Learning Decision Rules from Sets of Decision Trees

Mikhail Moshkov
King Abdullah University of Science and Technology (KAUST)/Computer Electrical and Mathematical Sciences & Engineering Division and Computational Bioscience Research Center
Thuwal, Saudi Arabia *mikhail.moshkov@kaust.edu.sa*

Beata Zielosko
University of Silesia in Katowice/Faculty of Science and Technology/Institute of Computer Science
Sosnowiec, Poland *beata.zielosko@us.edu.pl*

Evans Teiko Tetteh
University of Silesia in Katowice/Faculty of Science and Technology/Institute of Computer Science
Sosnowiec, Poland *evans.tetteh@us.edu.pl*

Anna Glid
University of Silesia in Katowice/Faculty of Science and Technology/Institute of Computer Science
Sosnowiec , Poland *anna.glid27@gmail.com*

Abstract

This paper is devoted to the study of the problems of learning inner and general decision rules that are true for the maximum number of decision trees from a given set. Inner rules correspond to paths in decision trees from the root to terminal nodes. General rules are arbitrary rules that use attributes from the considered decision trees. We propose a polynomial time algorithm for the optimization of inner rules, show that the problem of optimization of general rules is NP-hard, and describe a heuristic for this problem. We compare the considered algorithm and heuristic experimentally on artificially generated datasets and induced from them decision trees with Gini index as a splitting criterion.

Keywords: Decision Trees, Inner Decision Rules, General Decision Rules

1. Introduction

Decision trees and decision rules are widely used as classifiers, as a means of knowledge representation, and as algorithms [1, 3, 4, 5, 6, 7]. Learning decision trees from sets of decision rules is a nontrivial problem [2]. Learning decision rules from a decision tree is simple: we can "read" a decision rule when move from the root of the decision tree to its terminal node. In this paper, we show that there are complicated enough problems related to the learning decision rules from a given set of decision trees.

Let we have a finite set of decision trees S. We define two types of decision rules: general rules over S (arbitrary rules that use attributes from the decision trees belonging to S) and inner rules over S (rules corresponding to paths from the roots to terminal nodes in decision trees from S). A rule r is called true for a decision tree Γ from S if there exists an inner decision rule r' over Γ such that the set of elementary conditions in the left-hand side of r' is a subset of the set of elementary conditions in the left-hand side of r, and these rules have the same decision in the right-hand sides.

Our aim is to study two optimization problems: (i) to find an inner rule over S that is true for the maximum number of decision trees from S and (ii) to find a general rule over S that is true for the maximum number of decision trees from S.

Efficient algorithms for exact or approximate solving of these problems can be useful in the situation when different agents create their own experimental datasets using subsets of a common set of attributes and return knowledge about these datasets represented as decision trees. Rules that are true for the maximum number of decision trees can be considered as a common knowledge about these datasets. They allow us to discover major and general patterns hidden in the data. The novelty of the proposed way of induction of decision rules is its application to distributed knowledge.

We show that there is a polynomial (depending on the total number of nodes in the decision trees from S) algorithm $\mathcal{A}$, which can find an inner rule over S that is true for the maximum number of decision trees from S. However, this algorithm is not suitable if, for example, the decision trees from S have no common attributes and, hence, each inner rule is true for only one decision tree from S.

We prove that the problem of finding a general rule over S that is true for the maximum number of decision trees from S is NP-hard. In such a situation, we should consider polynomial time approximate algorithms for the optimization of general rules. We propose a heuristic $\mathcal{H}$ which uses the algorithm $\mathcal{A}$ iteratively and constructs a general rule over S.

The proposed heuristic has polynomial time complexity. However, we have not any theoretical bounds on the accuracy of this heuristic. One of the main aims of the paper is to compare the algorithm $\mathcal{A}$ and the heuristic $\mathcal{H}$ experimentally on artificially generated datasets and induced from them decision trees with Gini index as a splitting criterion.

The rest of the paper is organized as follows. In Section 2, we consider main notions and in Section 3 – the problem of optimization of inner rules. The problem of optimization of general rules is discussed in Sections 4 and 5. Section 6 contains results of computer experiments and Section 7 – conclusions.

2. Main Notions

Let $\omega = \{0, 1, 2, \ldots\}$ be the set of nonnegative integers and F be a set of attributes with values 0 and 1. A decision rule r over F is an expression of the form

$$(f_1 = \delta_1) \wedge \cdots \wedge (f_m = \delta_m) \to d,$$

where $m \in \omega$, $f_1, \ldots, f_m$ are pairwise different attributes from F, $\delta_1, \ldots, \delta_m \in \{0, 1\}$, and $d \in \omega$. The left-hand side of this rule is the conjunction of elementary conditions of the form $f_j = \delta_j$. We denote by $C(r)$ the set of these conditions. The right-hand side of r is the number d, which is interpreted as a decision. We denote it by $D(r)$. We will say that two rules r_1 and r_2 are incompatible if there exists an elementary condition $f = \delta$ such that $f = \delta$ belongs to $C(r_1)$ and $f = \neg\delta$ belongs to $C(r_2)$, where $\neg 0 = 1$ and $\neg 1 = 0$.

A decision tree over F is a marked directed tree with the root in which

- Each terminal node is labeled with a number from ω that is interpreted as a decision.
- Each nonterminal node (we call such nodes working) is labeled with an attribute from F. Each working node has two edges that leave this node and are labeled with the numbers 0 and 1, respectively.
- In each directed path from the root to a terminal node (we call such paths complete), attributes attached to working nodes are pairwise different.

Let Γ be a decision tree. Then the number of its terminal nodes is equal to the number of its working nodes plus 1. The number of complete paths is equal to the number of terminal nodes.

Let ξ be a complete path in Γ with m working nodes in which the terminal node is labeled with the decision d. We correspond to it a decision rule $rule(\xi)$. If $m = 0$, then this rule is equal to $\rightarrow d$. Let $m \geq 1$, the working nodes of ξ be labeled with the attributes $f_1, \ldots, f_m$, and edges leaving these nodes be labeled with the numbers $\delta_1, \ldots, \delta_m$, respectively. Then the rule $rule(\xi)$ is equal to

$$(f_1 = \delta_1) \wedge \cdots \wedge (f_m = \delta_m) \rightarrow d.$$

We denote by $\Xi(\Gamma)$ the set of complete paths in Γ and by $IR(\Gamma)$ we denote the set of decision rules $\{rule(\xi) : \xi \in \Xi(\Gamma)\}$ corresponding to complete paths in Γ. We will call the decision rules from $IR(\Gamma)$ inner decision rules over Γ.

Let S be a finite nonempty set of decision trees. Denote $IR(S) = \bigcup_{\Gamma \in S} IR(\Gamma)$, $F(S)$ – the set of attributes attached to working nodes of the decision trees from S, $D(S)$ – the set of decisions attached to terminal nodes of the decision trees from S, and $GR(S)$ – the set of decision rules over $F(S)$ that have decisions from $D(S)$ in their right-hand sides. Rules from $IR(S)$ will be called inner rules over S and rules from $GR(S)$ will be called general rules over S. It is clear that $IR(S) \subseteq GR(S)$.

Let Γ be a decision tree from the set S and r be a decision rule from $GR(S)$. We will say that the rule r is true for Γ if there exists an inner decision rule r' over Γ such that $D(r') = D(r)$ and $C(r') \subseteq C(r)$.

Remark 1 *If such an inner decision rule r' over Γ exists, then each other inner decision rule r'' over Γ and the decision rule r are incompatible.*

3. Optimization of Inner Decision Rules

In this section, we consider Inner Optimization Problem (IOP): for a given set of decision trees S, it is required to find an inner decision rule over S, which is true for the maximum number of decision trees from S.

Denote by $T(S)$ the total number of nodes in the decision trees from S. The number of inner rules in the set $IR(S)$ is at most $T(S)$. The number of elementary conditions in the left-hand side of each rule from $IR(S)$ is also at most $T(S)$. One can show that the set $IR(S)$ of inner rules can be constructed in a polynomial time depending on $T(S)$.

First, we describe an algorithm $\mathcal{A}_0$ that, for a given rule $r \in IR(S)$ and a decision tree $\Gamma \in S$, checks in a polynomial time depending on $T(S)$ if the rule r is true for the decision tree Γ.

Algorithm $\mathcal{A}_0$.

We sequentially compare the rule r with all rules from the set $IR(\Gamma)$. If there exists a rule $r' \in IR(\Gamma)$ such that $D(r') = D(r)$ and $C(r') \subseteq C(r)$, then the rule r is true for the decision tree Γ. Otherwise, the rule r is not true for the decision tree Γ.

We now consider an algorithm $\mathcal{A}$ that solves the problem IOP in a polynomial time depending on $T(S)$ if it is applied to the set $IR(S)$. We describe the work of $\mathcal{A}$ on an arbitrary subset I of the set $IR(S)$. In this case, the algorithm $\mathcal{A}$ returns in a polynomial time depending on $T(S)$ a rule from I that is true for the maximum number of decision trees from S.

Algorithm $\mathcal{A}$.

Using algorithm $\mathcal{A}_0$, for each rule $r \in I$ and each decision tree $\Gamma \in S$, we check if the rule r is true for the decision tree Γ. After that, we choose a rule from I, which is true for the maximum number of decision trees from S.

4. Optimization of General Decision Rules

In this section, we consider General Optimization Problem (GOP): for a given set of decision trees S it is required to find a general decision rule over S, which is true for the maximum number of decision trees from S. Let $F(S) = \{f_1, \ldots, f_m\}$. It is easy to show that a solution of GOP can be found among so-called complete decision rules over S, which are of the form

$$(f_1 = \delta_1) \wedge \cdots \wedge (f_m = \delta_m) \to d,$$

where $\delta_1, \ldots, \delta_m \in \{0, 1\}$ and $d \in D(S)$.

We now prove that GOP is NP-hard. To this end, we describe a polynomial time reduction of the NP-complete problem 3-SAT to GOP.

A 3-conjunctive normal form (3-CNF) is a Boolean formula of the form $N(x_1, \ldots, x_n) = C_1 \wedge \cdots \wedge C_k$ in which each clause C_j, $j = 1, \ldots, k$, is a disjunction of exactly three distinct literals and each literal is either a variable from the set $\{x_1, \ldots, x_n\}$ or a negation of a variable from this set. Each variable from the set $\{x_1, \ldots, x_n\}$ appears in at least one of the clauses $C_1, \ldots, C_k$.

3-Satisfiability Problem (3-SAT): for a given 3-CNF $N(x_1, \ldots, x_n)$, it is required to recognize if it is satisfiable, i.e, if there exists a tuple of values $(a_1, \ldots, a_n) \in \{0, 1\}^n$ of variables $x_1, \ldots, x_n$ such that $N(a_1, \ldots, a_n) = 1$.

For a given 3-CNF

$$N(x_1, \ldots, x_n) = C_1 \wedge \cdots \wedge C_k,$$

we construct a set $S_N = \{\Gamma_0, \Gamma_1, \ldots, \Gamma_k\}$ of decision trees corresponding to N. Let $i \in \{1, \ldots, k\}$ and $C_i = x_{t(i,1)}^{b_{i1}} \vee x_{t(i,2)}^{b_{i2}} \vee x_{t(i,3)}^{b_{i3}}$, where $x_{t(i,1)}, x_{t(i,2)}, x_{t(i,3)} \in \{x_1, \ldots, x_n\}$, $b_{i1}, b_{i2}, b_{i3} \in \{0, 1\}$, and $x^0 = \neg x$ and $x^1 = x$. It is clear that, for $b \in \{0, 1\}$, $x^b = 1$ if and only if $x = b$. An auxiliary decision tree Γ_0 and the decision tree Γ_i corresponding to the clause C_i are depicted in Fig. 1.

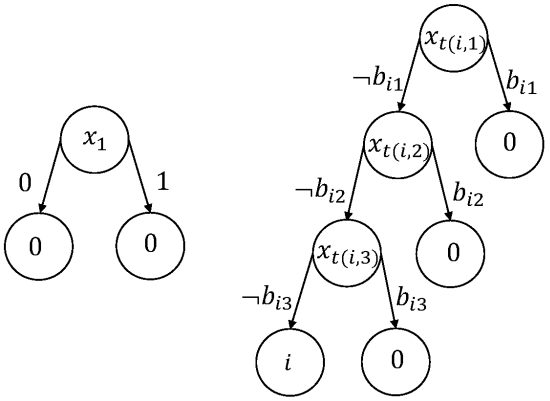

Fig. 1. Decision trees Γ_0 and Γ_i, $i = 1, \ldots, k$

We now show that the 3-CNF N is satisfiable if and only if there exists a complete decision rule over S_N, which is true for each decision tree from S_N.

Let there exist a tuple $(a_1, \ldots, a_n) \in \{0, 1\}^n$ such that $N(a_1, \ldots, a_n) = 1$. Then, for this tuple and each $i \in \{1, \ldots, k\}$, $C_i = 1$ and there exists $p_i \in \{1, 2, 3\}$ such that $a_{t(i,p_i)}^{b_{ip_i}} = 1$, i.e., $a_{t(i,p_i)} = b_{ip_i}$. We consider the following complete decision rule over S_N:

$$(x_1 = a_1) \wedge \cdots \wedge (x_n = a_n) \to 0.$$

One can show that this decision rule is true for each decision tree from S_N.

Let there exist a complete decision rule r over S_N, which is true for each decision tree from S_N. Then this rule has the decision 0 in the right-hand side. Let $x_1 = a_1, \ldots, x_n = a_n$ be all elementary conditions in the left-hand side of the rule r. Since this rule is true for each decision tree from S_N, for each $i \in \{1, \ldots, k\}$, there exists $p_i \in \{1, 2, 3\}$ such that $a_{t(i,p_i)} = b_{ip_i}$, i.e., $C_i = 1$. Thus, $N(a_1, \ldots, a_n) = 1$.

We now describe a polynomial time reduction of 3-SAT to GOP. Let N be a 3-CNF. We construct in a polynomial time depending on the number of literals in N the corresponding set of decision trees S_N and solve the problem GOP for S_N. As a result, we obtain a general decision rule r over S_N, which is true for the maximum number of decision trees from S_N. In a polynomial time, we check for each decision tree Γ from S_N if the rule r is true for Γ. If r is true not for each decision tree from S_N, then the 3-CNF N is not satisfiable. Otherwise, we can extend r to a complete decision rule over S_N, which is true for each decision tree from S_N. Therefore N is satisfiable.

5. Heuristic $\mathcal{H}$ for General Decision Rule Optimization

The algorithm $\mathcal{A}$ solves IOP in a polynomial time. However, the constructed rule can be true only for very small fraction of decision trees from S. Let, for example, each tree from S contain more than one node and sets of attributes attached to working nodes of decision trees from S be pairwise disjoint. Then each inner decision rule over S is true for only one decision tree from S. To improve this situation, we should use general decision rules over S. Since the problem of optimization of such rules is NP-hard, we propose a heuristics $\mathcal{H}$ for general decision rule optimization.

The proposed heuristic $\mathcal{H}$ uses an algorithm $\mathcal{B}$, which is based on the algorithm $\mathcal{A}$. This heuristic has polynomial time complexity depending on $T(S)$.

Heuristic $\mathcal{H}$.

For each decision $d \in D(S)$, we apply to the set of decision trees S the algorithm $\mathcal{B}$ described later. After that, among the constructed rules we choose one that is true for the maximum number of decision trees from S.

Algorithm $\mathcal{B}$.

Step 0. Construct the set I_0 that consists of all rules from $IR(S)$ in which the right-hand side is equal to d and the left-hand side is not empty. Construct the rule $\rightarrow d$ with empty left-hand side and denote this rule r_0.

Let step i, $i \geq 0$, be completed, and a set of decision rules I_i and a decision rule r_i be constructed.

Step $i + 1$. Apply to the set I_i the algorithm $\mathcal{A}$ and, as a result, obtain a decision rule ρ_{i+1}. Construct a decision rule r_{i+1} in which $D(r_{i+1}) = d$ and $C(r_{i+1}) = C(r_i) \cup C(\rho_{i+1})$. Remove from the set I_i all rules r such that r and r_{i+1} are incompatible and all rules r such that $C(r) \subseteq C(r_{i+1})$. Denote I_{i+1} the set of obtained rules. If $I_{i+1} = \emptyset$, then finish the work of the algorithm $\mathcal{B}$ and return the rule r_{i+1}. Otherwise, proceed to step $i + 2$.

Remark 2 *Using Remark 1, one can show that, for each step i of the algorithm $\mathcal{B}$, if the rule r_i is true for a decision tree Γ from S, then $I_i \cap IR(\Gamma) = \emptyset$.*

6. Results of Experiments

In this section, we compare the algorithm $\mathcal{A}$ and the heuristic $\mathcal{H}$ experimentally on artificially generated decision tables. Decision trees were constructed with stopping condition where all

instances associated with the terminal nodes have the same value of decision attribute and using Gini index as a splitting criterion. The experiments were performed using the Python language and ChefBoost framework [8].

Let's recall the main interpretation of the problem considered in the paper. We have n agents that study similar problems and use attributes from the common set containing m attributes. Each agent has its own decision table and returns a decision tree for this table. As a result, we have a set S of n decision trees. We need to find rules that are true for the maximum number of trees from S.

Two groups of experiments were performed. The main difference between them is in the choice of conditional attributes in decision tables for each agent. In the first group, we choose randomly 50% of attributes from the common set, in the second group - 100% of attributes from the common set. Denote $N = \{10, 20, 30, 40, 50\}$, $M_1 = \{10, 20, 30, 40, 50\}$, and $M_2 = \{5, 10, 15, 20, 25\}$.

We now describe the first group of experiments. Fix $n \in N$ and $m \in M_1$. Choose randomly $m/2$ attributes from the set $\{f_1, \ldots, f_m\}$ and construct a decision table with $m/2$ columns labeled with the chosen attributes and m pairwise different rows chosen randomly. Rows are filled with numbers from the set $\{0, 1\}$. They are labeled with decisions chosen randomly from the set $\{0, 1\}$. Using an algorithm based on Gini index, construct a decision tree for this decision table. Repeat the described step n times (each time, a new decision table is constructed). As a result, a set S_1 of n decision trees is obtained. Repeat the whole procedure 10 times. As a result, we obtain sets of decision trees $S_1, \ldots, S_{10}$. For $i = 1, \ldots, 10$, apply the algorithm $\mathcal{A}$ to S_i. As a result, we obtain a decision rule r_i. Let t_i be the number of trees from S_i for which the rule r_i is true. For the fixed m and n, the quality of the algorithm $\mathcal{A}$ is the number

$$\frac{t_1 + \cdots + t_{10}}{10n}.$$

For the fixed m and n, the quality of the heuristic $\mathcal{H}$ is obtained in the similar way. We find the quality of $\mathcal{A}$ and $\mathcal{H}$ for each $n \in N$ and $m \in M_1$. Figure 2 presents the described procedure for the first and the second groups of experiments.

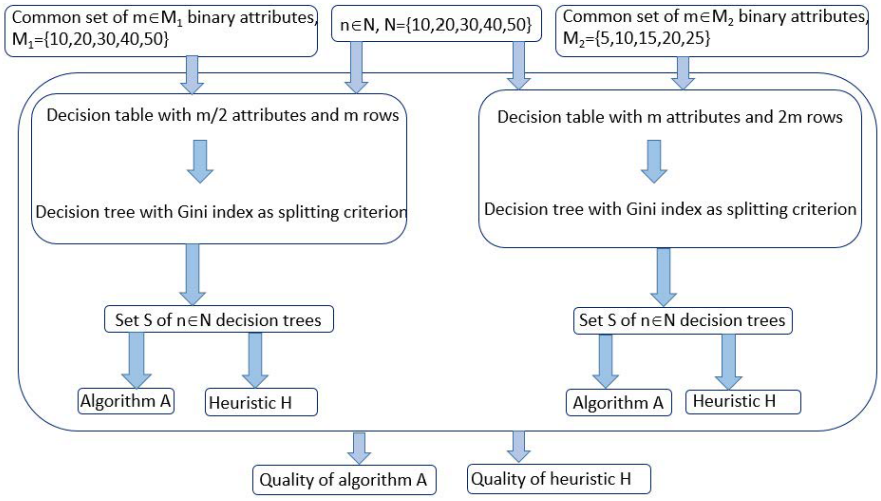

Fig. 2. Graphical presentation of the procedure of experiments

Table 1 presents results for the first group of experiments. At the intersection of row $m \in M_1$ and column $n \in N$ we have two numbers. The top one is the quality of $\mathcal{A}$ and the bottom one is the quality of $\mathcal{H}$ for m and n. The column Avg contains average values of qualities $\mathcal{A}$ and $\mathcal{H}$ for given m. The row Avg contains average values of qualities $\mathcal{A}$ and $\mathcal{H}$ for given n. At the

intersection of the row Avg and the column Avg, we have average values of qualities $\mathcal{A}$ and $\mathcal{H}$ for all m and n.

Table 1. The first group of experiments

		n					Avg
		10	20	30	40	50	
m	10	0.36	0.28	0.26	0.24	0.24	0.28
		0.82	0.75	0.69	0.69	0.66	0.72
	20	0.26	0.15	0.12	0.11	0.10	0.15
		0.95	0.82	0.74	0.71	0.67	0.78
	30	0.19	0.12	0.09	0.07	0.07	0.11
		0.96	0.84	0.79	0.75	0.69	0.81
	40	0.19	0.10	0.07	0.06	0.05	0.09
		0.99	0.92	0.82	0.78	0.74	0.85
	50	0.14	0.10	0.07	0.05	0.04	0.08
		0.99	0.94	0.84	0.82	0.77	0.87
	Avg	0.23	0.15	0.12	0.11	0.10	0.14
		0.94	0.85	0.78	0.75	0.71	0.80

We now describe the second group of experiments. It differs only in the way how the decision tables are constructed. Fix $n \in N$ and $m \in M_2$. Construct a decision table with m columns labeled with the attributes $f_1, \ldots, f_m$ and $2m$ pairwise different rows chosen randomly. Rows are filled with numbers from the set $\{0, 1\}$. They are labeled with decisions chosen randomly from the set $\{0, 1\}$. The remaining steps of the experiment procedure are the same as in the first group of experiments. Table 2 presents results for the second group of experiments. This table is structured like Table 1.

From the results presented in Tables 1 and 2 it follows that the heuristic $\mathcal{H}$ constructs rules which are true for a greater number of decision trees than the rules constructed by the algorithm $\mathcal{A}$. Taking into account the percentage of attributes from a common set (50% in case of the first group of experiments and 100% in case of the second group of experiments) it is possible to see that the algorithm $\mathcal{A}$ obtains on average better results in the case of Table 2. It follows from the fact that it is difficult to find an inner rule that is true for two decision trees if these trees use very different sets of attributes.

7. Conclusions

In this paper, we studied the problems of learning of inner and general decision rules that are true for the maximum number of decision trees from a given set. We proposed a polynomial time algorithm $\mathcal{A}$ for the first problem and proved that the second problem is NP-hard. We also proposed a polynomial time heuristic $\mathcal{H}$ for the construction of general rules. The algorithm $\mathcal{A}$ and the heuristic $\mathcal{H}$ were compared experimentally.

Decision rules are considered as popular and useful form of knowledge representation. They are easy accessible from the point of view of understanding and interpretation of knowledge represented by them. The novelty of the proposed way of induction of decision rules is its application to distributed knowledge and the fact that for constructed sets of decision trees, not all rules are taken into account but only those which are true for the maximum number of trees. Such an approach allows us to discover major patterns hidden in the data, especially when we work with data dispersed among different sources.

Among various applications for the proposed approach, we can distinguish a situation where a company or a hospital has several branches and each of them uses its own knowledge base.

Table 2. The second group of experiments

		n					Avg
		10	20	30	40	50	
m	5	0.71	0.67	0.66	0.65	0.63	0.66
		0.78	0.67	0.66	0.65	0.63	0.68
	10	0.39	0.29	0.29	0.28	0.30	0.31
		0.85	0.73	0.65	0.66	0.62	0.70
	15	0.26	0.18	0.16	0.13	0.13	0.17
		0.86	0.73	0.69	0.67	0.65	0.72
	20	0.21	0.15	0.10	0.10	0.09	0.13
		0.92	0.79	0.69	0.68	0.66	0.75
	25	0.20	0.11	0.09	0.07	0.07	0.11
		0.92	0.87	0.71	0.69	0.65	0.77
	Avg	0.35	0.28	0.26	0.25	0.25	0.28
		0.87	0.76	0.68	0.67	0.64	0.72

Each branch represents the collected knowledge in the form of a decision tree. The main goal is to obtain patterns that reflect the knowledge that is true for the most of the branches, or for the company/hospital as a whole.

In the future, it will be interesting to compare the length of the rules constructed by the algorithm $\mathcal{A}$ and by the heuristic $\mathcal{H}$. We are also planning to consider the problem of learning not only one rule (inner or general), which is true for the maximum number of decision trees from the considered set, but a group of rules each of which is true for a number of trees close to the maximum.

Acknowledgments

Research reported in this publication was supported by King Abdullah University of Science and Technology (KAUST). The authors are grateful to the anonymous reviewers for useful comments and suggestions.

References

1. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth and Brooks, Monterey, CA (1984)
2. Imam, I.F., Michalski, R.S.: Learning decision trees from decision rules: A method and initial results from a comparative study. J. Intell. Inf. Syst. 2(3), 279–304 (1993)
3. Moshkov, M.: Time complexity of decision trees. In: Peters, J.F., Skowron, A. (eds.) Trans. Rough Sets III, Lecture Notes in Computer Science, vol. 3400, pp. 244–459. Springer (2005)
4. Moshkov, M., Zielosko, B.: Combinatorial Machine Learning - A Rough Set Approach, Studies in Computational Intelligence, vol. 360. Springer, Heidelberg (2011)
5. Pawlak, Z.: Rough Sets - Theoretical Aspects of Reasoning about Data, Theory and Decision Library: Series D, vol. 9. Kluwer (1991)
6. Pawlak, Z., Skowron, A.: Rudiments of rough sets. Inf. Sci. 177 (1), 3–27 (2007)
7. Rokach, L., Maimon, O.: Data Mining with Decision Trees: Theory and Applications. World Scientific Publishing, River Edge, NJ (2008)
8. Serengil, S. I.: ChefBoost: A Lightweight Boosted Decision Tree Framework, https://doi.org/10.5281/zenodo.5576203. Accessed February 15, 2022 (2021)