IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 11, 2018, accepted December 26, 2018, date of publication January 11, 2019, date of current version February 4, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2892083

A Unified Framework for Decision Tree on Continuous Attributes

JIANJIAN YAN, ZHONGNAN ZHANG, (Member, IEEE), LINGWEI XIE, AND ZHANTU ZHU
Software School, Xiamen University, Xiamen 361005, China

Corresponding author: Zhongnan Zhang (zhongnan_zhang@xmu.edu.cn)

This work was supported by the Science and Technology Guiding Project of Fujian Province, China, under Grant 2016H0035.

ABSTRACT The standard algorithms of decision trees and their derived methods are usually constructed on the basis of the frequency information. However, they still suffer from a dilemma or multichotomous question for continuous attributes when two or more candidate cut points have the same or similar splitting performance with the optimal value, such as the maximal information gain ratio or the minimal Gini index. In this paper, we propose a unified framework model to deal with this question. We then design two algorithms based on Splitting Performance and the number of Expected Segments, called **SPES1** and **SPES2**, which determine the optimal cut point, as follows. First, several candidate cut points are selected based on their splitting performances being the closest to the optimal. Second, we compute the number of expected segments for each candidate cut point. Finally, we combine these two measures by introducing a weighting factor α to determine the optimal one from several candidate cut points. To validate the effectiveness of our methods, we perform them on 25 benchmark datasets. The experimental results demonstrate that the classification accuracies of the proposed algorithms are superior to the current state-of-the-art methods in tackling the multichotomous question, about 5% in some cases. In particular, according to the proposed methods, the number of candidate cut points converges to a certain extent.

INDEX TERMS Decision tree, classification, unified framework, split criteria.

I. INTRODUCTION

Classification is a commonly used technique in data mining tasks. A classifier is a function which maps input samples into one class label in two or more predictors. There are many different classifier algorithms, such as decision trees [1]–[4], logistic regression [5], [6], support vector machine (SVM) [7], [8], and neural network [9]–[11].

Among them, the decision tree is a simple method and has been widely used in knowledge discovery and pattern recognition. A decision tree is usually constructed by a recursive procedure that optimizes a splitting criterion in accordance with a training dataset that is recursively divided into two or more children of the root node. This recursive procedure is repeatedly implemented to generate partitions until a termination condition met. The decision trees have several advantages, such as superior classification accuracy in many cases compared with other classification models [12], few parameters [13] and easy to understand. Therefore, decision trees are still increasingly applied for various tasks, such as privacy protection [14], [15], biology [16], [17], intrusion detection [18], medicine [19]–[21] and healthcare systems [22].

Traditional heuristic decision tree approaches usually determine the optimal attribute on the basis of the most discriminative ability. The originator of the decision tree, ID3 [23], selects the splitting attribute on the basis of information gain. But ID3 has some deficiencies, including the preference the attributes with more values, hard to handle continuous attributes and the inability to miss values. C4.5 [24], [25] is proposed to improve the continuous attributes and missing values of ID3, which uses the information gain ratio to replace the information gain to identify the optimal attribute. The classification and regression trees (CART) [26] is proposed on the basis of the Gini index to split the input space in a way that maximally reduces the degree of example disorder, and its descendants SLIQ [27] and SPRINT [28] are proposed to effectively improve the learning. Reference [29] proposed a new node splitting measure based on distinct class to improve classification accuracy. Reference [30] presented a novel procedure for building decision tree through handling the imprecision in building decision tree to improve classification performance. Reference [31] proposed a new algorithm, Size Constrained Decision Tree (SCDT), which constructed the decision tree on the

TABLE 1. Six candidate cut points and their corresponding splitting performances and the number of expected segments.

	cp_1	cp_2	cp_3	cp_4	cp_5	cp_6
values	1.20	1.50	1.70	2.00	2.20	2.50
splitting performance	0.80	0.79	0.75	0.68	0.60	0.52
expected segments	80.00	75.00	70.00	65.00	60.00	50.00

basis of the number of leaf nodes. Reference citeb13 proposed an extension of clustering using unsupervised binary trees (CUBT), which primarily used heterogeneity criteria and dissimilarity measures based on mutual information, entropy and Hanming distance. However, for all the above-mentioned methods assume that all of attributes are nominal when selecting the optimal attribute. But for the continuous attributes, discretization [33], [34] should be performed prior to select the best cut point, typically by partitioning the range of the attribute into discrete format, then constructing decision tree like nominal data [35]–[37]. According to the number of intervals for discretizing continuous attributes, it is mainly divided into binary splitting and multiple splitting [38]–[41]. For ease of exposition, in this paper, we mainly focus on binary classification trees.

However, all the heuristic-based methods face a question: if there are two or more candidate cut points that have the same or similar discriminative ability with the splitting criteria, which one would be the most suitable? The traditional way of tackling this problem is to randomly select one to split the node from the multiple cut points with the best splitting performance. However, this may fail to induce a better and smaller tree. Wang *et al.* [31] proposed a two-stage method to handle this issue that is not only effective in improving the generalization capability, but also valid for reducing the size of the tree. Unfortunately, there are two major drawbacks. First, it may be more biased toward the candidate cut point's expected segments and, to some extent, ignores its discriminative ability, particularly for a large $\hat{K}$. For example, we assume that there are six cut points, that is to say, $\hat{K} = 6$, their corresponding splitting performances and the number of expected segments are listed in Table 1. According to [31], cp_6 is determined as the optimal candidate cut point whose the number of expected segments is the smallest. However, there is a big difference between its splitting performance and the maximum. Therefore, cp_6 is not necessarily desired cut point. Second, as the optimal $\hat{K}$ value is strongly associated with the dataset itself, it is difficult to find an appropriate $\hat{K}$ that matches the dataset in advance.

In this paper, we propose a unified model based on splitting performance and the number of expected segments of each candidate cut point. The reason why it's called unified model, because our model can incorporate any univariate decision tree algorithms [42], [43]. However, instead of applying the splitting performance and the number of expected segments directly in [31], we first normalize the number of expected segments, so that it has the same scale with the splitting performance. Then, we integrate these two measures by introducing the weighting factor α to determinate the optimal

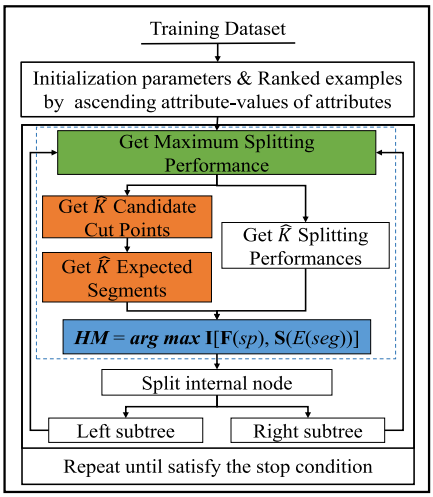

FIGURE 1. Decision trees flowchart based on the unified framework model.

cut point. To demonstrate the effectiveness of the proposed model, we empirically test it on 25 real-world datasets from the University of California, Irvine (UCI) Machine Learning Repository http://archive.ics.uci.edu/ml/datasets.html. The experimental results show that our proposed methods exhibit better performance compared to the baseline methods with respect to the classification accuracy, and an additional benefit is that, under the influence of the weighting factor α, the range of $\hat{K}$ value obviously converges with a fixed value.

The rest of this paper is organized as follows. In Section 2, we describe the proposed unified model in detail. In Section 3, we present some experimental results and a comparison of the proposed methods with several other baseline methods. Finally, in Section 4, we present the conclusion and list future work.

II. THE UNIFIED FRAMEWORK FOR DECISION TREES

The decision tree is to be induced from a training dataset which is represented as $\mathcal{S} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$, where x_i consists of a vector of m conditional attributes $\{a_1, a_2, \cdots, a_m\}$ and $y_i \in \{c_1, c_2, \cdots, c_L\}$ is the corresponding class label $y_i \in \{c_1, c_2, \cdots, c_L\}$ of x_i. We assume that the $\mathcal{S}'$ is a pair set that is sorted by the ascending attribute values under a given attribute a_k of $\mathcal{S}$, and denoted as $\mathcal{S}' = \{(x_1', y_1'), (x_2', y_2'), \cdots, (x_n', y_n')\}$, where $x_1' \leq \cdots \leq x_n'$ with respect to attribute a_k. We assume that a_{ji} is the value of the *ith* example of the *jth* attribute and written as $A_j(x_i)$, and y_i is the class label of x_i and written as $C(x_i)$.

Fig.1 shows the decision tree flowchart framework based on our proposed unified model, which is located in the blue dashed box. Our unified model consists of three modules: (1) splitting performance, (2) expected segments, and (3) proposed unified framework model.

A. SPLITTING PERFORMANCE

For the splitting performance, any one of the univariate splitting criterion, such as information gain, information gain

ratio or Gini index, can be used. In this work, we select information gain ratio as a part of the proposed model. Without any loss of generality, it could be applied to other decision tree algorithms. The information gain ratio can be defined as follows:

$$GainRatio(\mathcal{S}', a'_{ji}) = \frac{InfoGain(\mathcal{S}', a'_{ji})}{SplitInfo(\mathcal{S}', a'_{ji})} \tag{1}$$

where a'_{ji} is a cut point with respect to attribute a_j of $\mathcal{S}'$ and is defined as follows:

$$a'_{ji} = \frac{A_j(x'_i) + A_j(x'_{i+1})}{2}, \quad i \in [1, 2, \ldots, n] j \in [1, 2, \ldots, m] \tag{2}$$

The information gain is defined as follows:

$$InfoGain(\mathcal{S}', a'_{ji}) = Ent(\mathcal{S}') - \sum_{k=1}^{2} \frac{\|\mathcal{S}'_k\|}{\|\mathcal{S}'\|} Ent(\mathcal{S}'_k) \tag{3}$$

where $\|\mathcal{S}'_k\|$ and $\|\mathcal{S}'\|$ are the number of examples in $\mathcal{S}'_k$ and $\mathcal{S}'$. $Ent(\mathcal{S}')$ is the information entropy of $\mathcal{S}'$ and $Ent(\mathcal{S}'_k)$ is the information entropy of *kth* subset which is divided by a'_{ji}. The $Ent(\mathcal{S}')$ is defined as follows:

$$Ent(\mathcal{S}') = -\sum_{l=1}^{L} p_l \log_2 p_l, \quad l \in [1, 2, \ldots, L] \tag{4}$$

where p_l is the frequency of the *lth* class label in $\mathcal{S}'$, which is defined as follows:

$$p_l = \frac{\sum_{i=1}^{n} [I(C(x'_i) = l)]}{\|\mathcal{S}'\|} \tag{5}$$

where $I(C(x'_i) = l)$ is an indicator function which means that it is 1 if the class label of $C(x'_i)$ is l; otherwise, 0.

The splitting information is defined as follows:

$$SplitInfo(\mathcal{S}', a'_{ji}) = -\sum_{k=1}^{2} \frac{\|\mathcal{S}'_k\|}{\|\mathcal{S}'\|} \log_2 \frac{\|\mathcal{S}'_k\|}{\|\mathcal{S}'\|} \tag{6}$$

where $\|\mathcal{S}'_k\|$ and $\|\mathcal{S}'\|$ are the number of examples in $\mathcal{S}'_k$ and $\mathcal{S}'$, respectively.

B. EXPECTED SEGMENTS

Supposing that there exists two adjacent examples $x'_i, x'_{i+1} \in \mathcal{S}'$, having different classes, and if a'_{ji} satisfies $A_j(x'_i) < a'_{ji} < A_j(x'_{i+1})$ then a'_{ji} is referred to as a boundary point with respect to attribute a_j of $\mathcal{S}'$. It is easy to know that irrespective of the number of classes and how they are distributed, the optimal cut point will always occur on the boundary point between two classes [38]. Obviously, the less number of boundary points, the easier it is to split the training examples and the less depth of the constructed decision trees. To further describe the relationship between the permutation information and the boundary points of the training examples, we introduce the concept of *segment*.

Definition Segment: Assuming that the first and the last examples' attribute values in $\mathcal{S}'$ are regarded as the first and the last boundary point, respectively. We call the example sequence set between any two adjacent boundary points a *segment*.

The number of segments in $\mathcal{S}'$ is defined as follows:

$$Seg(\mathcal{S}') = \min_j \|Seg(\mathcal{S}', a_j)\|, \quad j \in [1, 2, \ldots, m] \tag{7}$$

In the ideal case, that is, when all the attribute-values are different, the number of segments of the ranked training examples is one less than the number of its boundary points. However, if there exists duplicated values for some examples, it is not sufficient to evaluate the discriminative ability of a_j. To handle this issue, we use the concept of *bar* from [31], which is defined as a sequence of examples ζ with the same attribute value under a given attribute. The number of segments in a *bar* $\mathcal{B}$ is defined as follows:

$$bSeg(\mathcal{B}) = \|Seg(\zeta)\| \tag{8}$$

How to compute the number of segments of a *bar* refer to [31].

Suppose that t is the number of bar in $\mathcal{S}'$ with respect to a_j and u is the number of non-bar sub-queues, which is denoted as $\mathfrak{S}_{*,j}$. Therefore, the number of segments in $\mathcal{S}'$ induced by attribute a_j, denoted by Seg($\mathcal{S}', a_j$), is computed as follows:

$$Seg(\mathcal{S}', a_j) = \sum_{i=1}^{u} \|Seg(\mathfrak{S}_{i,j})\| + \sum_{l=1}^{t} bSeg(\mathcal{B}_l) \tag{9}$$

Fig.2 shows distribution of sixteen examples, $\mathbb{S}$, which are ranked ascending order by values of attribute a_j. The second line represents the distribution of class labels with respect to examples and the third line represents their attribute values. From Fig.2, we can see that there are two bar points, eg., $\zeta_{1,j}$, $\zeta_{2,j}$, whose attribute values are 1.5 and 3.3, and three non-bar sub-queues, eg., $\mathfrak{S}_{1,j}$, $\mathfrak{S}_{2,j}$, $\mathfrak{S}_{3,j}$. The two bar sub-queues are $\mathcal{B}_1 = \zeta_{1,j} = (x_3, x_4, x_5)$ and $\mathcal{B}_2 = \zeta_{2,j} = (x_{12}, x_{13}, x_{14})$. The three non-bar sub-queues are $\mathfrak{S}_{1,j} = (x_1)$, $\mathfrak{S}_{2,j} = (x_6, x_7, x_8, x_9, x_{10})$ and $\mathfrak{S}_{3,j} = (x_{15}, x_{16})$. Then the number of segments of $\mathbb{S}$ is calculated by Equation (9):

$$\begin{aligned} Seg(\mathbb{S}, a_j) &= \|Seg(\mathfrak{S}_{1,j})\| + \|Seg(\mathfrak{S}_{2,j})\| + \|Seg(\mathfrak{S}_{3,j})\| \\ &\quad + bSeg(\mathcal{B}_1) + bSeg(\mathcal{B}_2) \\ &= 1 + 4 + 2 + 3 + 3 \\ &= 13 \end{aligned}$$

We denote the number of expected segments at the cut point a'_{ji} with respect to attribute a_j of $\mathcal{S}'$ as E($\mathcal{S}', a'_{ji}$) and it is defined as follows:

$$E(\mathcal{S}', a'_{ji}) = \frac{\|\mathcal{S}'_1\|}{\|\mathcal{S}'\|} \|Seg(\mathcal{S}'_1, a'_{ji})\| + \frac{\|\mathcal{S}'_2\|}{\|\mathcal{S}'\|} \|Seg(\mathcal{S}'_2, a'_{ji})\| \tag{10}$$

where $(\mathcal{S}'_1)$ and $(\mathcal{S}'_2)$ are two subsets of $\mathcal{S}'$ which is divided by a'_{ji}, while $\|\mathcal{S}'_1\|$ and $\|\mathcal{S}'_2\|$ represent the number of segments of $\mathcal{S}'_1$ and $\mathcal{S}'_2$, respectively.

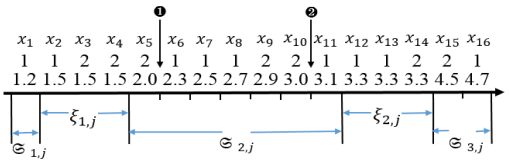

FIGURE 2. Distribution of sixteen examples ranked ascending order by attribute values and two cut points.

C. PROPOSED UNIFIED FRAMEWORK MODEL

1) REPRESENT CANDIDATE CUT POINTS

In general, a candidate cut point is usually represented as a numerical value that is between the minimal attribute value and the maximal attribute value [45], [46]. In this paper, we describe it as a binary pair, denoted as (*sp*, *E(seg)*), where *sp* and *E(seg)* represent its splitting performance and the number of expected segments, respectively. For example, Fig.2 shows two candidate cut points in the sorted sixteen examples, e.g., $cp_1 = 2.15$ and $cp_2 = 3.05$, respectively. According to Equation (1) and Equation (10), we can easily determine their splitting performances and the number of expected segments, $sp_1 = 0.0392$, $E(seg_1) = 4.06$, and $sp_2 = 0.0203$, $E(seg_2) = 3.63$, respectively. Therefore, we replace cp_1 and cp_2 with (0.0392,4.06) and (0.0203,3.63).

2) PROPOSED FRAMEWORK MODEL

We propose a unified framework model for decision trees on the basis of the splitting performance and the number of expected segments for each candidate cut point. The unified model is defined as follows:

$$HM = arg\max I(F(sp), S(E(seg))) \tag{11}$$

where *F(sp)* is a function which takes any impurity measurement as a variable and *S(E(seg))* is defined a function of the number of expected segments. According to Equation (11), the optimal candidate cut point must satisfy both of the following conditions:

(1) The splitting performance of the cut point should be as close as possible to the optimal value, eg., information gain ratio for C4.5, so that this model has relatively better classification performance [1], [2], [44].

(2) The number of expected segments of the cut point should be as small as possible, so that the smaller of the decision tree constructed will be [31].

To balance these two aspects, the most commonly used method is to introduce a weighting factor $\alpha \in [0,1]$ for the splitting performance and 1-α for the number of expected segments. Therefore, the unified model can be defined as follows:

$$HM = arg\max \alpha * F(sp) + (1-\alpha) * S(E(seg)) \tag{12}$$

where $F(sp)$ and $S(E(seg))$ can be defined as follows:

$$F(sp) = \exp(sp) \tag{13}$$

$$S(E(seg)) = \exp(E(seg)) \tag{14}$$

The problem reduces to finding a suitable cut point which maximize the *HM*.

However, a problem still remains. Suppose that *cp* is a candidate cut point, and its $sp = 0.50$, $E(seg) = 10.00$. If $\alpha = 0.5$, then $HM = 0.5 * e^{0.50} + 0.5 * e^{10.00} \approx 0.82 + 22026.47 = 22027.29$, where the value of *F(sp)*, 0.82, is so small that it can be ignored relative to the value of *S(E(seg))*, 22026.47. In that case, the selection of the optimal cut point severely biased towards that with the maximum expected segments. In order to tackle the aforementioned question, the priority is to constrain the number of expected segments by a normalization method to make it have the same order of magnitude as splitting performance.

Two different normalization methods are proposed for the unified model. This produces two novel algorithms that we call them, SPES1 and SPES2. The first normalization method is defined as follows:

$$p_i = E(seg)_i / \sum_{i=1}^{\hat{K}} E(seg)_i, \quad p_i \in (0,1) \tag{15}$$

where $\hat{K}$ is the number of candidate cut points whose splitting performances are closest to the optimal and $\sum_{i=1}^{\hat{K}} p_i = 1$. Since we are more inclined to the smaller expected segments, we, then, need to add a negative sign before them. Therefore, we can rewrite Equation (14) as follows:

$$S(E(seg)) = \exp(-p), \quad p \in (0,1) \tag{16}$$

Therefore, the SPES1 can be defined as follows:

$$HM = arg\max \alpha * exp(sp) + (1-\alpha) * exp(-p) \tag{17}$$

Apparently, Equation (17) can determine the optimal cut point when the cut point has the maximum splitting performance and the minimal number of expected segment.

The second normalization method is defined as follows:
(1) Constitute a vector $\bar{E}(seg)$ by the $\hat{K}$ expected segments:

$$\bar{E}(seg) = [E(seg)_1, E(seg)_2, \ldots, E(seg)_{\hat{K}}] \tag{18}$$

(2) Calculate reciprocal of the vector $\bar{E}(seg)$ and obtain $\bar{r}$:

$$\bar{r} = \frac{1}{\bar{E}(seg)} = [\frac{1}{E(seg)_1}, \frac{1}{E(seg)_2}, \ldots, \frac{1}{E(seg)_{\hat{K}}}] \tag{19}$$

(3) Normalize each of the entries in $\bar{r}$:

$$\hat{r}_i = r_i / \sum_{i=1}^{\hat{K}} r_i, \; r_i \in (0,1), \; i \in [1,2,\ldots,\hat{K}] \tag{20}$$

Obviously, $\sum_{i=1}^{\hat{K}} \hat{r}_i = 1$
Then, SPES2 can be defined as follows:

$$HM = arg\max \alpha * exp(sp) + (1-\alpha) * exp(\hat{r}) \tag{21}$$

Let $\mathcal{S}'$ be the given dataset, our new split criteria as follow:
First, the best sp^* is calculated as follows:

$$sp^* = \max_{\{j,i\}} GainRatio(\mathcal{S}', a'_{ji}) \tag{22}$$

where GainRatio($\mathcal{S}'$, a'_{ji}) is defined in Equation (1).

Algorithm 1 Splitting Performance and Expected Segments-Based DT for Continuous Valued Attributes

Input: training examples $\mathcal{S} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$ with m continuous values attributes $a = \{a_1, a_2, \cdots, a_j\}$ and one decision attribute $y_i \in \{c_1, c_2, \cdots, c_L\}$; threshold number $\hat{N}$ to stop splitting a node and parameter α and $\hat{K}$.

Output: A binary decision tree.

createDecisionTree($\mathcal{S}$)
If all the examples in $\mathcal{S}$ are from the same class l^* **then**
    Treat $\mathcal{S}$ as a leaf node and assign it the class label l^*.
If $\|\mathcal{S}\| < \hat{N}$ **then**
    Treat $\mathcal{S}$ as a leaf node and assign it the class label $l^* = \arg\max_{l=1,2,\cdots,L} p_l$.
For each attribute of $\mathcal{S}$, a_j:
    sorted the examples in ascending order by attribute values and denoted as $\mathcal{S}'$.
Get the cut points of each attribute by Equation (2), i.e., a'_{ji}.
Calculate the splitting performances of each candidate cut point by Equation (1).
Get the maximum sp^* by Equation (22).
Get $\hat{K}$ candidate cut points with splitting performances closest to sp^*.
Calculate the number of expected segments for each candidate cut point by Equation (10).
Determine the optimal cut point $a_{j^*i^*}$ by Equation (23).
Split $\mathcal{S}'$ into two child-nodes by $a_{j^*i^*}$,
    $\mathcal{S}'_1 = \{x'_j \in \mathcal{S}' \mid A_{j^*}(x'_i) \le s_{j^*i^*}\}$ and $\mathcal{S}'_2 = \{x'_j \in \mathcal{S}' \mid A_{j^*}(x'_i) > s_{j^*i^*}\}$.
createDecisionTree($\mathcal{S}'_1$).
createDecisionTree($\mathcal{S}'_2$).

Second, $\hat{K}$ candidate cut points with splitting performances closest to sp^* are selected from all of the cut points calculated by using Equation (2).

Third, The number of expected segments of $\hat{K}$ candidate cut points are calculated by Equation (10) and normalized.

Finally, the optimal candidate cut point cp^* is derived using:

$$cp^* = arg \max_{i\in[1,2,\ldots,\hat{K}]} \begin{cases} \alpha * exp(sp_i) + (1-\alpha) * exp(-p_i) \\ \alpha * exp(sp_i) + (1-\alpha) * exp(\hat{r}_i) \end{cases} \tag{23}$$

The unified model-based decision tree with continuous-value attributes of C4.5 is described in **Algorithm 1**.

III. EXPERIMENTS AND ANALYSIS

A. DATASETS AND EXPERIMENTAL SETUPS

The experiments were conducted on a set of real-world datasets from UCI Machine Learning Repository including 15 binary and 10 multiclass. The details of these datasets are presented in Table 2. As not all of the datasets are continuous, we preprocessed them first. We assume that if discrete values in an attribute are fewer than ten, the attribute will be removed from the dataset. Moreover, for each attribute, each input value was normalized to [0,1] by 1-$((v_{max} - v)/(v_{max} - v_{min}))$, where v is the value to be normalized. We set $\hat{K} = \{2, 4, 6, 8, 10, 15, 20, 30, 40, 50\}$ and $\alpha = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ for segment-based algorithm and our proposed algorithms. We selected the values with the highest test accuracy as the final parameters (Table 5). We set the terminal condition for splitting internal node when the number of examples in it was less than five, that is, $\hat{N} = 5$.

A standard 10-fold cross-validation was performed on each dataset per classification model, and the average value of the 10 results was considered the test result. However, there are some highly imbalanced datasets, that are not sufficient to conduct the 10-fold cross-validation. In this case, we conducted 5 × 2-fold cross-validation and observe the average value of 5 × 2 = 10 results. To ensure the validation of the experimental results, each method was implemented on the same training set and test set. We evaluated the performance of the proposed methods in terms of generalization capability by using the test accuracy and model complexity by using depth and the number of nodes of decision tree.

The experiments were performed on Python 3, which were executed on a computer with a 3.20GHz Intel®Core(TM) i5-6500 CPU, a maximum 4.00GB memory, and 64-bit Windows 7 system.

B. COMPARISON OF SIMULATION RESULTS

To validate the practical performance of the proposed algorithms, we compared with several classifiers, including ID3, C4.5, CART, and segment-based algorithm. The prediction accuracies of each classification model for all of the datasets are tabulated in Table 3. The best values are shown in bold.

Fig.3 demonstrates that the average relative reduction scales in the tree depths, number of nodes, and test accuracy of our proposed methods and Segment+C4.5 compared with C4.5 with the optimal parameters of each dataset. Suppose that $Acc_{C4.5}$ is the accuracy of C4.5, and that of Segment+C4.5 is Acc_{seg}; then, the relative reduction

TABLE 2. Description of the datasets.

No.	Datasets (Abbr.)	Examples	Continuous Attributes	Classes	Class Distribution
1	Haberman	306	3	2	225/81
2	Ionosphere	351	32	2	225/126
3	Cancer	683	8	2	444/239
4	Australian	690	7	2	307/383
5	German	1000	3	2	700/300
6	Liver Disorders (Bupa)	345	6	2	145/200
7	Heart	270	5	2	150/120
8	Breast Cancer Wisconsin (Wdbc)	569	30	2	212/357
9	Pima	768	8	2	268/500
10	Bands	365	16	2	230/135
11	Credit Approval (Crx)	635	6	2	357/296
12	Appendicits	106	7	2	85/21
13	South African Hearth (Saheart)	462	8	2	302/160
14	Spectheart	267	44	2	212/55
15	Glass	214	9	2	144/70
16	Ecoli	336	5	8	143/77/2/2/35/20/5/52
17	Libras	360	90	15	24×15
18	Vowel	990	10	11	90×11
19	Yeast	1484	6	10	244/429/463/44/51/163/35/30/20/5
20	Automobile	159	15	6	48/46/29/20/13/3
21	Segment	2310	16	7	330×7
22	Wine	178	14	3	59/71/48
23	Thyroid	215	5	3	150/35/30
24	Iris	150	4	3	50×3
25	Vehicle	846	18	4	218/217/212/199

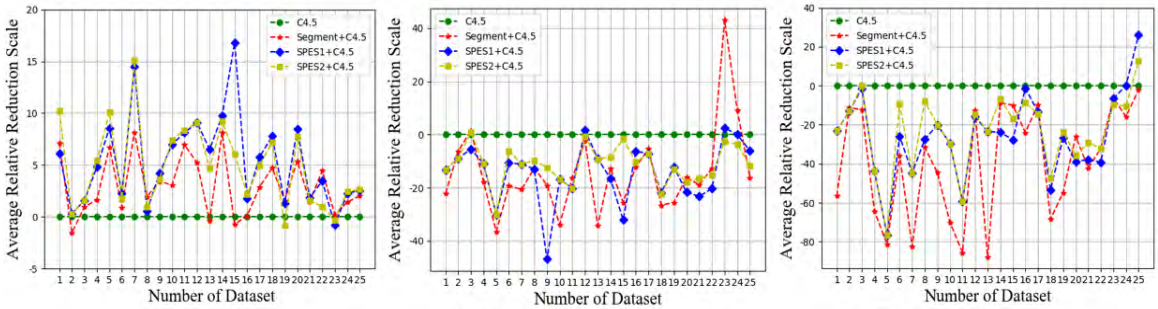

FIGURE 3. Average relative reduction in accuracy and decision tree scale of Segment+C4.5, SPES1+C4.5 and SPES2+C4.5 compared with C4.5.

scale in the accuracy of Segment+C4.5 is calculated as $(Acc_{seg} - Acc_{C4.5})/Acc_{C4.5}$. Similar calculations were also applied to SPES1+C4.5, SPES2+C4.5, the number of nodes and the decision tree depths. If the relative reduction scale is below zero, the measured value decreases; otherwise, it is increased.

Fig.3(a) shows that the test accuracy average variation ratio of our proposed methods and Segment+C4.5 compared with C4.5 for each datasets. The test accuracies obtained by our proposed methods (SPES1/SPES2) were higher than those obtained using the Segment+C4.5 algorithm's 21/21 datasets in 25 test datasets, and were superior to C4.5's 23/24 datasets in 25 test datasets. These findings validated our view that the decision tree constructed on the basis of the algorithm of the splitting performance and the expected segments exhibited better classification performance. In addition, it proved that a candidate cut point with the smallest number of expected segments is not necessarily the optimal cut point.

Fig.3(b) shows that the comparison of the depths of the decision tree with C4.5, Segment+C4.5 and our proposed algorithms. Obviously, the depths of the decision trees constructed by our algorithms were larger than the depths of the decision trees constructed by the Segment+C4.5 algorithm in most of the datasets, except for Ionosphere, Balana, Saheart, Cancer, Libras, Automobile and Wine. However, it was clear that the depth of the decision tree constructed for each dataset by using C4.5 was greater than those of the datasets built using our proposed algorithms and Segment+C4.5 algorithm.

Fig.3(c) shows that the average reduction scale of the number of nodes of the decision trees constructed using C4.5, Segment+C4.5 and the proposed algorithms. We observed that although the number of nodes in the decision trees constructed by our method and the segment-based method were smaller than that of the decision tree constructed using C4.5 for most of datasets, they evenly matched, 13 datasets and 14 datasets, respectively.

Table 3 summarizes the average test accuracy of the 10-fold cross-validation results of different classification models with the optimal parameters. The last line of Table 3

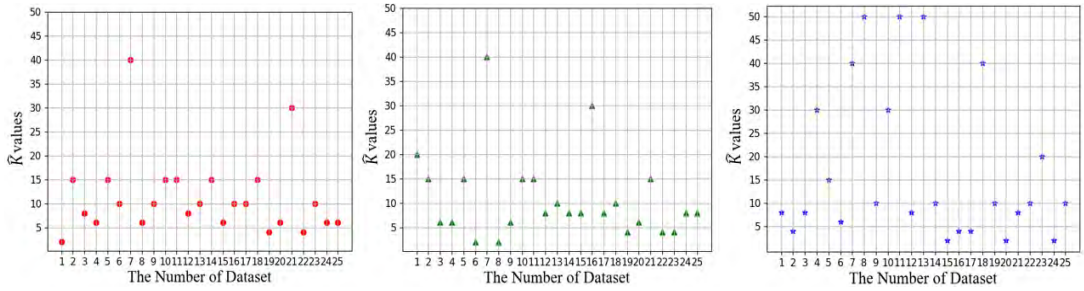

FIGURE 4. Distribution of $\hat{K}$ values of different methods for each dataset.

TABLE 3. Comparison of different decision tree methods: test accuracy.

Datasets	ID3	C4.5	CART	Segment+C4.5	SPEP1+C4.5	SPEP2+C4.5
Haberman	0.6400	0.6533	0.6500	0.6800	**0.6933**↑	**0.7200**↑
Ionosphere	0.9086	0.9343	0.8886	0.9200	**0.9371**↑	**0.9371**↑
Cancer	0.9500	0.9397	0.9515	0.9485	**0.9544**↑	**0.9544**↑
Australian	0.7116	0.6928	0.7188	0.7043	**0.7261**↑	**0.7304**↑
German	0.6180	0.5960	0.6140	0.6360	**0.6470**↑	**0.6560**↑
Bupa	0.6588	0.6559	0.6118	0.6618	**0.6706**↑	**0.6676**↑
Heart	0.6000	0.5889	0.6259	0.6270	**0.6741**↑	**0.6778**↑
Wdbc	0.9304	0.9232	0.9393	**0.9411**	0.9286↓	0.9321↓
Pima	0.7013	0.6921	0.7184	0.7158	**0.7211**↑	**0.7171**↑
Bands	0.6667	0.6361	0.6361	0.6556	**0.6806**↑	**0.6833**↑
Crx	0.6662	0.6631	0.6554	0.7092	**0.7169**↑	**0.7185**↑
Appendicits	0.8300	0.7700	0.7900	0.8100	**0.8400**↑	**0.8400**↑
Saheart	0.6239	0.6022	0.6130	0.6000	**0.6413**↑	**0.6304**↑
Spectheart	0.7462	0.7115	0.7231	0.7692	**0.7808**↑	**0.7769**↑
Glass	**0.7524**	0.7095	0.7190	0.7043	0.7143↑	**0.7524**↑
Ecoli	0.7274	0.7440	0.7381	0.7448	**0.7571**↑	**0.7601**↑
Libras	0.5167	0.5367	0.5389	0.5522	**0.5678**↑	**0.5633**↑
Vowel	**0.7172**	0.6651	0.6978	0.6966	0.7168↑	0.7131↑
Yeast	0.4968	0.4943	0.4830	**0.5021**	0.5008↓	0.4942↓
Automobile	0.6937	0.6582	**0.7418**	0.6937	0.7139↑	0.7089↑
Segment	0.9484	0.9368	0.9479	0.9506	**0.9541**↑	**0.9515**↑
Wine	0.9079	0.9101	0.9191	**0.9506**	0.9416↓	0.9491↓
Thyroid	0.9121	0.9159	**0.9178**	**0.9178**	0.9084↓	0.9131↓
Iris	0.9573	0.9360	0.9547	0.9493	**0.9573**↑	**0.9587**↑
Vehicle	**0.7069**	0.6733	0.6851	0.6870	0.7013↑	0.7010↑
Average	0.7435	0.7296	0.7392	0.7462	**0.7664**	**0.7642**

shows that the average performances over the datasets for each classifier. On the basis of these simulation results, we drawn a conclusion that SPES1+C4.5 exhibited the best classification performance in 17 of the 25 test benches, while SPES2+C4.5 exhibited in 18 of the 25 benches. This showed that our proposed methods were superior to other methods. Furthermore, ↑ and ↓ were used to demonstrate whether the proposed algorithms improved the performance of the segment-based algorithm. Obviously, the proposed methods had higher accuracy than the segment-based method, 20 for SPES1 and 21 for SPES2, respectively. This proved that a candidate cut point with the smallest expected segments is not necessarily for the optimal one. Therefore, when choosing an optimal cut point, we should consider not only the number of expected segments of the cut point but also its splitting performance.

Table 4 reports the average tree depth and the number of nodes for each method. For each dataset, the minimal values were highlighted in bold face. It is observed that the methods that achieved the smallest tree depth and the number of nodes for the different datasets are considerably different. In fact, each method can induce the smallest tree in some cases. Among them, ID3 provided the lowest average value. However, the comparative result between Segment+C4.5 and the proposed methods was clear. In the last three columns of Table 4, ↑ and ↓ were used to demonstrate whether our method can reduce the tree size compared with segment-based. Obviously, in most cases, the decision trees constructed using the proposed methods were more complex than segment-based algorithm irrespective of the depth or the number of nodes of the decision trees.

C. COMPARISON OF $\hat{K}$ BETWEEN SEGMENT + C4.5 AND PROPOSED METHODS

Table 5 lists the distribution of the optimal $\hat{K}$ values of the segment-based method and the proposed methods. The average values of $\hat{K}$ in the case of SPES1 and SPES2 were 11.28 and 10.92, while that in the case of Segment+C4.5 was

TABLE 4. Comparisons of different decision trees methods: tree scale.

Datasets	ID3		C4.5		CART		segment+C4.5		SPEP1+C4.5		SPEP2+C4.5	
	Nodes	Depth	Nodes	Depth	Nodes	Depth	Nodes	Depth	Nodes	Depth	Nodes	Depth
Haberman	126.60	14.70	149.00	23.70	124.20	13.60	116.00	13.30	129.40↑	18.20↑	129.40↑	18.20↑
Ionosphere	35.60	8.10	38.20	16.10	41.00	9.40	35.80	14.30	34.80 ↓	14.00↓	34.80↓	14.00↓
Cancer	44.60	8.20	51.80	8.10	50.600	7.40	52.60	7.10	49.00 ↓	8.00 ↑	52.20↓	8.10 ↑
Australian	199.40	16.30	252.20	56.50	200.40	15.60	207.40	20.10	224.60↑	31.80↑	224.60↑	31.80↑
German	435.00	23.40	596.00	202.70	424.40	23.00	378.80	38.00	417.20↑	47.60↑	417.60↑	47.60↑
Bupa	116.20	12.00	181.60	69.20	116.60	10.90	146.60	44.50	162.20↑	51.30↑	170.20↑	62.80↑
Heart	107.00	12.90	134.00	41.80	107.60	12.90	106.40	7.30	119.20↑	23.10↑	119.20↑	23.10↑
Wdbc	34.20	6.30	51.00	14.90	38.60	7.80	44.60	10.20	44.40 ↓	10.80↑	46.00↑	13.70↑
Pima	204.20	15.40	290.20	80.10	199.20	13.50	235.00	44.40	254.40↑	64.00↑	254.40↑	64.00↑
Bands	108.60	15.70	188.80	79.50	113.60	12.60	125.00	23.80	157.00↑	56.00↑	157.00↑	56.00↑
Crx	194.20	17.60	261.40	71.40	200.40	16.20	218.80	10.30	208.80↓	29.10↑	208.80 ↓	29.10↑
Appendicits	23.00	5.90	25.20	8.80	23.40	6.40	24.60	7.70	25.60 ↑	7.40 ↓	25.00↑	7.50 ↓
Saheart	144.20	17.10	241.20	95.90	144.00	13.80	158.60	11.50	219.40↑	73.40↑	219.40↑	73.40↑
Spectheart	44.80	11.10	66.60	25.10	47.80	7.70	58.20	22.90	55.60 ↓	19.10↑	61.00↑	23.40↑
Glass	53.60	9.40	63.60	17.00	54.40	9.40	47.40	15.30	43.20 ↓	12.30↓	62.60↑	14.10↑
Ecoli	53.00	7.60	62.60	15.00	55.80	8.00	55.00	11.40	58.60 ↑	14.80↑	56.20↑	13.70↑
Libras	78.60	8.40	97.00	20.80	79.40	10.00	91.80	18.80	90.20 ↓	18.00↓	90.00↓	17.80↓
Vowel	159.40	9.60	220.20	49.40	168.20	12.20	161.80	15.60	172.20↑	23.00↑	170.80↑	26.00↑
Yeast	379.80	19.20	468.60	73.00	378.20	21.20	349.00	33.00	411.40↑	53.60↑	407.60↑	55.70↑
Automobile	33.00	5.60	42.60	15.40	32.60	6.80	35.80	11.40	33.40 ↓	9.40↓	35.00↓	9.90↓
Segment	80.60	11.20	111.80	27.00	88.20	13.40	90.60	15.60	85.80 ↑	16.80↑	93.40↑	19.10↑
Wine	12.20	3.20	15.80	5.60	13.40	3.60	13.80	3.80	12.60 ↓	3.40 ↓	13.40↓	3.80
Thyroid	13.40	3.80	15.80	6.20	13.40	4.00	22.60	5.80	16.20 ↑	5.80	15.40↓	5.60 ↓
Iris	10.20	3.40	11.00	3.80	10.60	3.60	12.00	3.20	11.00 ↓	3.80 ↑	10.60↓	3.40 ↑
Vehicle	127.00	13.60	168.60	26.80	129.00	14.00	141.40	26.20	158.20↑	33.80↑	148.80↑	30.20↑

TABLE 5. Distribution of α and $\hat{K}$ of each datasets.

Datasets	SPEP1		SPEP2		Segment
	α	$\hat{K}$	α	$\hat{K}$	$\hat{K}$
Haberman	0.3	2	0.5	20	8
Ionosphere	0.1	15	0.9	15	4
Cancer	0.9	8	0.9	6	8
Australian	0.3	6	0.7	6	30
German	0.1	15	0.9	15	15
Bupa	0.6	10	0.8	2	6
Heart	0.2	40	0.8	40	40
Wdbc	0.1	6	0.6	2	50
Pima	0.1	10	0.9	6	40
Bands	0.1	15	0.9	15	30
Crx	0.1	15	0.9	15	50
Appendicits	0.2	8	0.8	8	8
Saheart	0.5	10	0.5	10	50
Spectheart	0.1	15	0.8	8	10
Glass	0.1	6	0.8	8	2
Ecoli	0.7	10	0.1	30	4
Libras	0.2	10	0.9	8	4
Vowel	0.1	15	0.9	10	40
Yeast	0.2	4	0.8	4	10
Automobile	0.1	6	0.9	6	2
Segment	0.1	30	0.9	15	8
Wine	0.1	4	0.9	4	10
Thyroid	0.1	10	0.9	4	20
Iris	0.1	6	0.9	8	2
Vehicle	0.1	6	0.9	8	2

17.24. It was about more than 6 on average than ours. There are two advantages. First, a small $\hat{K}$ value decrease the computational cost and the time cost considerably. The other and more important is that a large $\hat{K}$ includes more candidate cut points, and the splitting performances of the latter ones differ considerably from the maximum. In this case, the newly added candidate cut points, themselves, are not reasonable.

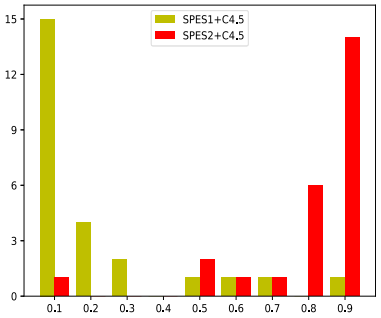

FIGURE 5. Distribution of α for the proposed methods.

Therefore, a smaller $\hat{K}$ helps maintain the discriminative ability to differentiate the examples as much as possible.

Fig.4 shows the scatter diagram of the $\hat{K}$ values of Segment+C4.5 and our methods. Each point (irrespective of whether red, green, or blue) represented the optimal $\hat{K}$ value of a dataset. Figures 4(a) and (b) show that the optimal $\hat{K}s$ were mainly concentrated in [2,15], 23 for SPES1 and 22 for SPES2, respectively. Figure 4(c) shows the distribution range of $\hat{K}$s generated by the segment-based method was wide, from $\hat{K} = 2$ to $\hat{K} = 50$. Therefore, we concluded that the proposed methods had a certain degree of convergence on $\hat{K}$ under the influence of the weighting factor α.

D. ANALYSIS OF WEIGHTING FACTOR α

Table 5 lists the values of α for the proposed methods when all of the datasets exhibited their best test accuracy. Fig.5 characterizes the distribution of α values by a histogram graphic. From Fig.5, we observed that the values of α were mainly concentrated in 0.1 and 0.2 for SPES1, in the case

of 15 datasets and 4 datasets, respectively, accounting for 76% of all the test datasets. While SPES2 exhibited the opposite trend, which were focused on 0.9 and 0.8, in the case of 14 datasets and 6 datasets, respectively, occupying 80%. As we can see from Equation (12), when α is greater than 0.5, the selection of the optimal cut point is more inclined to frequency information. In contrast, it is more biased towards the segment information. According to the completely opposite results of SPES1 and SPES2, this indicated that the value of α depends on the algorithm design itself in the case of the same dataset.

IV. CONCLUSION

In this paper, we propose a unified model for decision trees based on the splitting performance and the number of expected segments for each candidate cut point by introducing a weighting factor α. Then, we design two algorithms based on two normalization methods. To verify the validity of the methods, we apply them to 25 real-world datasets. The experimental results indicate that the proposed methods are superior to the state of the art in terms of the classification performance. In addition, we analyze and discuss two hyper-parameters α and $\hat{K}$. We observe some benefits from the experimental results: First, the values of α are related to the dataset and the algorithm itself. Second, the values of $\hat{K}$ are mainly clustered within a fixed value, from which we conclude that the proposed methods have a certain degree of convergence. There are several possible research issues regarding this topic for further study. First, it might be interesting to extend the future work to a multi-splitting environment with mixed types of attributes. Second, a more systematic and theoretical analysis on the two hyper-parameters is necessary.

REFERENCES

[1] K. Kim, "A hybrid classification algorithm by subspace partitioning through semi-supervised decision tree," *Pattern Recognit.*, vol. 60, pp. 157–163, Dec. 2016.
[2] X. Guan, J. Liang, Y. Qian, and J. Pang, "A multi-view OVA model based on decision tree for multi-classification tasks," *Knowl.-Based Syst.*, vol. 138, pp. 208–219, Dec. 2017.
[3] C.-C. Wu, Y.-L. Chen, Y.-H. Liu, and X.-Y. Yang, "Decision tree induction with a constrained number of leaf nodes," *Appl. Intell.*, vol. 45, pp. 673–685, Oct. 2016.
[4] M. Jaworski, P. Duda, and L. Rutkowski, "New splitting criteria for decision trees in stationary data streams," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2516–2529, Jun. 2018.
[5] L. Yang and Y. Qian, "A sparse logistic regression framework by difference of convex functions programming," *Appl. Intell.*, vol. 45, no. 2, pp. 241–254, 2016.
[6] Y. Yang and M. Loog, "A benchmark and comparison of active learning for logistic regression," *Pattern Recognit.*, vol. 83, pp. 401–415, Nov. 2018.
[7] S. Paul, M. Magdon-Ismail, and P. Drineas, "Feature selection for linear svm with provable guarantees," *Pattern Recognit.*, vol. 60, pp. 205–214, Dec. 2016.
[8] Y. Yin, D. Xu, X. Wang, and M. Bai, "Online state-based structured SVM combined with incremental PCA for robust visual tracking," *IEEE Trans. Cybern.*, vol. 45, no. 9, pp. 1988–2000, Sep. 2015.
[9] K. Anam and A. Al-Jumaily, "Evaluation of extreme learning machine for classification of individual and combined finger movements using electromyography on amputees and non-amputees," *Neural Netw.*, vol. 85, pp. 51–68, Jan. 2017.
[10] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
[11] G. Bologna and Y. Hayashi, "Characterization of symbolic rules embedded in deep DIMLP networks: A challenge to transparency of deep learning," *J. Artif. Intell. Soft Comput. Res.*, vol. 7, no. 4, pp. 265–286, 2017.
[12] T.-S. Lim, W.-Y. Loh, and Y.-S. Shih, "An empirical comparison of decision trees and other classification methods," Dept. Statist., Univ. Wisconsin-Madison, Madison, WI, USA, Tech. Rep. 979, 1997.
[13] J. Gehrke, V. Ganti, R. Ramakrishnan, and W.-Y. Loh, "BOAT—Optimistic decision tree construction," in *Proc. Int. Conf. Manage. Data (SICMOD)*, 1999, pp. 169–180.
[14] P. K. Fong and J. H. Weber-Jahnke, "Privacy preserving decision tree learning using unrealized data sets," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 2, pp. 353–364, Feb. 2011.
[15] R. Baghel and M. Dutta, "Privacy preserving classification by using modified C4.5," in *Proc. 6th Int. Conf. Contemp. Comput.*, Aug. 2013, pp. 124–129.
[16] T. Lajnef *et al.*, "Learning machines and sleeping brains: Automatic sleep stage classification using decision-tree multi-class support vector machines," *J. Neurosci. Methods*, vol. 250, pp. 94–105, Jul. 2015.
[17] H. E. Kretser *et al.*, "Mobile decision-tree tool technology as a means to detect wildlife crimes and build enforcement networks," *Biol. Conservat.*, vol. 189, pp. 33–38, Sep. 2015.
[18] S. Benferhat, A. Boudjelida, K. Tabia, and H. Drias, "An intrusion detection and alert correlation approach based on revising probabilistic classifiers using expert knowledge," *Appl. Intell.*, vol. 38, no. 4, pp. 520–540, 2013.
[19] P. U. Kasbekar, P. Goel, and S. P. Jadhav, "A decision tree analysis of diabetic foot amputation risk in indian patients," *Frontiers Endocrinology*, vol. 8, no. 5, p. 25, 2017.
[20] M. K. Bajre *et al.*, "Expanding the role of radiographers in reporting suspected lung cancer: A cost-effectiveness analysis using a decision tree model," *Radiography*, vol. 23, no. 4, pp. 273–278, 2017.
[21] V. Schetinin, L. Jakaite, and W. Krzanowski, "Bayesian averaging over Decision Tree models for trauma severity scoring," *Artif. Intell. Med.*, vol. 84, pp. 139–145, Jan. 2017.
[22] O. Ben-Assuli and M. Leshno, "Using electronic medical records in admission decisions: A cost effectiveness analysis," *Decis. Sci.*, vol. 44, no. 3, pp. 463–481, 2013.
[23] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986.
[24] S. L. Salzberg, "C4.5: Programs for machine learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993," *Mach. Learn.*, vol. 16, no. 3, pp. 235–240, 1994.
[25] J. R. Quinlan, "Improved use of continuous attributes in C4.5," *J. Artif. Intell. Res.*, vol. 4, pp. 77–90, 1996.
[26] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Belmont, CA, USA: Wadsworth International Group, 1984.
[27] M. Mehta, R. Agrawal, and J. Rissanen, "SLIQ: A fast scalable classifier for data mining," in *Proc. Int. Conf. Extending Database Technol.*, in Lecture Notes in Computer Science, vol. 1057. Berlin, Germany: Springer, 1996, pp. 18–32.
[28] J. C. Shafer, R. Agrawal, and M. Mehta, "SPRINT: A scalable parallel classi er for data mining," in *Proc. VLDB*, 1996, pp. 544–555.
[29] B. Chandra, R. Kothari, and P. Paul, "A new node splitting measure for decision tree construction," *Pattern Recognit.*, vol. 43, no. 8, pp. 2725–2731, 2010.
[30] C. J. Mantas and J. Abellán, "Analysis and extension of decision trees based on imprecise probabilities: Application on noisy data," *Expert Syst. Appl.*, vol. 41, pp. 2514–2525, Apr. 2014.
[31] R. Wang, S. Kwong, X. Z. Wang, and Q. Jiang, "Segment based decision tree induction with continuous valued attributes," *IEEE Trans. Cybern.*, vol. 45, no. 7, pp. 1262–1275, Jul. 2015.
[32] B. Ghattas, P. Michel, and L. Boyer, "Clustering nominal data using unsupervised binary decision trees: Comparisons with the state of the art methods," *Pattern Recognit.*, vol. 67, pp. 177–185, Jul. 2017.
[33] V. Franc, O. Fikar, K. Bartos, and M. Sofka, "Learning data discretization via convex optimization," *Mach. Learn.*, no. 107, pp. 333–355, Feb. 2018.
[34] S. Garcia, J. Luengo, J. A. Sáez, V. Lopez, and F. Herrera, "A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 4, pp. 734–750, Apr. 2013.
[35] S. Ramírez-Gallego, S. García, J. M. Benítez, and F. Herrera, "Multivariate discretization based on evolutionary cut points selection for classification," *IEEE Trans. Cybern.*, vol. 46, no. 3, pp. 595–608, Mar. 2016.

[36] D. Yan, D. Liu, and Y. Sang, "A new approach for discretizing continuous attributes in learning systems," *Neurocomputing*, vol. 133, pp. 507–511, Jun. 2014.
[37] U. M. Fayyad and K. B. Irani, "On the handling of continuous-valued attributes in decision tree generation," *Mach. Learn.*, vol. 8, no. 1, pp. 87–102, 1992.
[38] U. Fayyad and K. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," in *Proc. Int. Joint Conf. Artif. Intell.*, 1993, pp. 1022–1027.
[39] T. Elomaa and J. Rousu, "General and efficient multisplitting of numerical attributes," *Mach. Learn.*, vol. 36, no. 3, pp. 201–244, 1999.
[40] K. Shehzad, "EDISC: A class-tailored discretization technique for rule-based classification," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 8, pp. 1435–1447, Aug. 2012.
[41] Y. Sang, H. Qi, K. Li, Y. Jin, D. Yan, and S. Gao, "An effective discretization method for disposing high-dimensional data," *Inf. Sci.*, vol. 270, pp. 73–91, Jun. 2014.
[42] L. Ma, S. Destercke, and Y. Wang, "Online active learning of decision trees with evidential data," *Pattern Recognit.*, vol. 52, pp. 33–45, Apr. 2016.
[43] K. S. Hong, M. P.-L. Ooi, and Y. C. Kuang, "Sparse alternating decision tree," *Pattern Recognit. Lett.*, vols. 60–61, pp. 57–64, Aug. 2015.
[44] Y. Qian, H. Xu, J. Liang, B. Liu, and J. Wang, "Fusing monotonic decision trees," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 10, pp. 2717–2728, Oct. 2015.
[45] W. Zalewski, F. Silva, A. G. Maletzke, and C. A. Ferrero, "Exploring shapelet transformation for time series classification in decision trees," *Knowl-Based Syst.*, vol. 112, pp. 80–91, Nov. 2016.
[46] Y. Xia, C. Liu, Y. Li, and N. Liu, "A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring," *Expert Syst. Appl.*, vol. 78, pp. 225–241, Jul. 2017.

JIANJIAN YAN received the B.S. degree in computer science and technology from Jinggangshan University, in 2008, and the M.S. degree from Central South University, in 2012. He is currently pursuing the Ph.D. degree with the School of Software, Xiamen University, Xiamen, China. His current research interests include data mining, computer vision, and deep learning techniques.

ZHONGNAN ZHANG (M'15) received the B.E. and M.E. degrees in computer science and technology from Southeast University, Nanjing, China, in 1999 and 2001, respectively, and the Ph.D. degree in computer science from The University of Texas at Dallas, TX, USA, in 2008.

He was an Assistant Professor, from 2009 to 2012, and an Associate Professor, from 2012 to 2017, with the Software School, Xiamen University, Xiamen, China, where he has been a Full Professor, since 2017. His research interests include big data analysis, data mining, machine learning, and bioinformatics. He is an Editor of *PLoS One*.

LINGWEI XIE received the B.S. degree in computer science and technology from Zhejiang University City College, in 2013. He is currently pursuing the Ph.D. degree with the Software School, Xiamen University, Xiamen, China. His current research interests include machine learning, data mining, and bioinformatics.

ZHANTU ZHU received the B.S. degree in software engineering from Xiamen University, in 2012. In 2018, he joined the Tencent AI P.D., as a Summer Intern. He is currently pursuing the M.S. degree in computer science and technology with the School of Software, Xiamen University. His current research interests include CNN acceleration, object detection, and 3D pose estimation.