A Novel Definition of an Efficient Binary Decision Tree

Christopher Nwosisi
Computer Science Department, The College of Westchester and Pace University, White Plains, NY, USA
Email: cnwosisi@cw.edu

***Abstract*—Decision *trees* have been well studied, widely used in knowledge discovery and decision support systems. They are simple and practical prediction models but often suffer from excessive complexity and can even be incomprehensible. In this study, a *genetic algorithm* is used to construct decision trees of increased accuracy and efficiency compared to those constructed by the conventional ID3 or C4.5 decision tree building algorithms. An improved definition of an efficient binary decision tree is proposed and evaluated – instead of simply using the number of nodes in a tree, the average number of questions asked in the tree for all the database entries is proposed.**

***Index Terms*—binary decision trees, genetic algorithm and DVT**

I. Introduction

Decision trees approximate discrete-valued target functions as trees and are widely used practical methods for inductive inference in knowledge discovery and decision support systems because of their natural and intuitive paradigm to classify a pattern through a sequence of questions [1]. Algorithms for constructing decision trees such as ID3 [1]-[3] and C4.5 [4] often use heuristics to find a shorter tree. However, finding efficient and accurate decision trees is a difficult optimization problem [5]-[7].

Genetic algorithms (GAs) use an optimization technique based on natural evolution [8]-[11]. GAs has been used to find near-optimal decision trees in twofold. On the one hand, they were used to select attributes to be used to construct decision trees in a hybrid or preprocessing manner [12]-[14]. On the other hand, they were applied directly to decision trees [15], [16]. A problem that arises with this approach is that an attribute may appear more than once in the path of the tree.

In order to utilize genetic algorithms, decision trees must be represented as chromosomes where genetic operators such as mutation and crossover can be applied. The main contribution of this paper is proposing a new an improved definition of an efficient binary decision tree, instead of simply using the number of nodes in a tree, the average number of questions asked in the tree for all the database entries is proposed. The remainder of the paper is organized as follows. Section 2 reviews decision trees. Section 3 describes the dataset. Section 4 presents the encoding scheme and the results and section 5 discusses the conclusion.

II. Preliminary: Decision Trees

Decision trees are very popular data mining method for classification and regression, and can be conveniently induced, exchanged, and visualized by many tools. A decision tree consists of intermediate nodes, where attributes (variables) are tested, and leaves where decisions are stored [17], [18] (see Fig. 1).

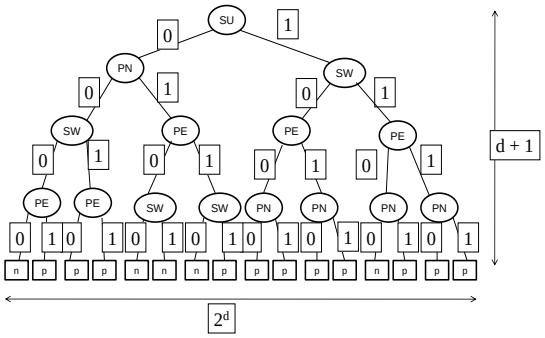


Figure 1. A sample of a full binary decision tree structure T1f

A decision tree is a hierarchal structure (a flowchart), where each internal node (no leaf node) signifies a test on an attribute. The branches represent the outcomes of the tests, and the leaf nodes (or terminal nodes) hold the class labels. The root node is the topmost node in a tree [17] (see Fig. 2).

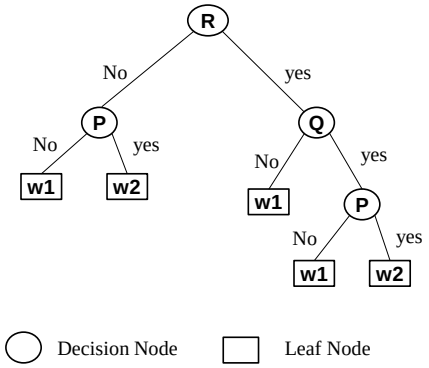


Figure 2. Representation of a generic decision tree

The formation of decision tree classifiers does not require any parameter setting that makes it appropriate for exploratory knowledge discovery. Decision trees can

Manuscript received March 8, 2015; revised July 10, 2015.

doi: 10.18178/lnit.3.1.35-41

manage high dimensional data. Their tree representation form is intuitive and generally easy to understand by humans.

The learning and classification steps of decision tree induction are simple and fast. Overall, decision tree classifiers seem to have good accuracy, which may also depend on the data that is provided. In medicine, manufacturing and production, financial analysis, astronomy, and molecular biology, decision tree induction algorithms have been used for classification [5].

	P	Q	R	S	*w*
x1	0	0	0	0	*w1*
x2	0	1	1	0	*w1*
x3	1	0	1	0	*w1*
x4	0	0	1	1	*w2*
x5	1	1	0	0	*w2*
x6	0	0	1	0	*w2*

(SU = Surgery, SW = Swelling, PN = leg pain and PE = Pulmonary embolism) [5]

Given a set of training data set D where each attribute have a value. D is a matrix with n instances where each instance x_i has a value which is one of c states of nature w. The database sample consists of $n = 6$, $d = 4$, $c = 2$ and $w = \{w1, w2\}$.

A decision tree is a rooted tree *T* that consists of internal nodes representing attributes, leaf nodes representing labels, and edges representing the attributes' possible values.

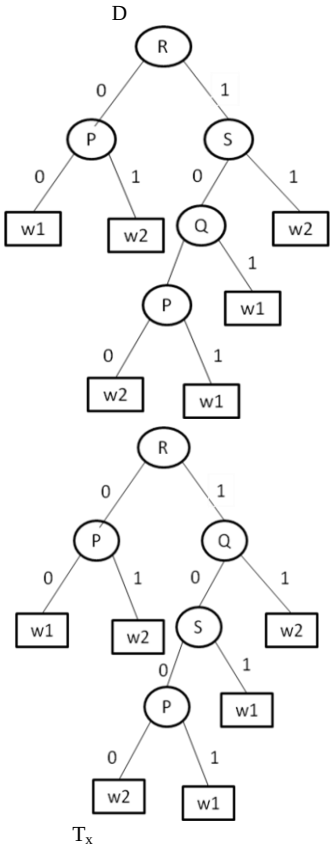


Figure 3. T_x is consistent with D and T_y is inconsistent with D [17]

Decision trees classify instances by traversing from root node to leaf node. The classification process starts from the root node of a decision tree, tests the attribute specified at this node, and moves down the tree branch according to the given attribute value. Fig. 3 shows two decision trees, T_x and T_y. T_x is said to be a consistent decision tree because it is consistent with all instances in D. T_y is inconsistent with D because x_2's class is actually w1 in D whereas T classifies it as w2.

There are two important properties of a binary decision tree:

- Property is *the size of a decision tree with l leaves is* $2l -$
- Property is *the lower and upper bounds of l for a consistent binary decision tree are c and n:* $c <= l <= n$.

The number of leaves in a consistent tree must be at least *c* in the best cases; If D represents a c-class classification problem. The number of leaves will be the size of D with each instance corresponding to a unique leaf, in the worst cases, for example, T_1 and T_2 [5], [19].

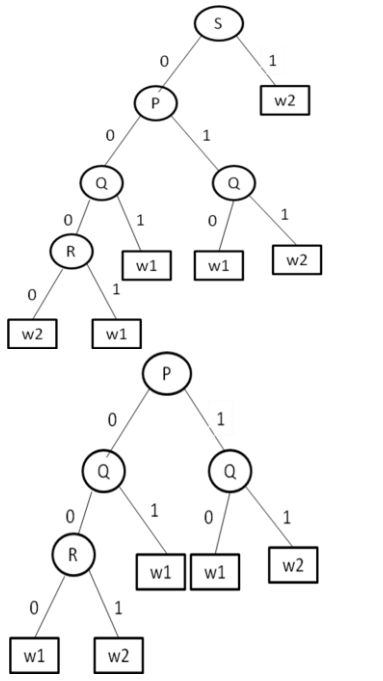


Figure 4. Consistent decision trees

Fig. 4 shows two consistent decision trees. All instances $x = \{x_1, \ldots, x_6\}$ are classified correctly by both decision trees T_1 and T_2. Conversely, an unknown instance (0, 0, 0, 1, ?), which is not in the training set, D is classified differently by the two decision trees. T_1 classifies the instance as w_2 whereas T_2 classifies it as w_1. This inductive inference is a fundamental problem in machine learning [17], [19]. In this case, the simpler decision tree is preferred, a strategy that agrees with a well- known principle known as Occam's razor [5], [11], [20]. The *minimum description length principle* formalized from Occam's razor [5], [21], [22] is a very important concept in machine theory [5], [21], [23]. Occam's razor is intuitive

because the additional components in a complex decision tree stand a greater chance of being fitted purely by chance. In the words of Einstein, "Everything should be made as simple as possible, but not simpler" [17], [20].

Two data sets were extracted from the databases in the Montefiore Medical Center Vascular Laboratory and the general patient registry. Then, selected attributes were converted into binary attributes, and shorter and/or more accurate decision trees were created using the genetic algorithm on both of the DVT datasets.

III. DVT Datasets

Deep Venous Thrombosis (*DVT)* is an intrinsic disease where blood clots form in a deep vein in the body. Known risk factors for *DVT* include diabetes, surgery, smoking, cancer, obesity, congestive heart failure, swelling, cellulitis, injury, and pulmonary embolism [5]. These factors can be determined by patients and physicians without medical examinations. Hence, eighteen potential attributes which can contribute to DVT were extracted from 515 records in databases at the Montefiore Medical Center Vascular Laboratory and the general patient registry. The dataset attributes are summarized in Table I together with the DVT outcome. Of the 515 records 350 patients were positive and 165 negative for DVT.

TABLE I. Dataset Attributes

	Name	Description
1	Sex (GN)	0 = female; 1 = male
2	Age (A6)	0 = age < 60; 1 = age ≥ 60
3	Diabetes (DB)	0 = normal; 1 = receiving some treatments
4	Smoking (SM) (SS, SB)	0 = never smoked; 1 = active Smoker; 2 = stopped smoking
5	Surgery (SR)	0 = never had surgery; 1 = previous surgery
6	Pain (PN) (LP, RP)	0 = none; 1 = pain in the leg {None, Right, Left, Bilateral}
7	Swelling (SW)	0 = none; 1 = swelling in the leg
8	Chest Pain (CP)	0 = none; 1 = pain in Chest
9	Cancer (CR)	0 = normal; 1 = positive
10	Cellulitis (CL)	0 = normal; 1 = positive
11	Injury (IJ)	0 =none; 1 = previous injuries
12	Pulmonary embolism (PE)	0 = never diagnosed; 1 = previously diagnosed
13	Congestive heart failure (HF)	0 = never diagnosed; 1 = previously diagnosed
14	Obesity (OB)	0 = none; 1 = specified
15	Accident (AC)	0 = none; 1 = had a fall
16	Hyperlipidemia (LIP)	0 = never diagnosed; 1 = previously diagnosed
17	Cardiac Dysrth-ythmia (CD)	0 = normal; 1 = previously diagnosed
18	Lymphoproliferat disease (LD)	0 = normal; 1 = previously diagnosed
	DVT	0 = negative for DVT; 1 = positive for DVT

To use the genetic algorithm to build a binary decision tree, the attribute types must be binary [5], [24]. The numeric data, 'age' attribute (A6) is binarized: 1 if over 60 and 0 otherwise. Non-binary nominal attributes include 'smoking' and 'pain' where they have three and four possible values, respectively. These are binarized as shown in Table II.

TABLE II. Nominal to Binary Preprocessing

Smoking		Leg Pain	
SB SS		LP RP	
1 1	Smoking	1 1	Bi
1 0	Stopped	1 0	L
0 0	Never	0 1	R
		0 0	None

The nominal type 'Leg Pain' attribute which has four possible values {L, R, Bi, N} in the original table is represented by two binary attributes, LP (pain in the left leg) and RP (pain in the right leg). The ternary attribute, 'Smoking' in the original table is represented by two binary attributes 'SB' (smoked before) and 'SS' (still smoking). Note that in certain datasets, the smoking attribute is denoted as simply 'SM' having either 0 (nonsmoker) or 1 (smoker). This is because not all questionnaires distinguish the stopped smoker. Similarly, the pain attribute may appear as simply 'PN' in some datasets.

Potential users for the proposed prediction models include patients at home and physicians. Two datasets were created – one for patients and one for physicians and those with medical knowledge. Because most patients have little medical knowledge, Dataset I (see Table I, Nos. 1-7) was created with attributes which can be determined easily without much medical knowledge. Dataset II (see Table I, Nos. 1-18) was created using all the attributes in Table I (except for PN) and this dataset is for physicians or users with some medical knowledge.

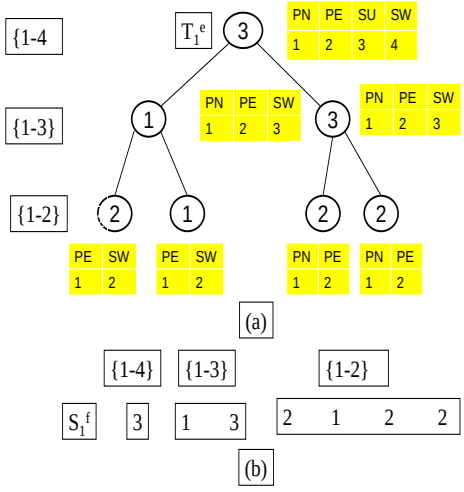


Figure 5. Illustration of encoding schema [17].

IV. Algorithm and Results

To construct decision trees using genetic algorithms, the tree must be encoded to enable the genetic operators, such as mutation and crossover to be applied. Let P = {p_1, p_2…

P_d} be the ordered attribute list. To show and describe the process, the full binary decision tree T_1^f in Fig. 1, P = {PN, PE, SU, SW}. The encoding process converts the attribute names in the full binary decision tree into index of the attribute as per the ordered attribute list P. It recursively, starts from the root as depicted in Fig. 5. For example, the root is R and its index in P is 3. For each sub-tree, the encoded decision tree T_e updates from 1 to $d - i + 1$ recursively. Final step, takes the breadth-first traversal to generate the chromosome string S. For T_1^f the chromosome string S_1 is given in Fig. 5 (b) [5].

In Fig. 6 are binary decision trees which are built from Dataset I. For each node the left branch is 0 (no) and the right branch is 1 (yes). Tree leaves indicate whether DVT is considered positive or negative.

The decision tree in Fig 6 (b) suggests that a patient might have DVT if he/she never had surgery but has diabetes and is over 60 year old or might have DVT if he/she had previous surgery and feels pain in the leg and had previously smoked. The positive DVT cases can be logically expressed in the disjunctions of conjunctions form: (SR = 0 ∧ DB = 1 ∧ A6 = 1) ∨ (SR=1 ∧ PN=1 ∧ SB = 1).

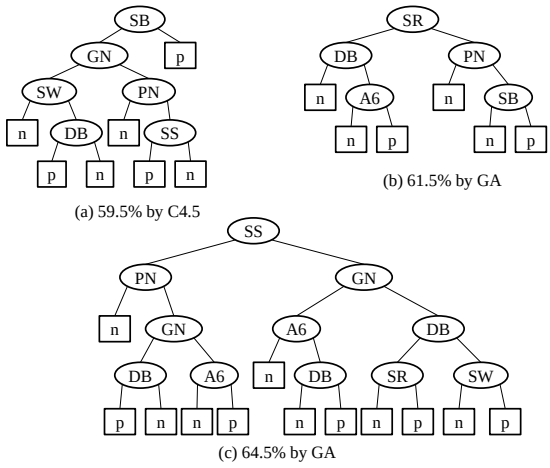


Figure 6. Decision trees from dataset I [11]

If a patient wants to predict the likelihood of DVT, the decision tree prediction model such as one in Fig. 6 (c) will prompt a sequence of questions. First, it will ask whether the patient is a current active smoker. When the patient answers with 'yes', it will prompt to ask about the gender. If the patient is a female, it will prompt whether she is over 60 year old. If the answer is "yes", it will ask whether she is a diabetic. If so, the decision tree predicts that she has a significant risk for DVT; in fact according to current laboratory records, one has a 66.67% chance of having a DVT under these conditions. Also, note that even though the decision tree predicts "No" in the left-most branch in Fig. 6 (c) where the patient is not currently smoking and does not feel pain, the chances that the patient may have DVT according to the database is about 45.6%. The decision tree is capable of providing the probabilities.

The popular decision tree algorithm C4.5 constructs pruned decision trees [6]; and was used to construct the tree shown in Fig. 6 (a) having a performance of 59.5%. The most basic and popular algorithm to construct decision trees, called *ID3*, constructs short trees [8]. However, the decision tree constructed by ID3 is not shown here because it was unreasonably large and too complex for patients and perhaps even physicians to use. However, its performance on Dataset I was 72% for DVT prediction.

In this study, a genetic algorithm is used to find shorter and/or more accurate decision trees. It starts with 100 random decision trees, and only short and good decision trees survive to the next generation. Using mutation and cross-over operations, the next 100 generations are generated. Mutation and crossover are the two most common genetic operators. The mutation operator is defined as changing the value of a certain position in a string to one of the likely values in the range. Fig. 7 illustrate the mutation process on the attribute selection scheduling string S_1^f = (3, 1, 3, 2, 1, 2, 2) and with P= (PN, PE, SU, SW). If a mutation occurs in the first position and changes the value to 4, which is in the range {1... 4}, T_4^f is generated. If a mutation happens in the third position and changes the value to 2, which is in the range {1… 3}, then T_5^f is generated. As long as the changed value is with the allowed range, the new string result will always generate a valid full binary decision tree.

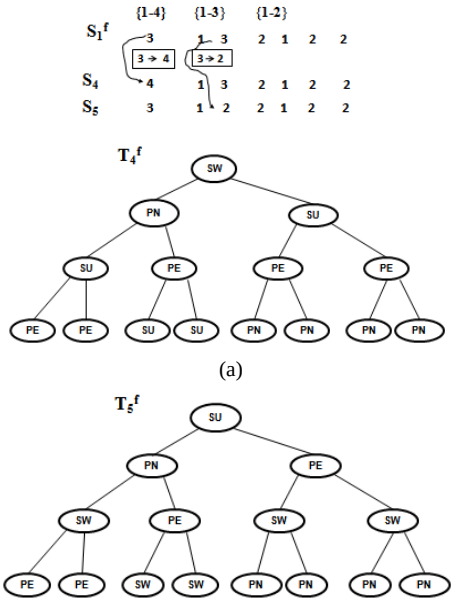
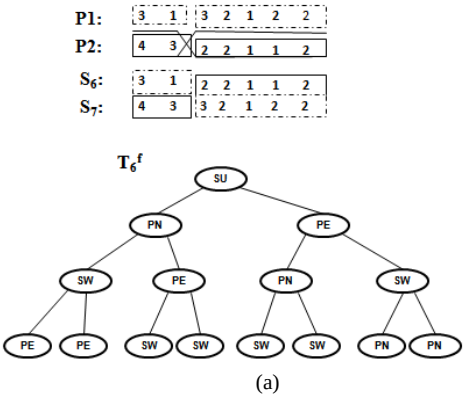
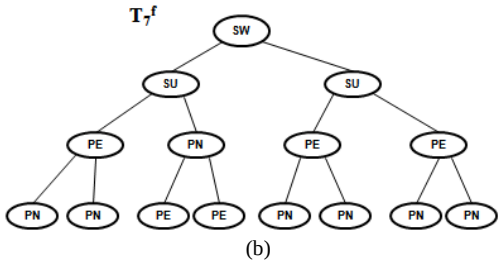


Figure 7. Illustration of mutation operator [11]





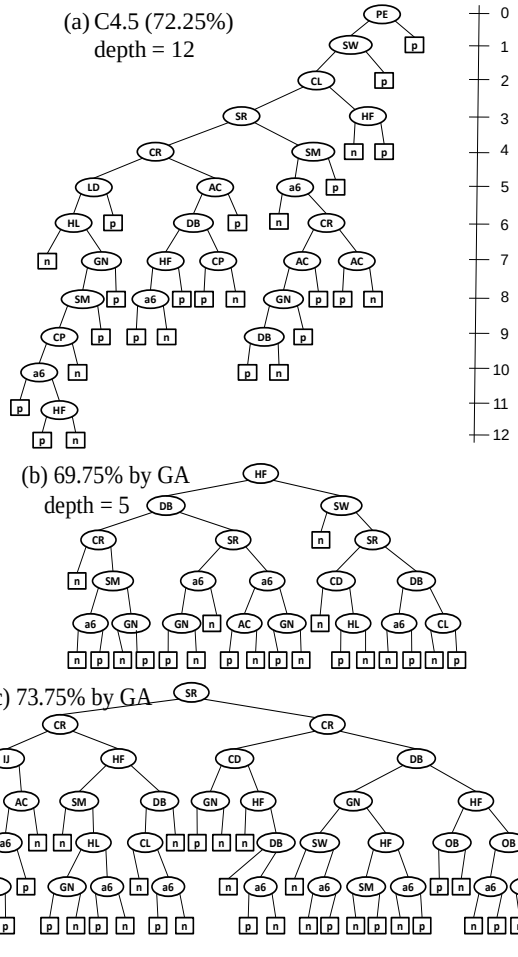
(b)

Figure 8. Illustration of crossover operator [11]

Fig. 8 shows the crossover process with two parents attribute selection scheduling strings, P1 and P2. After randomly selecting a split point, the first part of P1 and the last part of P2 contribute to yield a child strings S_6. Reversing the crossover produces a second child S_7. T_6^f and T_7^f full decision trees resulted from these two children.

For dataset I, several decision trees which are shorter and more accurate than the one created by ID3 in Fig. 6 (a) were identified. Shorter depth and more accurate decision tree is shown in Fig. 6 (b) and an even more accurate one but of the same depth is shown in Fig. 6 (c).

For dataset II, Fig. 9 shows a decision tree by the C4.5 algorithm, and three decision trees by GA. The C4.5 decision tree is a skewed and deep (depth = 12) with an accuracy of 72.25%. When the tree is deep, strange rules can be found; for example, HF at the bottom of Fig. 9 (a) tree has the negative DVT when HF is positive, a rule which is not statistically valid.



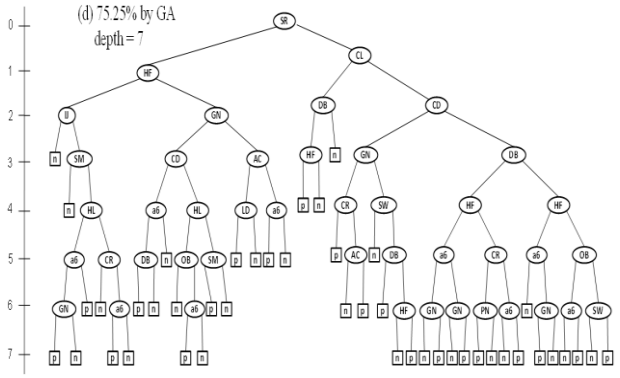


Figure 9. Decision trees from dataset II.

To find shorter and more accurate trees, the GA was performed for 200 generations. By limiting tree depth to 5, the decision tree of Fig. 9 (b) was obtained. Its performance rate, however, is lower than that of C4.5. Fig. 9 (c) and (d) show trees found by limiting the tree depth to 6 and 7, respectively, and have accuracies of 73.75% and 75.25%. It has been observed that greater depth usually results in higher accurate until over-fitting occurs.

The best measure of efficiency (shortness) for a decision tree is probably the average number of questions required to obtain a prediction. Other measures might be the depth of the tree or the number of nodes in the tree.

TABLE III COMPLEXITY OF DECISION TREES WITH DIFFERENT DEPTH LIMITS

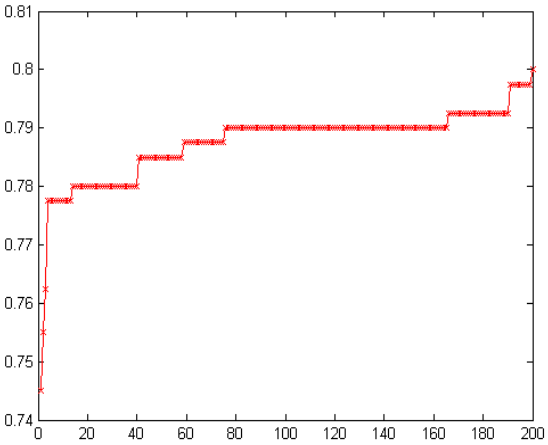
GA		
Depth limit	Performance rate	The average # of question
5	**69.75**	**2.9525**
6	**73.75**	**3.3725**
7	**75.25**	**3.8955**
8	76.50	4.3275
9	76.75	4.8225
10	78.00	5.1225
11	78.50	5.4675
12	79.50	5.8675
13	80.25	6.3075
C4.5		
12	**72.25**	**7.485**
ID3		
16	**80.0**	

Table III shows the depth limits in GA, the performance rate, and the average number of questions to be asked. Note that the average number of questions increases monotonically with the depth limit, indicating that depth also appears to be a good measure of efficiency [5], [17]. The average number of questions to be asked of a user is 7.485 for the C4.5 decision tree in Fig. 9 (a) whereas there are several shorter ones listed in Table 3. The number of nodes is apparently not a good measure of efficiency – the C4.5 decision tree has 25 compared to 19, 32, and 44 in Fig. 9 (b), (c), and (d).

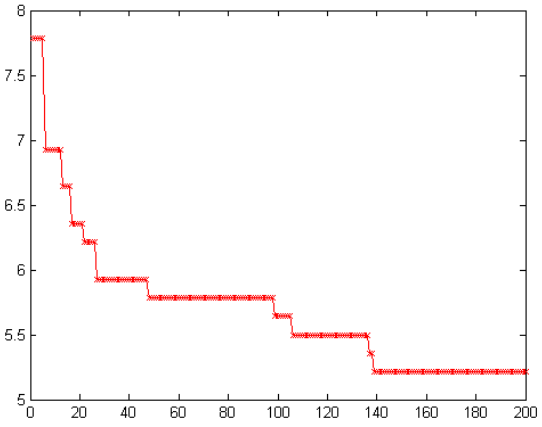
From both a depth and average-number-of-questions perspective, the complexity of the decision tree in Fig. 9 (d) can be considered much more efficient (simpler) than the decision tree from the C4.5 algorithm.

It was observed that accuracy increases as depth increases. At the depth of 12 the GA performance was 79.50 as compared with the C4.5 performance of 72.25 at the same depth. ID3 depth grows until the depth of 16 with a performance rate of 80% versus GA 80.25% with the depth of 13. These results clearly show that trees constructed by GA are both more accurate and more efficient.

Fig. 10 (a) and (b) show the highest performance positive prediction rate and the lowest number of questions needed, respectively, to determine DVT for the entire test set for 200 generations.



(a)



(b)

Figure 10. Prediction rate (a) and number of questions (b) fitness function of GA generations on dataset II [11].

V. Discussion

For the purpose of DVT classification, the genetic algorithm is exploited to find shorter and/or more accurate decision trees than ones produced by the conventional ID3 and C4.5 algorithms. Experimental results on two datasets suggest that more accurate and efficient decision trees can be found by the GA. The efficiency (lower complexity) of a decision tree is best defined by the average number of questions asked to users, not by the number of nodes in the decision tree. In view of this argument, GA trees were found to produce more accurate and more efficient trees than ones produced by conventional methods such as the ID3 and C4.5 algorithms.

The decision trees produced by the GA have significant clinical relevance. The results shown here increase the probability of predicting whether a patient would develop or have had DVT, which provides advancement in the diagnosis of DVT. The more efficient shorter trees add additional support for the GA method.

With more iteration and deepening the depth of the tree, the decision trees produced by the GA depth limit clearly outperform the one produced by the ID3 method. This study introduced a simple decision tree to help lay people, medical technologists, and physicians identify the probability of a patient having DVT that prompts for testing before any complication occurs.

The decision trees found by using GA tend to be almost full binary trees, i.e., the width is large while the depth is short. For future work, the C4.5 pruning mechanism could be applied to decision trees produced by GA to make trees sparse and to further avoid the potential *over-fitting* problem.

Acknowledgment

The author thanks Taylor Reed (PA), Enid Nwosisi, Dr. Cha, Dr. Tappert and Dr. Lipsitz for their support and for their useful contributions.

References

[1] J. R. Quinlan, "Induction of decision trees," *Machine Learning Research*, vol. 1, no. 1, pp. 81-106, 1986.

[2] L. Hyafil and R. L. Rivest, "Constructing optimal binary decision trees is NP-complete," *Information Processing Letters*, vol. 5, no. 1, pp. 15-17, 1976.

[3] H. Bodlaender and H. Zantema, "Finding small equivalent decision trees is hard," *International Journal of Foundations of Computer Science*, vol. 11, no. 2, pp. 343-354, 2011.

[4] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 21, no. 3, pp. 660-674, 1991.

[5] C. Nwosisi, S. H. Cha, Y. J. An, C. Tappert, and E. Lipsitz, "Predicting deep venous thrombosis using binary decision trees," *International Journal of Engineering and Technology*, vol. 3, no. 5, pp. 467-472, 2011

[6] J. Gehrke, V. Ganti, R Ramakrishnan, and W Loh, "BOAT -optimistic decision tree construction," in *Proc. ACMSIGMOD Conference on Management of Data*, 1999, pp. 169-180.

[7] Q. Zhao and M. Shirasaka, "A study on evolutionary design of binary decision trees," in *Proc. Congress on Evolutionary Computation, IEEE*, 1999, vol. 3, pp. 1988-1993.

[8] T. M. Mitchell, *Machine Learning*, McGraw-Hill, 1997.

[9] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed, Wiley Interscience, 2001.

[10] D. L Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[11] M. Mitchell, "An introduction to genetic algorithms," *Massachusetts Institute of Technology*, 1996.

[12] K. M. Kim, J. P. Joong, M. H. Song, C. Kim, and C. Y. Suen, "Binary decision tree using genetic algorithm for recognizing defect patterns of cold mill strip," *LNCS*, vol. 3060, pp. 1611-3349, 2004.

[13] S. P. Teeuwsen, I. Erlich, M. A. El-Sharkawi, and U. Bachmann, "Genetic algorithm and decision tree based oscillatory stability assessment," *IEEE Transactions on Power Systems*, vol. 21, no. 2, pp. 746-753, May 2006.

[14] J. Bala, J. Huang, H. Vafaie, K. DeJong, and H. Wechsler, "Hybrid learning using genetic algorithms and decision tress for pattern classification," in *Proc. 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada, 1995, pp. 719-724.

[15] A. Papagelis and D. Kalles, "GA tree: Genetically evolved decision trees," in *Proc. 12th IEEE International Conference on Tools with Artificial Intelligence*, 2000, pp. 203-206.
[16] Z. Fu, "An innovative ga-based decision tree classifier in large scale data mining," *LNCS*, vol. 1704, pp. 348-353, 1999.
[17] C. Nwosisi, "Developing a genetic algorithm to construct efficient binary decision trees," Doctoral dissertation, Pace University, New York, 2010.
[18] R. V. Kankaria, "A tool for constructing and visualizing trees augmented bayesian networks for survey data," Master's thesis, University of Minnesota, Duluth, 2004.
[19] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.
[20] D. Nikovski and V. Kulev, "Induction of compact decision trees for personalized recommendation," in *Proc. ACM Symposium on Applied Computing*, France, April 2006.
[21] S. H. Cha, "Comprehensive survey on distance/ similarity measures between probability density functions," *International Journal of Mathematical Models and Methods in Applied Sciences*, vol. 1, pp. 300-307, 2007.
[22] Z. Fu, "Innovative GA-based decision tree classifier in large scale data mining," *LNCS*, vol. 1704, pp. 348-353, 1999.
[23] H. W. Ian and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. San Francisco: Morgan Kaufmann, 2005.
[24] S. H. Cha and C. Tappert, "A genetic algorithm for constructing compact binary decision trees," *Journal of Pattern Recognition Research*, vol. 4, no. 1, pp. 1-13, 2009.

Dr. C. Nwosisi obtained his doctorate degree in Computing from Pace University, Master of Science in management of technology from Polytechnic University and BA in computer science from Hunter College of the City University of New York. He has co-authored several papers in scientific journals and International Conferences. Currently, he works for the College of Westchester in White Plains, New York as an Associate Chair and Professor. He had received numerous presidential commendations, faculty recognition awards and the Shining Star award in Teaching from the College of Westchester. In 2010, he received the Upsilon Pi Epsilon Honors award for the Computing and Information Disciplines from Pace University. In 2007, he received the IEEE senior membership award. In 1993 and 1994 respectively, he was the recipient of the Recognition and Appreciation Awards from the Association for System Management. Dr. Nwosisi has over 25 years of professional experience and over 17 years of teaching experience. His current research interests include Machine Learning, Data mining, Teaching Methodology, and Pattern recognition.