Relationships Between Average Depth and Number of Nodes for Decision Trees

Igor Chikalov, Shahid Hussain and Mikhail Moshkov

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

Keywords Decision trees · Number of nodes · Total path length · Average depth

1 Introduction

Decision trees are widely used as predictors, as a way of knowledge representation, and as algorithms for problem solving. These 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

I. Chikalov · S. Hussain (✉) · M. Moshkov
Computer, Electrical and Mathematical Sciences and Engineering, King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia
e-mail: shahid.hussain@kaust.edu.sa

I. Chikalov
e-mail: igor.chikalov@kaust.edu.sa

M. Moshkov
e-mail: mikhail.moshkov@kaust.edu.sa

F. Sun et al. (eds.), *Knowledge Engineering and Management*,
Advances in Intelligent Systems and Computing 214,
DOI: 10.1007/978-3-642-37832-4_47,

of misclassifications sequentially [2–5]. The aim of this paper is to study the relationships between total path length (average depth) and number of nodes of 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 Introduction. Section 2 presents some basic notions related to decision tables, decision trees, and the two cost functions. Section 3 gives an algorithm to construct a directed acyclic graph (DAG) $\Delta(T)$ that captures all possible decision trees for a given decision table T. The main algorithm for computing relationships between total path length (average depth) and number of nodes is presented in Sect. 4. Section 5 shows some experimental results of work of the algorithm on data tables acquired from UCI ML Repository and Sect. 6 concludes the paper followed by References and an appendix for "transformation of functions" proposition used in Sect. 4.

2 Basic Notions

In the following section we define the main notions related to the study of decision trees and tables and two cost functions for decision trees.

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 conditional attributes; $c_1, \ldots, c_N$ are nonnegative integers which can be interpreted as the 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

Fig. 1 Decision table

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

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. We denote by $N(T)$ the number of rows in the decision table T.

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 *decision tree* Γ *over* the table T is a finite directed tree with a 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. We 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 common decision for $T(v)$.
- Otherwise, v is labeled with an attribute $f_i \in E(T(v))$ and, if $E(T(v), f_i) = \{a_1, \ldots, a_t\}$, then t edges leave node v, and these edges are labeled with $a_1, \ldots, a_t$ 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 the *result of the work of decision tree* Γ *on* r.

For an arbitrary row r of the decision table T, we denote by $l(r)$ *the length of path* from the root to the terminal node v of T such that r is in $T(v)$. We say that *the total path length*, represented as Λ, is the sum of path lengths for all rows 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 *average depth*, represented as h_{avg} of Γ 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.

For a decision tree Γ of a decision table T, we represent the total number of nodes of Γ by $L(\Gamma)$. It is interesting to note that the cost functions Λ and L are

bounded above by values depending upon the size of the table. That is, mN and $2N - 1$ are the upper bounds for Λ and L 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.

Assume 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 $(p + 2)$. 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.

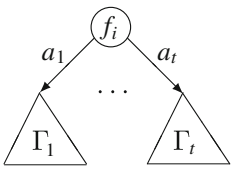
Let Θ be a nonterminal node (table) then there is a number of bundles of edges starting in Θ. We consider an arbitrary bundle and describe the set of decision trees corresponding to this bundle. Let the considered bundle be an f_i-bundle where $f_i \in (\Theta)$ and $E(\Theta, f_i) = \{a_1, \ldots, a_t\}$. 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. 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.*

Fig. 2 Trivial DT

Fig. 3 Aggregated DT



Proof We prove this proposition by induction on nodes in the graph $\Delta(T)$. For each terminal node Θ, only one decision tree exists as depicted in Fig. 2, and the set $D(T)$ contains only this tree. Let Θ be a nonterminal node and the statement of proposition hold for all its descendants.

Consider an arbitrary decision tree $\Gamma \in D(\Theta)$. Obviously, Γ contains more than one node. Let the root of Γ be labeled with an attribute f_i and the edges leaving root be labeled with the numbers $a_1, \ldots, a_t$. For $j = 1, \ldots, t$, denote by Γ_j the decision tree connected to the root with the edge labeled with the number a_j. From the definition of the set $D(\Theta)$ it follows that f_i is contained in the set $E(\Theta)$, $E(\Theta, f_i) = \{a_1, \ldots, a_t\}$ and for $j = 1, \ldots, t$, the decision tree Γ_j belongs to the set $D(\Theta(f_i, a_j))$. According to the inductive hypothesis, the tree Γ_j is a decision tree for the table $\Theta(f_i, a_j)$. Then the tree Γ is a decision tree for the table Θ.

Now we consider an arbitrary decision tree Γ for the table Θ. According to the definition, the root of Γ is labeled with an attribute f_i from the set $E(\Theta)$, edges leaving the root are labeled with numbers from the set $E(\Theta, f_i)$ and the subtrees whose roots are nodes, to which these edges enter, are decision trees for corresponding descendants of the node Θ. Then, according to the definition of the set $D(\Theta)$ and to inductive hypothesis, the tree Γ belongs to the set $D(\Theta)$.

4 Relationships

In the following we consider relationships between average depth (total path length) and number of nodes 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 $f_1, \ldots, f_m$, and $D(T)$ be the set of all decision trees for T (as discussed in Sects. 2 and 3). We will use the notion of total path length instead of average depth for clarity and ease of implementation.

We denote $B_{\Lambda,T} = \{\beta, \beta + 1, \ldots, mN\}$ and $B_{L,T} = \{\alpha, \alpha + 1, \ldots, 2N - 1\}$, here $\beta = \beta(T)$ and $\alpha = \alpha(T)$ are minimum total path length and minimum number of nodes, respectively, of some decision tree in $D(T)$ (not necessarily the same tree). We define two functions $\mathcal{G}_T : B_{\Lambda,T} \to B_{L,T}$ and $\mathcal{F}_T : B_{L,T} \to B_{\Lambda,T}$ as follows:

$$\mathcal{F}_T(n) = \min\{\Lambda(\Gamma) : \Gamma \in D(T) : L(\Gamma) \leq n\}, \quad n \in B_{L,T}$$
$$\mathcal{G}_T(n) = \min\{L(\Gamma) : \Gamma \in D(T) : \Lambda(\Gamma) \leq n\}, \quad n \in B_{\Lambda,T}.$$

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 the rows of decision table 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 $\Lambda(\Gamma_b) = 0$ and $L(\Gamma_b) = 1$ for the table Θ as well as $\alpha(\Theta) = 1$, therefore, $\mathcal{F}_\Theta(n) = 0$ for any $n \in B_{L,\Theta}$.

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

Let $v_1, \ldots, v_t$ be the minimum values from $B_{L,\Theta(f_i,a_1)}, \ldots, B_{L,\Theta(f_i,a_t)}$, respectively. Let

$$B_{L,\Theta,f_i} = \{\alpha_i, \alpha_i + 1, \ldots, 2N - 1\}, \quad \text{where } \alpha_{\mathrm{i}} = 1 + \sum_{\mathrm{j}=1}^{\mathrm{t}} v_{\mathrm{j}}.$$

One can show that α_i is the minimum number of nodes of a decision tree from $D(\Theta)$ for which f_i is attached to the root and $\alpha(\Theta) = \min\{\alpha_i : f_i \in E(\Theta)\}$, where $\alpha(\Theta)$ is the minimum value from $B_{L,\Theta}$.

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

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

where the minimum is taken over all $n_1, \ldots, n_t$ such that $n_j \in B_{L,\Theta(f_i,a_j)}$ for $j = 1, \ldots, t$ and $n_1 + \cdots + n_t + 1 \leq n$. [It should be noted that computing $\mathcal{F}^{f_i}_\Theta$ is a nontrivial task. We describe the method in detail in the following subsection.] It is not difficult to show that for all $n \in B_{L,\Theta}$,

$$\mathcal{F}_\Theta(n) = \min\{\mathcal{F}^{f_i}_\Theta(n) : f_i \in E(\Theta), n \in B_{L,\Theta,f_i}\}.$$

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

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

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

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

computed. Let the values of $\mathcal{F}_{\Theta(f_i,a_j)}$ be given by the tuple of pairs, $\left((\gamma_j, \lambda^j_{\gamma_j}), (\gamma_j+1, \lambda^j_{\gamma_j+1}), \ldots, (2N-1, \lambda^j_{2N-1})\right)$, where $\gamma_j = \alpha(\Theta(f_i,a_j))$ and $\lambda^k_j = \mathcal{F}_{\Theta(f_i,a_j)}(k)$. We need to compute $\mathcal{F}^{f_i}_{\Theta}(n)$ for all $n \in B_{L,\Theta,f_j}$;

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

for $n_j \in B_{L,\Theta(f_i,a_j)}$, such that $n_1 + \cdots + n_t + 1 \leq n$.

We construct a layered directed acyclic graph (DAG) $\delta(\Theta,f_i)$ to compute $\mathcal{F}^{f_i}_{\Theta}$ as following. The DAG $\delta(\Theta,f_i)$ contains nodes arranged in $t+1$ layers $(l_0, l_1, \ldots, l_t)$. Each node has a pair of labels and each layer $l_j (1 \leq j \leq t)$ contains at most $j(2N-1)$ nodes. The first entry of labels for nodes in a layer l_j is an integer from $\{1, 2, \ldots, j(2N-1)\}$. The layer l_0 contains only one node labeled with $(0,0)$.

Each node in a layer l_j $(0 \leq jt)$ has at most $2N-1$ outgoing edges to nodes in layer l_{j+1}. These edges are labeled with the 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 $x+\gamma_j$ to $x+2N-1$ (as a first entry in their label-pairs) in layer l_{j+1}, with edges labeled as $(\gamma_{j+1}, \lambda^{j+1}_{\gamma_{j+1}}), (\gamma_{j+1}+1, \lambda^{j+1}_{\gamma_{j+1}+1}), \ldots, (2N-1, \lambda^{j+1}_{2N-1})$, respectively.

The function $\mathcal{F}^{f_i}_{\Theta}(n)$ for $n \in B_L$ 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 follows:

Each node in layer l_1 gets its second value copied from the corresponding second value in incoming edge label to the node (since there is only one incoming edge for each node in layer l_1). Let (k,λ) be a node in layer l_j, $2 \leq j \leq t$. Let $E = \{(v_1,\lambda_1), (v_2,\lambda_2), \ldots, (v_r,\lambda_r)\}$ be the set of incoming nodes to (k,λ) such that $(\alpha_1,\beta_1), (\alpha_2,\beta_2), \ldots, (\alpha_r,\beta_r)$ are the labels of these edges between the nodes in E and (k,λ), respectively. It is clear that $k = v_i + \alpha_i$, $1 \leq i \leq r$. Then $\lambda = \min_{1 \leq i \leq r}\{\lambda_i + \beta_i\}$. 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 for the nodes in layer l_t, we can use these labels to compute $\mathcal{F}^{f_i}_{\Theta}(n)$. Let $(k_1,\lambda_1), \ldots, (k_s,\lambda_s)$ be all label-pairs attached to the nodes in l_t. One can show that

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

An example of working of the algorithm can be found in Fig. 4.

Let us evaluate the number of arithmetic operations of the considered algorithm. The DAG $\delta = \delta(\Theta,f_i)$ has $t+1$ layers and each layer l_j has at most $j(2N-1)$ nodes. Therefore, the total number of nodes in δ is $O(t^2N)$. Since every node has at most $2N-1$ outgoing edges (except the nodes in layer l_t), the number of edges in δ is $O(t^2N^2)$. Hence, to build the graph δ, we need $O(t^2N^2)$ operations. To find the second labels we need a number of additions and comparisons bounded from

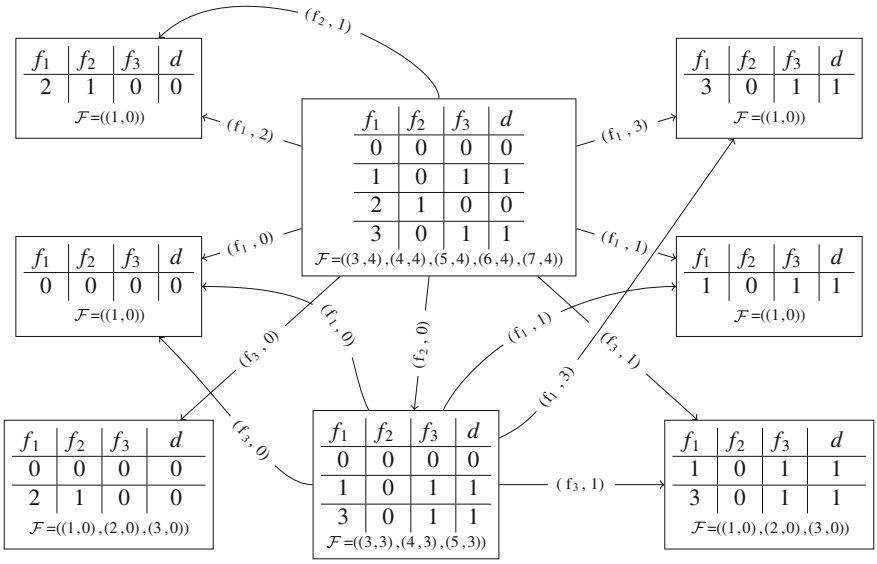


Fig. 4 Example illustrating the working of the algorithm

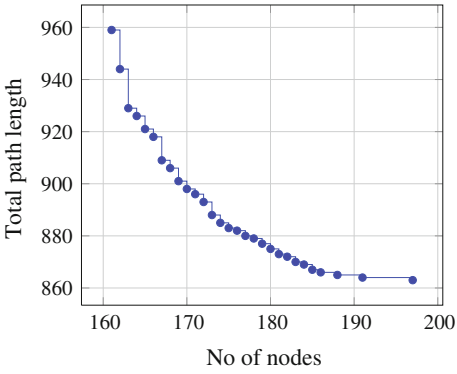


Fig. 5 BREAST-CANCER dataset (10 attributes and 267 rows)

above by the number of edges, i.e, $O(t^2N^2)$. Similarly, to find values of $\mathcal{F}_{\Theta}^{f_i}$ we need $O(tN)$ comparisons. Therefore, the total number of additions and comparisons is $O(t^2N^2)$.

5 Experimental Results

We performed several experiments on datasets (decision tables) acquired from UCL ML Repository [1]. The resulting plots are depicted in Figs. 5, 6, 7, and 8. These plots show the relationship between two cost functions, take for example the

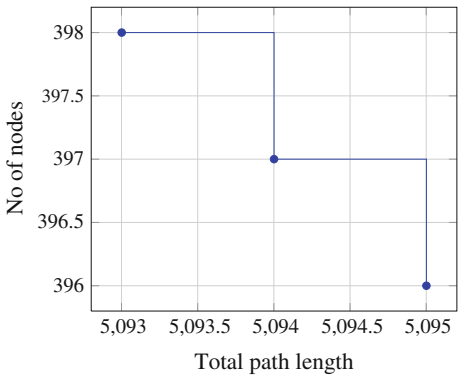


Fig. 6 CARS dataset (6 attributes and 1729 rows)

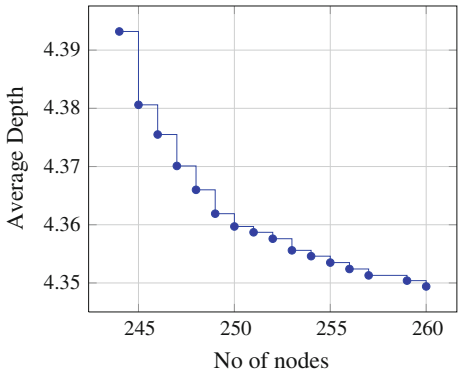


Fig. 7 TIC-TAC-TOE dataset (9 attributes and 959 rows)

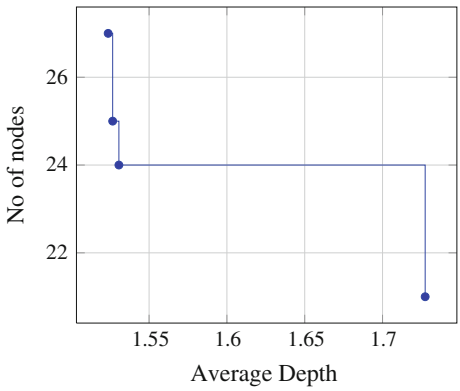


Fig. 8 MUSHROOM dataset (22 attributes and 8125 rows)

plot in Fig. 7. It shows that when the number of nodes is 250 the average depth of the tree is 4.36. These plots help us understand the nature of trees for the particular dataset. The plots in Figs. 7 and 8 use average depth instead of total path length.

6 Conclusion

This paper presents a tool for studying the relationships between the number of nodes and average depth (total path length) for decision trees. Further studies will be connected with the extensions of this tool including studying relationships between depth and average depth of decision trees.

A.1 7 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 $B_f = \{m_f, m_f + 1, \ldots, M_f\}$ and $B_g = \{n_g, n_g + 1, \ldots, N_g\}$ 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 follows:

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

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

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. In this case, there exits m_1 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 $\big(\mathcal{G}(m_f), \mathcal{G}(m_f + 1), \ldots, \mathcal{G}(M_f)\big)$ 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.

References

1. Frank A, Asuncion A (2010) UCI Machine Learning Repository
2. Alkhalid A, Chikalov I, Moshkov M (2010) On algorithm for building of optimal α-decision trees. In: Szczuka MS, Kryszkiewicz M, Ramanna S, Jensen R, Hu Q (eds) RSCTC. Springer, Heidelberg, pp 438–445
3. Alkhalid A, Chikalov I, Moshkov M (2010) A tool for study of optimal decision trees. In: Yu J, Greco S, Lingras P, Wang G, Skowron A (eds) RSKT. LNCS, vol 6401. Springer, Heidelberg, pp 353–360
4. Alkhalid A, Chikalov I, Hussain S, Moshkov M (2012) In: Extensions of dynamic programming as a new tool for decision tree optimization. SIST, vol 13. Springer, Heidelberg, pp 16–36
5. 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: Francisco V. C. Ficarra (ed) Computational informatics, social factors and new information technologies: hypermedia perspectives and avant-garde experiencies in the Era of communicability expansion. Blue Herons, Bergamo, pp 29–39
6. Chikalov I, Hussain S, Moshkov M (2011) Relationships between depth and number of missclassifications for decision trees. In: Kuznetsov SO, Slezak D, Hepting DH, Mirkin B (eds) Thirteenth international conference on rough sets, fuzzy sets, data mining and granualr computing (RSFDGrC 2011). LNCS, vol 6743. Springer, Heidelberg, pp 286–292
7. Pawlak Z (1991) Theoretical aspects of reasoning about data. Kluwer Academic Publishers, Dordrecht
8. Skowron A, Rauszer C (1992) 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. Kluwer Academic Publishers, Dordrecht, pp 331–362
9. Nguyen HS (1998) From optimal hyperplanes to optimal decision trees. Fundam Inf 34(1–2): 145–174