \subseteq A$ or $C \subseteq A$. Let us assume the contrary: there is $s \in B$ such that $s \notin A$, and there is $t \in C$ such that $t \notin A$. We denote by x a n-tuple $(x_1, \ldots, x_n)$ from I such that $x_i = b_i$ if $i \in \{1, \ldots, n\} \setminus \{s, t\}$, $x_s \neq b_s$, and $x_t \neq b_t$. It is clear that x is covered by the rule ρ and $\nu_r(x) = 0$, but this is impossible.

If $B \subseteq A$ then $L(T, \rho) \geq L(T, \rho_1)$ and $C(T, \rho) \leq C(T, \rho_1)$. If $C \subseteq A$ then $L(T, \rho) \geq L(T, \rho_2)$ and $C(T, \rho) \leq C(T, \rho_2)$. Therefore either $(C(T, \rho), L(T, \rho)) \in$

$\{(p_1, m), (p_2, m+1)\}$ or the values $(C(T, \rho), L(T, \rho))$ are worse (from the point of view of length and coverage) than either (p_1, m) or $(p_2, m+1)$. Furthermore, $p_1 < p_2$ and $m < m+1$. From here it follows that there is no a totally optimal exact decision rule for T and r. ■

Theorem 1 can be formulated without any reference to a complete decision table since its conditions relate to a non-constant, multivariate function defined on a finite domain. Algorithms for the construction of totally optimal decision rules such as those mentioned in Theorem 1 can be found in [2].

Example 2. *Let us now consider an example:*

Let $c_1 = 3$, $c_2 = 4$, $c_3 = 4$, and $c_4 = 5$. The system of equations from Theorem 1 in this case result in the following:

$$\begin{array}{lll} c_3 \times c_4 \leq c_1 \times c_2 \times c_3 & 4 \times 5 \leq 3 \times 4 \times 4 & : m = 1 \\ c_4 \leq c_1 \times c_2 & 5 \leq 3 \times 4 & : m = 2 \\ 1 \leq c_1 & 1 \leq 3 & : m = 3 \end{array}$$

From here it follows that since all of the equalities hold, we are guaranteed that there is a totally optimal decision rule for each row of this table.

Example 3. *Let us now consider another example:*

Let $c_1 = 2$, $c_2 = 2$, $c_3 = 5$. The system of equations from Theorem 1 in this case result in the following:

$$\begin{array}{lll} c_3 \leq c_1 \times c_2 & 5 \leq 2 \times 2 & : m = 1 \\ 1 \leq c_1 & 1 \leq 2 & : m = 2 \end{array}$$

Here the first equation is clearly false.

We consider now a special case when all $c_1, \ldots, c_n$ are equal. It is easy to prove the following statement.

Proposition 4. *Let $c_1 = \ldots = c_n = c \geq 2$. Then the system of inequalities (2) is equal to the system $\{c^{n-m-1} \leq c^{n-m} : m = 1, \ldots, n-1\}$ in which each inequality holds.*

Let n, k be natural numbers, $k \geq 2$, and $E_k = \{0, \ldots, k-1\}$. A function $f : E_k^n \to E_k$ is called a *function of k-valued logic* (if $k = 2$ then f is a *Boolean* function). We correspond to this function a complete decision table $T_f = (E_k^n, f)$ with conditional attributes $x_1, \ldots, x_n$. From Theorem 1 and Proposition 4 it follows that, for each row r of the decision table T_f, there exists a totally optimal exact decision rule for T_f and r relative to C and L.

We consider now another special case when among $c_1, \ldots, c_n$ there are only two different numbers.

Proposition 5. *Let $c_1 = \ldots = c_{n_a} = c_a$ and $c_{n_a+1} = \ldots = c_{n_a+n_b} = c_b$, where $n = n_a + n_b$, $n_a > 0$, $n_b > 0$, and $2 \leq c_a < c_b$. If $n_a > n_b$ then each inequality from the system of inequalities (2) holds if and only if the inequality $c_b^{n_b} \leq c_a^{n_b+1}$ holds. If $n_a \leq n_b$ then each inequality from the system of inequalities (2) holds if and only if the inequality $c_b^{n_a-1} \leq c_a^{n_a}$ holds.*

Proof. For $m \in \{1, \ldots, n-1\}$, we denote by $ineq(m)$ the inequality $c_{m+2} \times \ldots \times c_n \leq c_1 \times \ldots \times c_{n-m}$ from the system of inequalities (2). We will say that two inequalities are *equivalent* if they either both hold or both do not hold.

Let $n_a > n_b$. We consider three cases. If $1 \leq m \leq n_b$ then $ineq(m)$ is equal to $c_a^{n_a-m-1} \times c_b^{n_b} \leq c_a^{n_a} \times c_b^{n_b-m}$, and this inequality is equivalent to $c_b^m \leq c_a^{m+1}$. If $n_b < m < n_a$ then $ineq(m)$ is equal to $c_a^{n_a-m-1} \times c_b^{n_b} \leq c_a^{n_a-(m-n_b)}$, and this inequality is equivalent to $c_b^{n_b} \leq c_a^{n_b+1}$. If $n_a \leq m \leq n-1$ then $ineq(m)$ is equal to $c_b^{n-m-1} \leq c_a^{n-m}$ (note that $0 \leq n-m-1 \leq n_b-1$).

The inequality $c_b^{n_b} \leq c_a^{n_b+1}$ is equivalent to the inequality $ineq(n_b)$. If each inequality from the system (2) holds then the inequality $c_b^{n_b} \leq c_a^{n_b+1}$ holds. Let now the inequality $c_b^{n_b} \leq c_a^{n_b+1}$ hold. If $n_b < m < n_a$ then the inequality $ineq(m)$ holds (it is equivalent to $c_b^{n_b} \leq c_a^{n_b+1}$). If $1 \leq m \leq n_b$ or $n_a \leq m \leq n-1$ then the inequality $ineq(m)$ is equivalent to the inequality $c_b^k \leq c_a^{k+1}$ for some $k \in \{0, \ldots, n_b\}$. Let $t = n_b - k$. If we divide the left-hand side of the inequality $c_b^{n_b} \leq c_a^{n_b+1}$ by c_b^t and the right-hand side by c_a^t we obtain the inequality $c_b^k \leq c_a^{k+1}$ which holds also. Therefore each inequality from the system (2) holds.

Let $n_a \leq n_b$. We consider three cases. If $1 \leq m < n_a$ then $ineq(m)$ is equal to $c_a^{n_a-m-1} \times c_b^{n_b} \leq c_a^{n_a} \times c_b^{n_b-m}$, and this inequality is equivalent to $c_b^m \leq c_a^{m+1}$. If $n_a \leq m \leq n_b$ then $ineq(m)$ is equal to $c_b^{n_b-(m+1-n_a)} \leq c_a^{n_a} \times c_b^{n_b-m}$, and this inequality is equivalent to $c_b^{n_a-1} \leq c_a^{n_a}$. If $n_b < m \leq n-1$ then $ineq(m)$ is equal to $c_b^{n-m-1} \leq c_a^{n-m}$ (note that $0 \leq n-m-1 \leq n_a-2$).

The inequality $c_b^{n_a-1} \leq c_a^{n_a}$ is equivalent to the inequality $ineq(n_a)$. If each inequality from the system (2) holds then the inequality $c_b^{n_a-1} \leq c_a^{n_a}$ holds. Let now the inequality $c_b^{n_a-1} \leq c_a^{n_a}$ hold. If $n_a \leq m \leq n_b$ then the inequality $ineq(m)$ holds (it is equivalent to $c_b^{n_a-1} \leq c_a^{n_a}$). If $1 \leq m < n_a$ or $n_b < m \leq n-1$ then the inequality $ineq(m)$ is equivalent to the inequality $c_b^k \leq c_a^{k+1}$ for some $k \in \{0, \ldots, n_a-1\}$. Let $t = n_a - k - 1$. If we divide the left-hand side of the inequality $c_b^{n_a-1} \leq c_a^{n_a}$ by c_b^t and the right-hand side by c_a^t we obtain the inequality $c_b^k \leq c_a^{k+1}$ which also holds. Therefore each inequality from the system (2) holds. ■

Corollary 6. *Let $c_1 = c_a$ and $c_2 = \ldots = c_n = c_b$, where $2 \leq c_a < c_b$. Then each inequality from the system of inequalities (2) holds.*

4. Application on UCI Machine Learning Repository

In this section we will present our findings when working with datasets from the UCI Machine Learning Repository [4]. We chose to work with datasets containing only categorical attributes. Furthermore, we first sanitized some of the tables by removing duplicate rows and filling in missing values using the most common values.

In Table 1 we can find details from our experimentation using 24 categorical tables from the UCI Machine Learning Repository. Eight of these datasets were complete decision tables. The columns "Attr." and "Rows" contain the number of attributes and the number of rows in each dataset respectively. Column

Table Name	Attr.	Rows	Theorem 1	Tot. Opt.
adult-stretch	4	16	Yes	Yes
agaricus-lepiota	22	8124		No
balance-scale	4	625	Yes	Yes
breast-cancer	9	266		No
cars	6	1728	Yes	Yes
flags	26	194		No
hayes-roth-data	4	69		Yes
house-votes-84	16	279		No
lenses	4	24	Yes	Yes
lymphography	18	148		No
monks-1-test	6	432	Yes	Yes
monks-1-train	6	124		No
monks-2-test	6	432	Yes	Yes
monks-2-train	6	169		No
monks-3-test	6	432	Yes	Yes
monks-3-train	6	122		Yes
nursery	8	12960	No	Yes
shuttle-landing	6	15		No
soybean-small	35	47		No
spect-test	22	169		No
teeth	8	23		Yes
tic-tac-toe	9	958		No
zoo-data	16	59		No

Table 1: Details regarding tables from UCI ML Repository

"Theorem 1" is labeled with "Yes" for datasets where the system of equalities from Theorem 1 hold and "No" where they do not. The column is left blank for all datasets that were not complete decision tables. Finally, Column "Tot. Opt." states whether or not totally optimal decision rules exist for every row in the dataset. It is interesting to note that the "nursery" dataset, despite failing Theorem 1 still has totally optimal decision rules for each row. Further, many datasets for which Theorem 1 does not apply also have totally optimal decision rules for each row. The information for the "Tot. Opt." column was obtained by algorithms described in [3].

5. Conclusions

Decision rules can be optimized in many different ways. Two of these involve minimization of length and maximization of coverage. This paper considers the presence of decision rules that are simultaneously optimal with respect to both. Such rules negate the need to consider trade-offs and Pareto optimality. In particular, for complete decision tables, we provide a system of equations that can guarantee the presence of totally optimal rules for each row when true. One important aspect of this system of equations is the implication that it holds with regards to total Boolean functions and other total functions of k-valued logic ($k > 2$). All such functions are guaranteed to have at least one totally

optimal rule for each row.

6. Acknowledgements

Research reported in this publication was supported by King Abdullah University of Science and Technology (KAUST).

The authors are grateful to the anonymous reviewer for useful comments and suggestions.

[1] Alkhalid, A., Amin, T., Chikalov, I., Hussain, S., Moshkov, M., Zielosko, B., 2011. Dagger: A tool for analysis and optimization of decision trees and rules. In: Computational Informatics, Social Factors and New Information Technologies: Hypermedia Perspectives and Avant-Garde Experiences in the Era of Communicability Expansion. Blue Herons, pp. 29–39.

[2] Amin, T., Chikalov, I., Moshkov, M., Zielosko, B., 2013. Dynamic programming approach for exact decision rule optimization. In: Skowron, A., Suraj, Z. (Eds.), Rough Sets and Intelligent Systems. Professor Zdzisław Pawlak in Memoriam. Springer, pp. 211–228.

[3] Amin, T., Chikalov, I., Moshkov, M., Zielosko, B., 2014. Relationships between length and coverage of decision rules. Fundam. Inform. 129 (1-2), 1–13.

[4] Bache, K., Lichman, M., 2013. UCI machine learning repository.
URL `http://archive.ics.uci.edu/ml`

[5] Bazan, J. G., Szczuka, M. S., Wojna, A., Wojnarski, M., 2004. On the evolution of rough set exploration system. In: Tsumoto, S., Słowinski, R., Komorowski, H. J., Grzymała-Busse, J. W. (Eds.), RSCTC 2004. Vol. 3066 of LNCS. Springer, pp. 592–601.

[6] Boros, E., Crama, Y., Hammer, P., Ibaraki, T., Kogan, A., Makino, K., 2011. Logical analysis of data: classification with justification. Annals of Operations Research 188 (1), 33–61.

[7] Boros, E., Hammer, P., Ibaraki, T., Kogan, A., 1997. Logical analysis of numerical data. Mathematical Programming 79 (1), 163–190.

[8] Carbonell, J. G., Michalski, R. S., Mitchell, T. M., 1983. An overview of machine learning. In: Michalski, R. S., Carbonell, J. G., Mitchell, T. M. (Eds.), Machine Learning, An Artificial Intelligence Approach. Tioga Press, Palo Alto, CA.

[9] Clark, P., Niblett, T., 1989. The cn2 induction algoirthm. Mach. Learn. 3 (4), 261–283.

[10] Cortes, C., Vapnik, V., 1995. Support-vector networks. Mach. Learn. 20 (3), 273–297.

[11] Dasarathy, D. V. (Ed.), 1991. Nearest Neighbor Norms: NN Pattern Classification Techniques. IEEE Press, Los Alamos, CA.

[12] Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., 1996. From data mining to knowledge discovery in databases. AI Mag. 17, 37–54.

[13] Furnkranz, J., 1999. Separate-and-conquer rule learning. Artif. Intell. Rev. 13, 3–54.

[14] Grzymała-Busse, J. W., 1992. LERS – a system for learning from examples based on rough sets. In: Słowiński, R. (Ed.), Intelligent Decision Support. Handbook of Applications and Advances of the Rough Sets Theory. Kluwer Academic Publishers, pp. 3–18.

[15] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. H., 2009. The WEKA data mining software: an update. SIGKDD Explor. 11 (1), 10–18.

[16] Hammer, P. L., Kogan, A., Simeone, B., Szedmák, S., 2004. Pareto-optimal patterns in logical analysis of data. Discrete Appl. Math. 144 (12), 79 – 102.

[17] Moshkov, M., Piliszczuk, M., Zielosko, B., 2008. Partial Covers, Reducts and Decision Rules in Rough Sets – Theory and Applications. Vol. 145 of Studies in Computational Intelligence. Springer, Heidelberg.

[18] Moshkov, M., Zielosko, B., 2011. Combinatorial Machine Learning – A Rough Set Approach. Vol. 360 of Studies in Computational Intelligence. Springer, Heidelberg.

[19] Muggleton, S., 2000. Learning stochastic logic programs. Electron. Trans. Artif. Intell. 4 (B), 141–153.

[20] Øhrn, A., Komorowski, J., Skowron, A., Synak, P., 1998. The design and implementation of a knowledge discovery toolkit based on rough sets: The ROSETTA system. In: Rough Sets in Knowledge Discovery 1: Methodology and Applications. Vol. 18 of Studies in Fuzziness and Soft Computing. Physica-Verlag, pp. 376–399.

[21] Pawlak, Z., Skowron, A., 2007. Rough sets and boolean reasoning. Inf. Sci. 177 (1), 41–73.

[22] Quinlan, J. R., 1986. Induction of decision trees. Mach. Learn. 1 (1), 81–106.

[23] Quinlan, J. R., 1987. Simplifying decision trees. Int. J. Man. Mach. Stud. 27 (3), 221–234.

[24] Sikora, M., 2010. Decision rule-based data models using TRS and NetTRS – methods and algorithms. Trans. Rough Sets 11, 130–160.

[25] Ślęzak, D., Wróblewski, J., 2003. Order based genetic algorithms for the search of approximate entropy reducts. In: Wang, G., Liu, Q., Yao, Y., Skowron, A. (Eds.), RSFDGrC 2003. Vol. 2639 of LNCS. Springer, pp. 308–311.

[26] Zielosko, B., Chikalov, I., Moshkov, M., Amin, T., 2014. Optimization of decision rules based on dynamic programming approach. In: Innovations in Intelligent Machines (4). Springer, pp. 369–392.