Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 35 (2014) 514 – 521

Procedia
Computer Science

18th International Conference on Knowledge-Based and Intelligent
Information & Engineering Systems - KES2014

Total path length and number of terminal nodes for decision trees

Shahid Hussain

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

Abstract

This paper presents a new tool for study of relationships between total path length (average depth) and number of terminal nodes for decision trees. These relationships are important from the point of view of optimization of decision trees. In this particular case of total path length and number of terminal nodes, the relationships between these two cost functions are closely related with space-time trade-off. In addition to algorithm to compute the relationships, the paper also presents results of experiments with datasets from UCI ML Repository[1]. These experiments show how two cost functions behave for a given decision table and the resulting plots show the Pareto frontier or Pareto set of optimal points. Furthermore, in some cases this Pareto frontier is a singleton showing the total optimality of decision trees for the given decision table.

Keywords: Decision trees, number of terminal nodes, total path length, average depth, Pareto frontier, Pareto optimal.

1. Introduction

A *decision tree* is a finite directed tree with the root in which terminal nodes are labeled with *decisions*, nonterminal nodes with *attributes*, and edges are labeled with *values of attributes*. Decision trees are widely used as *predictors*[2], as a way of *representing knowledge*[3], and as *algorithms* for problem solving[4]. Each such use has a different optimization objective. That is, we need to minimize the number of misclassifications in order to achieve more accurate decision trees (from the perspective of *prediction*). To have more understandable decision trees we need to minimize the number of nodes in a decision tree (*knowledge representation*). Decision trees, when used as *algorithms*, need to be shallow i.e., we need to minimize either the depth or average depth (or in some cases both) of a decision tree in order to reduce algorithm complexity. Unfortunately, almost all problems connected with decision trees optimization are NP-hard[4,5].

Several exact algorithms for decision tree optimization are known including brute-force algorithms[6], algorithms based on dynamic programming[7,8,9], and algorithms using branch-and-bound technique[10]. Similarly, different algorithms and techniques for construction and optimization of approximate decision trees have been extensively studied by researchers in the field, for example, using genetic algorithms[11], simulated annealing[12], and ant colony[13]. Most

E-mail address: shahid.hussain@kaust.edu.sa

doi:10.1016/j.procs.2014.08.132

approximation algorithms for decision trees are greedy, in nature. Generally, these algorithms employ a top-down approach and at each step minimize some impurity. Several different impurity criteria are known in literature, for example information-theoretic[14], statistical[3], and combinatorial[15,4]. See[16,17,18,19,20,21,22,14] for comparison of different impurity criteria.

We have created a software system for decision trees (as well as for 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[23,24,25].

Decision tree optimization and sequential optimization naturally lead to questions such as what is the *relationship* between two *cost functions* for construction of decision trees, or how *uncertainty* effect the overall structure of trees (i.e., how *number of nodes* or average depth of decision trees is associated with *entropy*, or *misclassification error*?) These questions are very important from the practical point of view, related to building optimal yet cost effective decision trees. To this end, we have created algorithms to answer such questions.

In this paper, we consider relationships between two important cost functions closely related with time and space complexity that describe the trade-off, i.e., total path length/average depth (time complexity) and number of terminal nodes (space complexity) of a decision tree. We also give details about experimental results for several datasets acquired from UCI ML Repository[1] as well as demonstrate working of algorithm on a simple example. The result of algorithm for computing the relationship between total path length and number of terminal nodes is stored in a vector of points in the plane. These points form *Pareto frontier* or *Pareto set* i.e., the set of Pareto optimal points. For example, the point $(11, 154)$ in Fig. 8 tells us that the best decision tree with at least 11 terminal nodes will have 154 as total path length. In cases, where the set of points is singleton the corresponding decision trees are called *totally optimal* with respect to the two considered cost functions, see for example plot in Fig 6.

Relationships between different cost functions as well as between cost function and uncertainty measure for decision trees have been studied extensively[26,27,28,29]. The presented algorithms and their implementation in the software tool Dagger together with similar algorithms devised by the author (see for example[30]) can be useful for investigations, in particular, in Rough Sets[31,32] where decision trees are used as classifiers[33].

This paper is divided into six sections including the introduction. Section 2 defines basic notions related with decision tables and trees. Section 3 presents the algorithm to construct sets of decision trees. The main algorithm for computing the relationships is considered in Section 4. Section 5 shows experimental results and Section 6 concludes the paper followed by references.

2. Decision tables and 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 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.

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

Fig. 1. Decision table

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. 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 $\Lambda(T, \Gamma)$, 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 terminal nodes* for decision tree Γ for the table T are denoted as $\tau(\Gamma) = \tau(T, \Gamma)$. It is clear for a given decision table T with m attributes and N rows, the upper bound on total path length of a tree is mN and upper bound on number of terminal nodes is N.

3. Sets of decision trees

We 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 DT

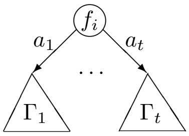

Fig. 3. Aggregated DT

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 $f_i \in E(\Theta)$ 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 ([34]). *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 terminal 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 Section 2 and Section 3).

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

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

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

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 $(f_i, a_1), \ldots, (f_i, a_t)$ 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 $B_{\tau,\Theta(f_i,a_1)}, \ldots, B_{\tau,\Theta(f_i,a_t)}$, respectively. Let

$$B_{\tau,\Theta,f_i} = \{\alpha_i, \alpha_i + 1, \ldots, N\}, \quad \text{where } \alpha_i = \sum_{j=1}^{t} v_j.$$

One can show that α_i is the minimum number of terminal 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 $B_{\tau,\Theta}$.

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

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

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

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

We can use the following proposition to construct the function $\mathcal{G}_T$ from the function $\mathcal{F}_T$.

Proposition 2 ([28]). *For any $n \in B_{\Lambda,T}$, $\mathcal{G}_T(n) = \min\{p \in B_{\tau,T} : \mathcal{F}_T(p) \leq n\}$.*

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

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, $\left((\gamma_j, \lambda_{\gamma_j}^{j}), (\gamma_j + 1, \lambda_{\gamma_j+1}^{j}), \ldots, (N, \lambda_N^{j})\right)$, where $\gamma_j = \alpha(\Theta(f_i, a_j))$ and $\lambda_j^k = \mathcal{F}_{\Theta(f_i,a_j)}(k)$. We need to compute $\mathcal{F}_{\Theta}^{f_i}(n)$ for all $n \in B_{\tau,\Theta,f_i}$;

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

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

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 $(l_0, l_1, \ldots, l_t)$. Each node has a pair of labels and each layer $l_j (1 \leq j \leq t)$ contains at most jN nodes. The first entry of labels for nodes in a layer l_j is an integer from $\{1, 2, \ldots, jN\}$. The layer l_0 contains only one node labeled with $(0, 0)$.

Each node in a layer l_j $(0 \leq j < t)$ has at most N 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 $x + \gamma_{j+1}$ to $x + N$ (as a first entry in their label-pairs) in layer l_{j+1}, with edges labeled as $(\gamma_{j+1}, \lambda_{\gamma_{j+1}}^{j+1}), (\gamma_{j+1} + 1, \lambda_{\gamma_{j+1}+1}^{j+1}), \ldots, (N, \lambda_N^{j+1})$, respectively. It is important to note here that each layer l_j, $0 \leq j \leq t$ has at most one node that has the value j as first label in its label-pair.

The function $\mathcal{F}_{\Theta}^{f_i}(n)$ for $n \in B_\tau$ 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 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 predecessor nodes of (k, λ) such that $(\alpha_1,\beta_1), (\alpha_2,\beta_2), \ldots, (\alpha_r,\beta_r)$ are the labels of 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 $\mathcal{F}_{\Theta}^{f_i}(n)$. 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

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

An example of working of the algorithm can be found in Fig. 4. In this figure the pair of values in each box represent the function $\mathcal{F}$ for that particular subtable. According to the algorithm, the number of pairs of values should

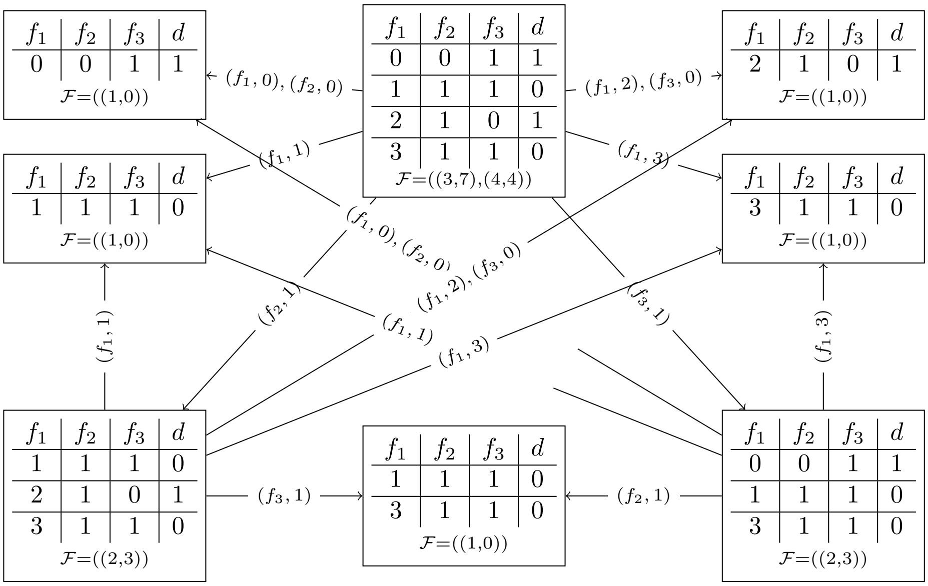

Fig. 4. Example illustrating the working of the algorithm

be equal to the number of rows in that particular subtable however, in this figure, and in subsequent experimental results, the remaining unfeasible pairs of values are discarded as they do not correspond to any decision tree for the specific subtable.

It is clear that the considered algorithm has polynomial time complexity depending on $N(T)$ and t (with $t \leq N(T)$), here $N(T)$ is the number of rows in decision table T and t is number of possible values for some attribute f_i in T.

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 interesting to note that plot for the dataset CARS has only one point. This shows that there is a decision tree for this dataset which is simultaneously optimal for number of terminal nodes and total path length.

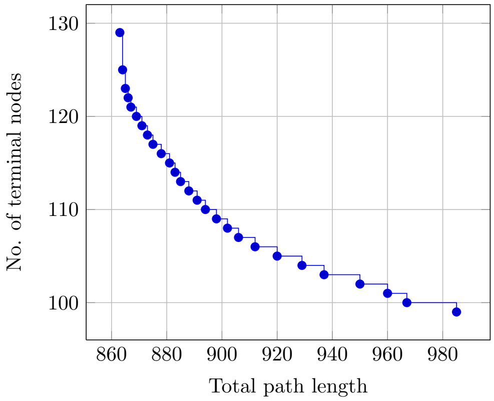

Fig. 5. BREAST-CANCER dataset (10 attributes and 267 rows)

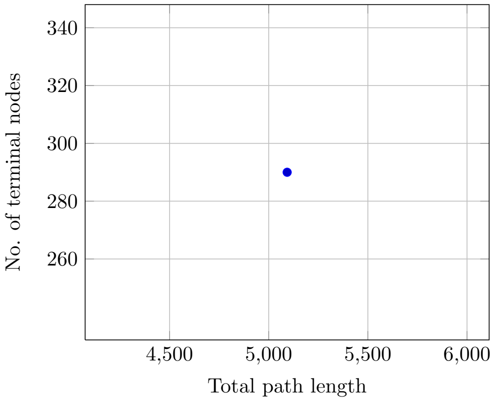

Fig. 6. CARS dataset (6 attributes and 1729 rows)

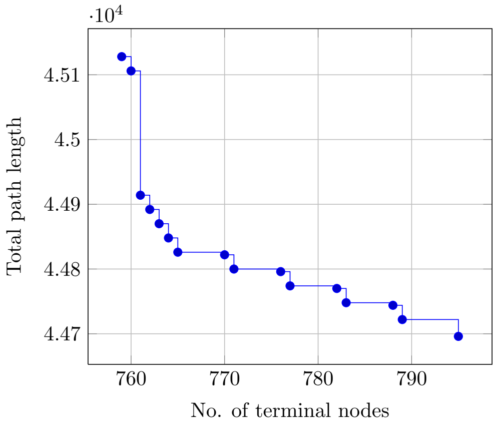

Fig. 7. NURSERY dataset (8 attributes and 12964 rows)

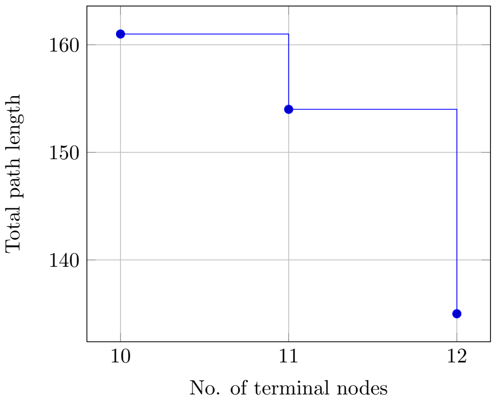

Fig. 8. zoo dataset (16 attributes and 59 rows)

6. Conclusion

This paper is devoted to the consideration of algorithm for computing the relationships between a total path length (average depth) and number of terminal nodes of decision trees. The paper presents, in details, the algorithm together with a simple example to demonstrate how the algorithm works and some experimental results using standard datasets from UCI. This algorithm along with other similar algorithms have been implemented in a software system called Dagger [23,24]. Further studies in this direction will be devoted to consideration of relationships between space and time complexity of decision trees corresponding Boolean functions.

Acknowledgment

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

References

1. Frank, A., Asuncion, A.. UCI Machine Learning Repository. 2010. URL: `http://archive.ics.uci.edu/ml`.
2. Hastie, T., Tibshirani, R., Friedman, J.H.. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag; 2001.
3. Breiman, L., Friedman, J., Olshen, R., Stone, C.. *Classification and Regression Trees*. Monterey, CA: Wadsworth and Brooks; 1984.
4. Moshkov, M.J.. Time complexity of decision trees. In: Peters, J.F., Skowron, A., editors. *T. Rough Sets*; vol. 3400 of *Lecture Notes in Computer Science*. Heidelberg: Springer; 2005, p. 244–459.
5. Hyafil, L., Rivest, R.L.. Constructing optimal binary decision trees is NP-complete. *Inf Process Lett* 1976;**5**(1):15–17.
6. Riddle, P., Segal, R., Etzioni, O.. Representation design and brute-force induction in a Boeing manufacturing domain. *Applied Artificial Intelligence* 1994;**8**:125–147.
7. Garey, M.R.. Optimal binary identification procedures. *SIAM Journal on Applied Mathematics* 1972;**23**:173–186.
8. Martelli, A., Montanari, U.. Optimizing decision trees through heuristically guided search. *Commun ACM* 1978;**21**(12):1025–1039.
9. Schumacher, H., Sevcik, K.C.. The synthetic approach to decision table conversion. *Commun ACM* 1976;**19**(6):343–351.
10. Breitbart, Y., Reiter, A.. A branch-and-bound algorithm to obtain an optimal evaluation tree for monotonic boolean functions. *Acta Inf* 1975;**4**:311–319.
11. Chai, B.B.. Binary linear decision tree with genetic algorithm. In: *Proceedings of the International Conference on Pattern Recognition (ICPR '96)*; vol. 7472 of *ICPR '96*. Washington, DC, USA: IEEE Computer Society; 1996, p. 530–534.
12. Heath, D.G., Kasif, S., Salzberg, S.. Induction of oblique decision trees. In: Bajcsy, R., editor. *IJCAI*. Morgan Kaufmann; 1993, p. 1002–1007.
13. Boryczka, U., Kozak, J.. New algorithms for generation decision trees—ant-miner and its modifications. In: Abraham, A., Hassanien, A.E., de Leon Ferreira de Carvalho, A.C.P., Snásel, V., editors. *Foundations of Computational Intelligence*. 2009, p. 229–262.
14. Quinlan, J.R.. Induction of decision trees. *Machine Learning* 1986;**1**(1):81–106.

15. Moret, B.M.E., Thomason, M.G., Gonzalez, R.C.. The activity of a variable and its relation to decision trees. *ACM Trans Program Lang Syst* 1980;**2**(4):580–595.
16. Alkhalid, A., Chikalov, I., Moshkov, M.. Comparison of greedy algorithms for α-decision tree construction. In: Yao, J., Ramanna, S., Wang, G., Suraj, Z., editors. *RSKT*. 2011, p. 178–186.
17. Alkhalid, A., Chikalov, I., Moshkov, M.. Comparison of greedy algorithms for decision tree construction. In: Filipe, J., Fred, A.L.N., editors. *KDIR*. 2011, p. 438–443.
18. Alkhalid, A., Chikalov, I., Moshkov, M.. Decision tree construction using greedy algorithms and dynamic programming – comparative study. In: *20th International Workshop on Concurrency, Specification and Programming*. Pultusk, Poland; 2011, p. 1–9.
19. Fayyad, U.M., Irani, K.B.. The attribute selection problem in decision tree generation. In: *AAAI*. 1992, p. 104–110.
20. Kononenko, I.. On biases in estimating multi-valued attributes. In: *IJCAI*. 1995, p. 1034–1040.
21. Martin, J.K.. An exact probability metric for decision tree splitting and stopping. *Mach Learn* 1997;**28**(2–3):257–291.
22. Mingers, J.. Expert systems – rule induction with statistical data. *Journal of the Operational Research Society* 1987;**38**:39–47.
23. 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: Ficarra, F.V.C., Kratky, A., Veltman, K.H., Ficarra, M.C., Nicol, E., Brie, M., editors. *Computational Informatics, Social Factors and New Information Technologies: Hypermedia Perspectives and Avant-Garde Experiencies in the Era of Communicability Expansion*. Blue Herons; 2011, p. 29–39.
24. Alkhalid, A., Chikalov, I., Hussain, S., Moshkov, M.. Extensions of dynamic programming as a new tool for decision tree optimization. In: Ramanna, S., Jain, L.C., Howlett, R.J., editors. *Emerging Paradigms in Machine Learning*; vol. 13 of *Smart Innovation, Systems and Technologies*. Springer Berlin Heidelberg; 2013, p. 11–29.
25. 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., editors. *RSCTC*. Heidelberg: Springer; 2010, p. 438–445.
26. Chikalov, I., Hussain, S., Moshkov, M.. On cost and uncertainty of decision trees. In: Yao, J., Yang, Y., Slowinski, R., Greco, S., Li, H., Mitra, S., et al., editors. *Rough Sets and Current Trends in Computing - 8th International Conference, RSCTC*. 2012, p. 190–197.
27. Chikalov, I., Hussain, S., Moshkov, M.. Relationships between average depth and number of nodes for decision trees. In: Sun, F., Li, T., Li, H., editors. *Knowledge Engineering and Management*; vol. 214 of *Advances in Intelligent Systems and Computing*. Springer Berlin Heidelberg. ISBN 978-3-642-37831-7; 2014, p. 519–529.
28. Hussain, S.. Relationships among various parameters for decision tree optimization. In: Faucher, C., Jain, L.C., editors. *Innovations in Intelligent Machines-4 - Recent Advances in Knowledge Engineering*; vol. 514 of *Studies in Computational Intelligence*. Springer; 2014, p. 393–410.
29. Chikalov, I., Hussain, S., Moshkov, M.. Relationships between average depth and number of misclassifications for decision trees. *Fundamenta Informaticae* 2014;**129**(1-2):15–26.
30. Chikalov, I., Hussain, S., Moshkov, M.. Relationships between depth and number of misclassifications for decision trees. In: Kuznetsov, S.O., Ślęzak, D., Hepting, D.H., Mirkin, B., editors. *Rough Sets, Fuzzy Sets, Data Mining and Granular Computing*; vol. 6743 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg. ISBN 978-3-642-21880-4; 2011, p. 286–292.
31. Pawlak, Z.. *Theoretical Aspects of Reasoning about Data*. Dordrecht: Kluwer Academic Publishers; 1991.
32. Skowron, A., Rauszer, C.. The discernibility matrices and functions in information systems. In: Slowinski, R., editor. *Intelligent Decision Support. Handbook of Applications and Advances of the Rough Set Theory*. Dordrecht: Kluwer Academic Publishers; 1992, p. 331–362.
33. Nguyen, H.S.. From optimal hyperplanes to optimal decision trees. *Fundamenta Informaticae* 1998;**34**(1-2):145–174.
34. Chikalov, I., Hussain, S., Moshkov, M.. Relationships for cost and uncertainty of decision trees. In: Skowron, A., Suraj, Z., editors. *Rough Sets and Intelligent Systems - Professor ZdzisÅaw Pawlak in Memoriam*; vol. 43 of *Intelligent Systems Reference Library*. Springer Berlin Heidelberg. ISBN 978-3-642-30340-1; 2013, p. 203–222.