Average Depth and Number of Misclassifications for Decision Trees

Igor Chikalov, Shahid Hussain, and Mikhail Moshkov

Mathematical and Computer Sciences & Engineering Division
King Abdullah University of Science and Technology
Thuwal 23955-6900, Saudi Arabia
{igor.chikalov, shahid.hussain, mikhail.moshkov}@kaust.edu.sa

Abstract. This paper presents a new tool for the study of relationships between total path length or average depth and number of misclafficiations for decision trees. In addition to algorithm, the paper also presents the results of experiments with datasets from UCI ML Repository [1].

1 Introduction

Decision trees are widely used as predictors, as a way of knowledge representation, and as algorithms for problem solving. There uses require optimizing decision trees for certain cost functions such as the number of misclassifications, depth/average depth, and number of nodes. That is minimizing one of these cost functions yields more accurate, faster, or more understandable decision trees (respectively).

We have created a software system for decision trees (as well as decision rules) called DAGGER—a tool based on dynamic programming which allows us to optimize decision trees (and decision rules) relative to various cost functions such as depth (length), average depth (average length), total number of nodes, and number of misclassifications sequentially [2–5]. The aim of this paper is to study the relationships between total path length (average depth) and number of misclaffications for decision trees and present a new tool (an extension of our software) for computing such relationships. We also consider the work of this tool on decision tables from UCI ML Repository [1].

The presented algorithm and its implementation in the software tool DAGGER together with similar algorithms devised by the authors (see for example [6]) can be useful for investigations in Rough Sets [7, 8] where decision trees are used as classifiers [9].

This paper is divided into six sections including the Introduction. Section 2 presents basic notions about decision tables/trees and cost functions. In Sect. 3 an algorithm for constructing a directed acyclic graph for representation of decision trees is presented. Section 4 presents the main algorithm and results for this paper. Section 5 shows experimental results of the work of the algorithm presented in this paper on different datasets and we conclude the paper in Sect. 6.

2 Basic Notions

In the following section we define the main notions related with the study of decision trees and tables and different cost functions for the decision tree construction.

2.1 Decision Tables and Decision Trees

In this paper, we consider only decision tables with discrete attributes. These tables do not contain missing values and equal rows. Consider a *decision table* T depicted in Fig. 1. Here $f_1, \ldots, f_m$ are the names of columns (conditional

$f_1 \ldots f_m$	d
$b_{11} \ldots b_{1m}$	c_1
$\ldots$	$\ldots$
$b_{N1} \ldots b_{Nm}$	c_N

Fig. 1. Decision table

attributes); $c_1, \ldots, c_N$ are nonnegative integers which can be interpreted as decisions (values of the decision attribute d); b_{ij} are nonnegative integers which are interpreted as values of conditional attributes (we assume that the rows $(b_{11}, \ldots, b_{1m}), \ldots, (b_{N1}, \ldots, b_{Nm})$ are pairwise different). We denote by $E(T)$ the set of attributes (columns of the table T), each of which contains different values. For $f_i \in E(T)$ let $E(T, f_i)$ be the set of values from the column f_i.

Let $f_{i_1}, \ldots, f_{i_t} \in \{f_1, \ldots, f_m\}$ and $a_1, \ldots, a_t$ be nonnegative integers. We denote by $T(f_{i_1}, a_1) \ldots (f_{i_t}, a_t)$ the subtable of the table T, which consists of such and only such rows of T that at the intersection with columns $f_{i_1}, \ldots, f_{i_t}$ have numbers $a_1, \ldots, a_t$ respectively. Such nonempty tables (including the table T) will be called *separable subtables* of the table T. For a subtable Θ of the table T, we will denote by $R(\Theta)$ the number of unordered pairs of rows that are labeled with different decisions. A minimum decision value which is attached to the maximum number of rows in a nonempty subtable Θ will be called the *most common decision* for Θ.

A *decision tree* Γ *over* the table T is a finite directed tree with root in which each terminal node is labeled with a decision. Each nonterminal node is labeled with a conditional attribute, and for each nonterminal node the outgoing edges are labeled with pairwise different nonnegative integers. Let v be an arbitrary node of Γ. We now define a subtable $T(v)$ of the table T. If v is the root then $T(v) = T$. Let v be a node of Γ that is not the root, nodes in the path from the root to v be labeled with attributes $f_{i_1}, \ldots, f_{i_t}$, and edges in this path be labeled with values $a_1, \ldots, a_t$ respectively. Then $T(v) = T(f_{i_1}, a_1), \ldots, (f_{i_t}, a_t)$.

Let Γ be a decision tree over T. We will say that Γ is a *decision tree for* T if any node v of Γ satisfies the following conditions:

– If $R(T(v)) = 0$ then v is a terminal node labeled with the most common decision for $T(v)$;
– Otherwise, v is either a terminal node labeled with the most common decision for $T(v)$, or v is labeled with an attribute $f_i \in E(T(v))$ and if $E(T(v), f_i) = \{a_1, \dots, a_t\}$, then t edges leave node v, and these edges are labeled with $a_1, \dots, a_t$ respectively.

Let Γ be a decision tree for T. For any row r of T, there exists exactly one terminal node v of Γ such that r belongs to the table $T(v)$. Let v be labeled with the decision b. We will say about b as about the *result of the work of decision tree Γ on r*. We denote by $N(T(v))$ the number of rows in the subtable $T(v)$ and $N(T(v), b)$ the number of rows in $T(v)$ labeled with decision b.

For an arbitrary row r of the decision table T, we denote by $l(r)$ the length of path from the root to a terminal node v of T such that r is in $T(v)$. We say that *the total path length*, represented as $\Lambda(Y, \Gamma)$, is the sum of path lengths $l(r)$ for all rows r in T. That is

$$\Lambda(T, \Gamma) = \sum_r l(r),$$

where we take the sum on all rows r of the table T. Note that the *average depth of Γ relative to T*, represented as $h_{\text{avg}}(T, \Gamma)$ is equal to the total path length divided by the total number of rows in T i.e.,

$$h_{\text{avg}}(T, \Gamma) = \frac{\Lambda(T, \Gamma)}{N(T)}.$$

We will drop T when it is obvious from the context. That is, we will write $\Lambda(\Gamma)$ instead of $\Lambda(T, \Gamma)$ if T is known.

The *number of misclassifications* for decision tree Γ for the table T, denoted as $\mu(\Gamma) = \mu(T, \Gamma)$, is the number of rows r in T for which the result of the work of decision tree Γ on r is not equal to the decision attached to the row r. We should note that the cost functions Λ and μ are bounded above by values depending upon the size of the table. That is, mN and N are upper bounds for Λ and μ for a decision table with m conditional attributes and N rows.

3 Representation of Sets of Decision Trees

Consider an algorithm for construction of a graph $\Delta(T)$, which represents the set of all decision trees for the table T. Nodes of this graph are some separable subtables of the table T. During each step we process one node and mark it with the symbol *. We start with the graph that consists of one node T and finish when all nodes of the graph are processed.

Let the algorithm has already performed p steps. We now describe the step number $(p + 1)$. If all nodes are processed then the work of the algorithm is finished, and the resulting graph is $\Delta(T)$. Otherwise, choose a node (table) Θ that has not been processed yet. If $R(\Theta) = 0$, label the considered node with the *common decision* b for Θ, mark it with symbol * and proceed to the step number $(p+2)$. If $R(\Theta) > 0$, then for each $f_i \in E(\Theta)$ draw a bundle of edges from the node Θ (this bundle of edges will be called f_i-*bundle*). Let $E(\Theta, f_i) = \{a_1, \ldots, a_t\}$. Then draw t edges from Θ and label these edges with pairs $(f_i, a_1), \ldots, (f_i, a_t)$ respectively. These edges enter into nodes $\Theta(f_i, a_1), \ldots, \Theta(f_i, a_t)$. If some of the nodes $\Theta(f_i, a_1), \ldots, \Theta(f_i, a_t)$ are not present in the graph then add these nodes to the graph. Mark the node Θ with the symbol * and proceed to the step number $(p+2)$.

Fig. 2. Trivial decision tree

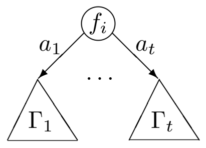

Fig. 3. Aggregated decision tree

Now for each node Θ of the graph $\Delta(T)$, we describe the set of decision trees corresponding to the node Θ. We will move from terminal nodes, which are labeled with numbers, to the node T. Let Θ be a node, which is labeled with a number b. Then the only trivial decision tree depicted in Fig. 2 corresponds to the node Θ.

Let Θ be a nonterminal node (table) then there is a number of bundles of edges starting in Θ. We consider an arbitrary bundle and describe the set of decision trees corresponding to this bundle. Let the considered bundle be an f_i-bundle where $f_i \in (\Theta)$ and $E(\Theta, f_i) = \{a_1, \ldots, a_t\}$. Let $\Gamma_1, \ldots, \Gamma_t$ be decision trees from sets corresponding to the nodes $\Theta(f_i, a_1), \ldots, \Theta(f_i, a_t)$. Then the decision tree depicted in Fig. 3 belongs to the set of decision trees, which correspond to this bundle. All such decision trees belong to the considered set, and this set does not contain any other decision trees. Then the set of decision trees corresponding to the node Θ coincides with the union of sets of decision trees corresponding to the bundles starting in Θ and the set containing one decision tree depicted in Fig. 2, where b is the most common decision for Θ. We denote by $D(\Theta)$ the set of decision trees corresponding to the node Θ.

The following proposition shows that the graph $\Delta(T)$ can represent all decision trees for the table T.

Proposition 1. *Let T be a decision table and Θ a node in the graph $\Delta(T)$. Then the set $D(\Theta)$ coincides with the set of all decision trees for the table Θ.*

4 Relationships

In the following, we consider relationships between average depth (total path length) and number of misclassifications for decision trees and give an algorithm to compute the relationships. We also provide an illustration of working of the algorithm on an example decision table.

Let T be a decision table with N rows and m columns labeled with $f_1, \dots, f_m$. Let Θ be a node of $\Delta(T)$ and $D(\Theta)$ be the set of all decision trees for Θ (as discussed in Sec. 2). We will use the notion of total path length instead of average depth for clarity and ease of implementation.

We define two sets $B_\Lambda = \{0, 1, \dots, mN\}$ and $B_\mu = \{0, 1, \dots, N\}$ and functions on these sets, $\mathcal{G}_\Theta : B_\Lambda \to B_\mu$ and $\mathcal{F}_\Theta : B_\mu \to B_\Lambda$ as following:

$$\mathcal{F}_\Theta = \min\{\mu(\Gamma) : \Gamma \in D(\Theta), \Lambda(\Gamma) \leq n\}, \;\; n \in B_\Lambda,$$
$$\mathcal{G}_\Theta = \min\{\Lambda(\Gamma) : \Gamma \in D(\Theta), \mu(\Gamma) \leq n\}, \;\; n \in B_\mu.$$

We now describe an algorithm which allows us to construct the function $\mathcal{F}_\Theta$ for every node Θ from the graph $\Delta(T)$. We begin from terminal nodes and move upward to the node T.

Let Θ be a terminal node. It means that all rows of Θ are labeled with the same decision b and the decision tree Γ_b as depicted in Fig. 2 belongs to $D(\Theta)$. It is clear that $\Lambda(\Gamma_b) = \mu(\Gamma_b) = 0$ for the table Θ. Therefore $\mathcal{F}_\Theta(n) = 0$ for any $n \in B_\Lambda$.

Let us consider a nonterminal node Θ and a bundle of edges, which start from this node. Let these edges be labeled with the pairs $(f_i, a_1), \dots, (f_i, a_t)$ and enter into the nodes $\Theta(f_i, a_1), \dots, \Theta(f_i, a_j)$, respectively, to which the functions $\mathcal{F}_{\Theta(f_i,a_1)}, \dots, \mathcal{F}_{\Theta(f_i,a_t)}$ are already attached.

We correspond to this bundle (f_i-bundle) the function $\mathcal{F}_\Theta^{f_i}$ for any $n \in B_\Lambda$ $n \geq N(\Theta)$,

$$\mathcal{F}_\Theta^{f_i}(n) = \min \sum_{j=1}^{t} \mathcal{F}_{\Theta(f_i,a_j)}(n_j),$$

where the minimum is taken over all $n_1, \dots, n_t$ such that $n_j \in B_\Lambda$ for $j = 1, \dots, t$ and $n_1 + \dots + n_t \leq n - N(\Theta)$. [It should be noted that computing $\mathcal{F}_\Theta^{f_i}$ is a nontrivial task. We describe the method in detail in the following subsection.] Furthermore, we know that there is only one decision tree in $D(\Theta)$ for which the total path length is at most $N(\Theta) - 1$. This is the tree Γ_b as depicted in Fig. 2, where b is *the most common decision* for Θ, therefore, for any $n \in B_\Lambda$, $n < N(\Theta)$

$$\mathcal{F}_\Theta^{f_i}(n) = N(\Theta) - N(\Theta, b).$$

We can use the following proposition to construct the function $\mathcal{G}_T$ (using the method of transformation of functions described in the Appendix).

Proposition 2. *For any* $n \in B_\mu$,

$$\mathcal{G}_T(n) = \min\{p \in B_\Lambda : \mathcal{F}_T(p) \leq n\}.$$

Note that to find the value $\mathcal{G}_T(n)$ for some $n \in B_\mu$ it is enough to make $O(\log |B_\mu|) = O(\log N)$ operations of comparisons.

4.1 Computing $\mathcal{F}_{\Theta}^{f_i}$

Let Θ be a nonterminal node in $\Delta(T)$, $f_i \in E(\Theta)$ and $E(\Theta, f_i) = \{a_1, \dots, a_t\}$. Furthermore, we assume that the functions $\mathcal{F}_{\Theta(f_i,a_j)}$, for $j = 1, \dots, t$ have already been computed. Let the values of $\mathcal{F}_{\Theta(f_i,a_j)}$ be given by the tuple of pairs

$$\Big((0, \mu_0^j), (1, \mu_1^j), \dots, (mN, \mu_{mN}^j)\Big)$$

here $\mu_k^j = \mathcal{F}_{\Theta(f_i,a_j)}(k)$ for $k = 0, \dots, mN$. We need to compute $\mathcal{F}_{\Theta}^{f_i}$ for all $n \in B_\Lambda$, $n \geq N(\Theta)$,

$$\mathcal{F}_{\Theta}^{f_i}(n) = \min \sum_{j=1}^{t} \mathcal{F}_{\Theta(f_i,a_t)}(n_j),$$

for $n_j \in B_\Lambda$ such that $n_1 + \cdots + n_t \leq n - N(\Theta)$.

We construct a layered directed acyclic graph (DAG) $\delta(\Theta, f_i)$ to compute $\mathcal{F}_{\Theta}^{f_i}$ as following. The DAG $\delta(\Theta, f_i)$ contains nodes arranged in $t+1$ layers $(l_0, l_1, \dots, l_t)$. Each node has a pair of labels and each layer l_j $(1 \leq j \leq t)$ contains $jmN + 1$ nodes. The first entry of labels for nodes in a layer l_j is an integer from $\{0, 1, \dots, jmN\}$. The layer l_0 contains only one node labeled with $(0, 0)$.

Each node in layer l_j $(0 \leq j < t)$ has $mN + 1$ outgoing edges to nodes in layer l_{j+1}. These edges are labeled with corresponding pairs in $\mathcal{F}_{\Theta(f_i,a_{j+1})}$. A node with label x as a first entry in its label-pair in a layer l_j connects to nodes with labels $x + 0$ to $x + mN$ (as a first entry in their label-pairs) in layer l_{j+1}, with edges labeled as $\Big((0, \mu_0^{j+1}), (1, \mu_1^{j+1}), \dots, (mN, \mu_{mN}^{j+1})\Big)$, respectively.

The function $\mathcal{F}_{\Theta}^{f_i}(n)$ for $n \in B_\Lambda \setminus \{0\}$ can be easily computed using the DAG $\delta(\Theta, f_i)$ for $\Theta \in \Delta(T)$ and for the considered bundle of edges for the attribute $f_i \in E(\Theta)$ as following:

Each node in layer l_1 gets its second value copied from the corresponding second value from the incoming edge label to the node (since there is only one incoming edge for each node in layer l_1). Let (k, μ) be a node in layer l_j, $(2 \leq j \leq t)$. Let $E = \{(\lambda_1, \mu_1), (\lambda_2, \mu_2), \dots, (\lambda_r, \mu_r)\}$ be the set of incoming nodes to (k, μ) such that $(\alpha_1, \beta_1), (\alpha_2, \beta_2), \dots, (\alpha_r, \beta_r)$ are the labels of these edges between the nodes in E and (k, μ), respectively. It is clear that $k = \lambda_i + \alpha_i$, $(1 \leq i \leq r)$. We set

$$\mu = \min_{1 \leq i \leq r} \{\mu_i + \beta_i\}.$$

We do this for every node layer-by-layer till all nodes in $\delta(\Theta, f_i)$ have received their second label.

Once we finish computing the second value of label-pairs of layer l_t, we can use these labels to compute $\mathcal{F}_{\Theta}^{f_i}(n)$. Let $(k_1, \mu_1), \dots, (k_s, \mu_s)$ be all label-pairs to the nodes in l_t. One can show that for all $n \in B_\Lambda$, $n \geq N(\Theta)$,

$$\mathcal{F}_{\Theta}^{f_i}(n) = \min\{\mu_q : q \in \{1, \dots, s\}, k_q \leq n - N(\Theta)\}.$$

An example of working of the algorithm can be found in the Fig. 4. It is important to note that in this example we only show the values for total path lengths possible for the set of trees that can be constructed for the given subtable (table) and their corresponding number of misclassifications.

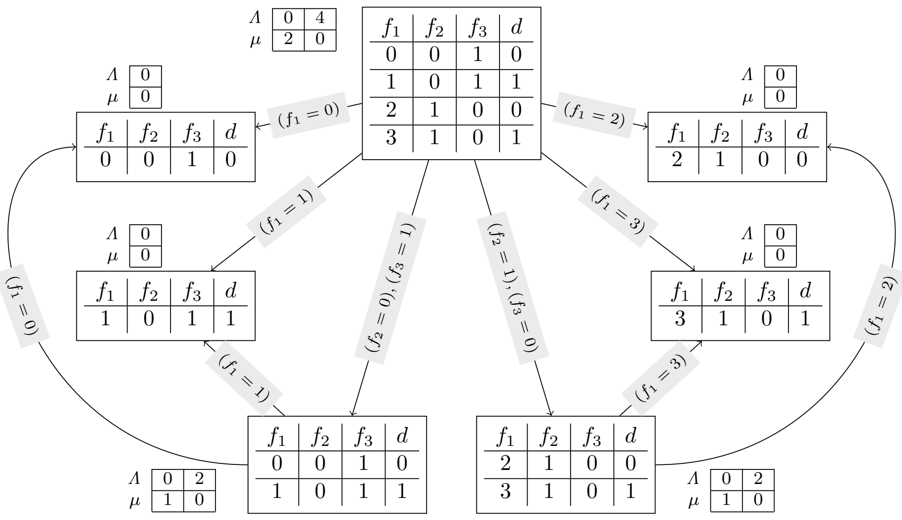

Fig. 4. Example DAG with relationship values

5 Experimental Results

We performed several experiments on datasets (decision tables) acquired from UCI ML Repository [1]. The resulting plots are depicted in Figs. 5, 6, 7, and 8.

It is clear to see in above figures that when the total path length is zero we get maximum number of misclassifications (as it is the base case) which makes the figures less clear in other regions. We remove the $(0, 67)$ point from Fig. 5 and show the resulting plot for the average depth instead of total path length in Fig. 9. Similarly, Fig. 10 shows in inverse of plot shown in Fig. 6 without the base case $(0, 3916)$.

6 Conclusions

This paper is devoted to the consideration of a new tool for decision tree study. We present and explain in detail the algorithm to compute the relationships between the total path length (average depth) and number of misclassifications for decision trees. That is, for a given decision table we can find minimum number of misclassifications for a given value for total path length when we consider the

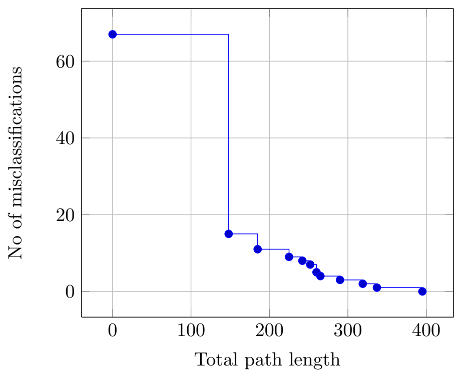

Fig. 5. LYMPHOGRAPHY dataset (18 attributes and 148 rows)

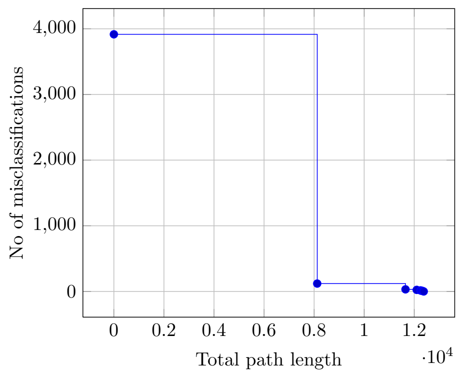

Fig. 6. MUSHROOM dataset (22 attributes and 8124 rows)

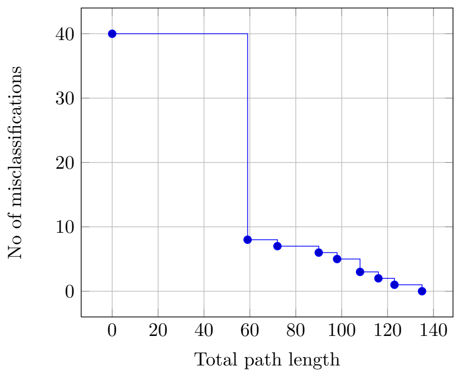

Fig. 7. ZOO-DATA dataset (16 attributes and 101 rows)

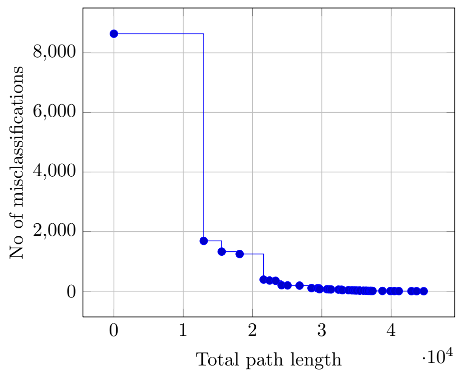

Fig. 8. NURSERY dataset (8 attributes and 12960 rows)

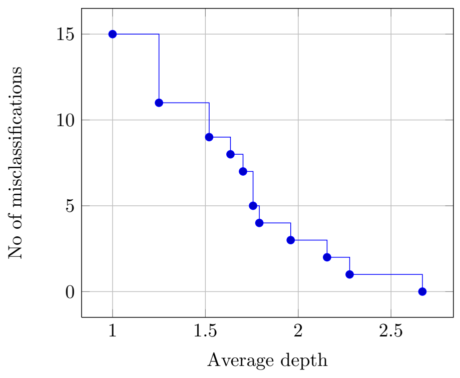

Fig. 9. LYMPHOGRAPHY dataset (18 attributes and 148 rows)

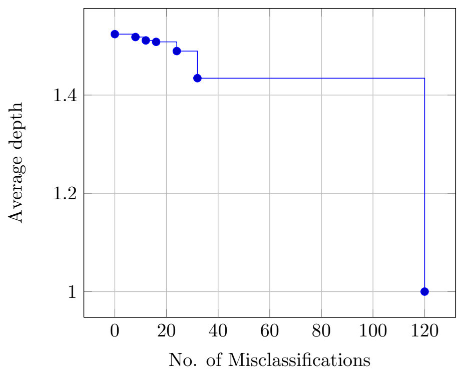

Fig. 10. MUSHROOM dataset (22 attributes and 8124 rows)

exact decision trees. Future studies will be connected with the extension of this tool to other cost functions specially the relationships between the depth and average depth of decision trees.

References

1. Frank, A., Asuncion, A.: UCI Machine Learning Repository (2010)
2. Alkhalid, A., Chikalov, I., Moshkov, M.: On algorithm for building of optimal α-decision trees. In Szczuka, M.S., Kryszkiewicz, M., Ramanna, S., Jensen, R., Hu, Q., eds.: RSCTC 2010, LNCS (LNAI). Volume 6086., Heidelberg, Springer (2010) 438–445
3. Alkhalid, A., Chikalov, I., Moshkov, M.: A tool for study of optimal decision trees. In Yu, J., Greco, S., Lingras, P., Wang, G., Skowron, A., eds.: RSKT. Volume LNCS 6401., Springer (2010) 353–360
4. Alkhalid, A., Chikalov, I., Hussain, S., Moshkov, M. In: Extensions of dynamic programming as a new tool for decision tree optimization. Volume 13 of SIST. Springer, Heidelberg (2012) 16–36
5. Alkhalid, A., Amin, T., Chikalov, I., Hussain, S., Moshkov, M., Zielosko, B.: 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 Experiencies in the Era of Communicability Expansion. Blue Herons (2011) 29–39
6. Chikalov, I., Hussain, S., Moshkov, M.: Relationships between depth and number of missclassifications for decision trees. In Kuznetsov, S.O., Slezak, D., Hepting, D.H., Mirkin, B., eds.: Thirteenth International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granualr Computing (RSFDGrC 2011). Volume LNCS 6743., Springer (2011) 286–292
7. Pawlak, Z.: Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers, Dordrecht (1991)
8. Skowron, A., Rauszer, C.: The discernibility matrices and functions in information systems. In Slowinski, R., ed.: Intelligent Decision Support. Handbook of Applications and Advances of the Rough Set Theory, Dordrecht, Kluwer Academic Publishers (1992) 331–362
9. Nguyen, H.S.: From optimal hyperplanes to optimal decision trees. Fundamenta Informaticae **34**(1-2) (1998) 145–174

Appendix: Transformation of Functions

Let f and g be two functions from a set A onto C_f and C_g respectively, where C_f and C_g are finite sets of nonnegative integers. Let $B_f = \{m_f, m_f + 1, \dots, M_f\}$ and $B_g = \{n_g, n_g + 1, \dots, N_g\}$ where $m_f = \min\{m : m \in C_f\}$ and $n_g = \min\{n : n \in C_g\}$. Furthermore, M_f and N_g are natural numbers such that $m \leq M_f$ and $n \leq N_g$ for any $m \in C_f$ and $n \in C_g$, respectively.

We define two functions $\mathcal{F} : B_g \to B_f$ and $\mathcal{G} : B_f \to B_g$ as following:

$$\mathcal{F}(n) = \min\{f(a) : a \in A, g(a) \leq n\}, \ \forall n \in B_g, \tag{1}$$

$$\mathcal{G}(m) = \min\{g(a) : a \in A, f(a) \leq m\}, \ \forall m \in B_f. \tag{2}$$

It is clear that both $\mathcal{F}$ and $\mathcal{G}$ are nonincreasing functions.

The following proposition states that the functions $\mathcal{F}$ and $\mathcal{G}$ can be used interchangeably and we can evaluate $\mathcal{F}$ using $\mathcal{G}$ and vice versa, i.e., it is enough to know only one function to evaluate the other.

Proposition 3. *For any $n \in B_g$,*

$$\mathcal{F}(n) = \min\{m \in B_f : \mathcal{G}(m) \leq n\},$$

and for any $m \in B_f$,

$$\mathcal{G}(m) = \min\{n \in B_g : \mathcal{F}(n) \leq m\}.$$

Proof. Let for some $n \in B_g$

$$\mathcal{F}(n) = m_0. \tag{3}$$

Furthermore, we assume that

$$\min\{m \in B_f : \mathcal{G}(m) \leq n\} = t. \tag{4}$$

From (3) it follows that

(i) there exists $b \in A$ such that $g(b) \leq n$ and $f(b) = m_0$;
(ii) for any $a \in A$ if $g(a) \leq n$ then $f(a) \geq m_0$.

From (i) it follows that $\mathcal{G}(m_0) \leq n$. This implies $t \leq m_0$. Let us assume that $t < m_0$. In this case, there exits $m_1 < m_0$ for which $\mathcal{G}(m_1) \leq n$. Therefore, there exists $a \in A$ such that $f(a) \leq m_1$ and $g(a) \leq n$, but from (ii) it follows that $f(a) \geq m_0$, which is impossible. So $t = m_0$.

Similarly we can prove the second part of the statement.

Proposition 3 allows us to transform the function $\mathcal{G}$ given by a tuple $(\mathcal{G}(m_f), \mathcal{G}(m_f + 1), \ldots, \mathcal{G}(M_f))$ into the function $\mathcal{F}$ and vice versa. We know that $\mathcal{G}(m_f) \geq \mathcal{G}(m_f + 1) \geq \cdots \geq \mathcal{G}(M_f)$, to find the minimum $m \in B_f$ such that $\mathcal{G}(m) \leq m$ we can use binary search which requires $O(\log|B_f|)$ comparisons of numbers. So to find the value $\mathcal{F}(n)$ for $n \in B_g$ it is enough to make $O(\log|B_f|)$ operations of comparison.