Journal of Network Intelligence ©2018 ISSN 2414-8105 (Online)
Taiwan Ubiquitous Information Volume **3**, Number **1**, February **2018**

A Review of Handling Continuous and Unknown Attribute Values of C4.5 Algorithm

Mao-Hsiung Hung*, Qian Chen, Yi Chen

College of Information Science and Engineering, Fujian University of Technology
Fujian Provincial Key Laboratory of Big Data Mining and Applications,
Fujian University of Technology
No.33, Xueyuan Road, University Town, Minhou, Fuzhou City, 350118, China
*Corresponding author:mhhung@fjut.edu.cn
850342095@qq.com, 914225761@qq.com

Received October 2017; revised November 2017

Abstract. *C4.5 is one of most classic algorithms for creating decision tree. The important improvement of handling continuous and unknown attribute values achieves a larger generalization for C4.5 algorithm. This paper presents a mathematical and systematic review to describe basic algorithm and the two handles. We input datasets of few samples for evaluation and the experimental results indicated that C4.5 algorithm performs good rule conclusion for datasets. Deeper studies and more applications for C4.5 algorithm will be contributed by this paper.*

Keywords: C4.5 algorithm, continuous attribute value, unknown attribute value

1. Introduction.

In past three decades, decision tree algorithm have received many attentions in research communities devoting machine learning. Due to its high popularity, most of machine learning textbooks often spend a whole of chapter to introduce the algorithm and its modified versions [1]. Among these modified versions of decision tree algorithm, C4.5 is acknowledged one of most classic and excellent algorithms by many researchers.

In 1979, J. R. Quuinlan proposed the first versions of decision tree algorithm called by ID3 [2]. In the next few years, he continued to propose several modified versions based on ID3, such as C4.0, C4.5 and C5.0. In C4.5 algorithm, J. R. Quuinlan proposed several schemes to solve some applicable problems for a various of datasets, such as continuous and unknown attribute value [3]. These improvements enhanced decision tree algorithm's generalization and its wide range of applications is guaranteed.

Continuous and unknown attribute values are often in many data acquisition. Because information gain computation depends on discrete values, the discretization of continuous attribute value is employed to compute information gain [4]-[6]. Unknown attribute value, also as known missing value, means partial attribute values of a sample are not available or missing due to incomplete data acquisition [7]-[8]. Although these samples of incomplete attribute values may not input to training algorithm, the way still causes to loss a amounts of available values. C4.5 algorithm also proposed a scheme to compute information gain for samples of incomplete attribute values. Although C4.5 proposed methods to handle continuous and unknown attribute values, it is still a shortage of a mathematical and systematic statement, even if C4.5 program has been released for many years. Therefore, this paper presents a mathematical and systematic review to describe handling continuous and unknown attribute values for of algorithm. The review will increase deeper understanding to C4.5 algorithm for readers and guide more researchers to suitably apply C4.5 algorithm.

The remainder of this paper is organized as follows. Section 2 describes the proposed methods. Section 3 demonstrates and discusses experimental results of the proposed methods. The conclusions are drawn in Section 4.

2. **Proposed Method.**

2.1. **Basic algorithm.** The basic algorithm of decision tree learning is described as follow. We input a training set,D, which contains m samples' feature vectors, $x_1, x_2, ..., x_m$, and their labeled classes, $y_1, y_2, ..., y_m$. As input, an attribute set,A, is composed by d attributes, $a_1, a_2, ..., a_d$. The algorithm is recursive, and a decision tree is generated for classification when the function call of most upper level returns .

```
Input: Training set, $D = \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\}$; attribute set, $A = \{a_1, a_2, ..., a_d\}$
function GenerateTree($D, A$)
generate a new node
if all samples in $D$ are belonging to a same class of $C$, then
  let the node to be a leaf node which is labeled by $C$ class; return
end if
if $A = \emptyset$ or all samples in $D$ have the same values in $A$ then
  let the node to be a leaf node and label its class outcome which the most samples are
  belonging to; return
end if
select the best split attribute $a_*$ from $A$
for each value $a_*^v$ in $a_*$ do
  generate a branch for the value $a_*^v$ in the node
  Let $D^v$ to be a subset of the same value of $a_*^v$ in $a_*$ extracted from $D$
  if $D^v = \emptyset$ then
    let the branch to be a leaf node and label its class outcome which the most samples
    are belonging to; return
  else
    call GenerateTree($D^v$, $A \setminus \{a_*\}$) to generate a new node for the branch
  end if
end for
```

The first step of the learning algorithm is to generate a new node. After that, the first return condition is that when all samples in D are belonging to a same class of C, that means no more any branch is required in further levels. Therefore, the classification outcome of the new node is labeled by C class. Similarly, the second return condition is that when the attribute set of A is deleted to an empty set or when all samples in D have the same attribute values in A, that also means no more any branch is required in further levels. Therefore, the class outcome of the new node is labeled by a class which the most samples are belonging to.

After that, the algorithm selects the best split attribute for the new node. The selection criterion uses maximum of information gain. An entropy of D is first computed by Eq.(1) where p_k means the probability of k-th class and $|Y|$ is the number of classes. Then, Eq.(2) is used to compute an information gain of an attribute of a in D. Assuming a contains V values, $a^1, a^2, ..., a^V$, D^v presents the data set having a^v in a. Calling Eq.(1) to compute $Ent(D)$ and $Ent(D^v)$, $Gain(D,a)$ is obtained by the $Ent(D)$ subtracting the summation of $Ent(Dv)$ multiplying the ratio of D^v to D for all a^v. After all computation of $Gain(D,a)$ for all a, the best split attribute of a_* is determined by the argmax operation as Eq.(3).

$$\begin{aligned} Ent(t) &= -\sum_{k=1}^{|Y|} p_k \log_2 p_k & (1)\\ Gain(D,a) &= Ent(D) - \sum_{v=1}^{V} \frac{|D^v|}{|D|} Ent(D^v) & (2)\\ a_* &= \arg\max_{x \in A} Gain(D,a) & (3) \end{aligned}$$

After the attribute selection, we generate a branch for each value of a_*^v in a_*. Subsequently, a subset of D^v is extracted from D according to a_*^v. Considering D^v, if D^v is empty, the branch would be a leaf node and label its class outcome which the most samples of D^v are belonging to. Otherwise, the next function call is required for a nonempty set of D^v and the a_* is deleted form A.

In the testing phrase, a feature vector inputs to the decision tree and a tree travel starts at the root node. When the travel reaches at a non-leaf node, one of attributes is selected and the next walk goes to a branch according to its attribute values. Until the travel reaches a leaf node, the classification outcome can be determined as the leaf's labeled class.

2.2. **Handling Continuous Attribute Value.** As mentioned in the above subsection, we introduced the basic algorithm of decision tree generation which just handles the discrete attribute values. For continuous attribute values, C4.5 algorithm uses a bi-partition method to discretize continuous values. The cooperation of the discretization technique makes that the basic algorithm of discrete version has ability to handle continuous values [4]-[6].

Given a dataset of D and a continuous attribute of a, there are n different and ascending ordered values in a and D, denoted by $a_1, a_2, ..., a_n$. Based a threshold of t, D can be partitioned into two subsets of $D^+(t)$ and $D^-(t)$ which respectively contain values less than t and values greater than t. We simply let the middle values of the two consecutive value of $a_1, a_2, ..., a_n$ to be the threshold candidate as T_a in Eq.(4).

$$T_a = \left\{ \frac{a^i + a^{i+1}}{2}, \text{ for } i = 1, 2, ..., n-1 \right\} \qquad (4)$$

In Eq.(5), we denote the information gain of a partition threshold of t in a as $Gain(D,a,t)$. The information gain of $Gain(D,a)$ is equal to the maximum of $Gain(D,a,t)$ for all threshold candidates in T_a. As similar as Eq.(2), $Gain(D,a,t)$ can be equal to a information gain of two subsets of $D^+(t)$ and $D^-(t)$ partitioned from D.

$$\begin{aligned} Gain(D,a) &= \max_{t \in T_a} Gain(D,a,t) \\ &= \max_{t \in T_a} \left[Ent(D) - \sum_{\lambda \in \{-,+\}} \frac{|D^\lambda(t)|}{|D|} Ent(D^\lambda(t)) \right] & (5) \end{aligned}$$

TABLE 1. Six discrete attributes and their values

Attribute	color	stem (shape)	knock (sound)	texture	navel (shape)	touch
#value	3	3	3	3	3	2
Value	dark-green	curled-up	voiced	clear	dented	hard-slippery
	jet-black	slightly-curled	dull	slightly-blurry	slightly-dented	soft-sticky
	shallow-white	stiff	crisp	blurry	flat	

2.3. **Handling Unknown Attribute Value.** Similarly, a scheme of handling unknown attribute values is required to cooperate the basic algorithm, when partial attribute values are not available in a dataset. Considering the unknown attribute values, two problems are required to solved. Problem I is how to select a best split attribute when unknown values happen. After a best split attribute is determined, Problem II is how to split a sample which contains an unknown value in the best split attribute.

Given a training set of D and an attribute of a, let $\tilde{D}$ represent a sample subset of no unknown value in a. For Problem I, we can evaluate a's quality according to $\tilde{D}$ only. Let $\tilde{D}^v$ represent a subset of a_v value in a and $\tilde{D}_k$ represent a subset belonging to k-th class. And, then we assign every samples a weight of w_x and $w_x = 1$. And, then define the following three factors related to w_x sums of D, $\tilde{D}$, $\tilde{D}_v$ and $\tilde{D}_k$ in Eq.(6)-(8). ρ means a proportion of w_x sums of no unknown value, $\tilde{r_v}$ means a proportion of w_x sums of no unknown a_v value and $\tilde{p_k}$ means a proportion of w_x sums of no unknown value belong to k-th class.

$$\rho = \frac{\sum_{x \in \tilde{D}} w_x}{\sum_{x \in D} w_x} \tag{6}$$

$$\tilde{r_v} = \frac{\sum_{x \in \tilde{D}^v} w_x}{\sum_{x \in \tilde{D}} w_x}, \text{ for } v = 1, 2, ..., V \tag{7}$$

$$\tilde{p_k} = \frac{\sum_{x \in \tilde{D}_k} w_x}{\sum_{x \in D} w_x}, \text{ for } k = 1, 2, ..., |Y| \tag{8}$$

TABLE 2. Melon dataset I

No	color	stem	knock	texture	navel	touch	good/bad
1	dark-green	curled-up	voiced	clear	dented	hard-slippery	good
2	jet-black	curled-up	dull	clear	dented	hard-slippery	good
3	jet-black	curled-up	voiced	clear	dented	hard-slippery	good
4	dark-green	curled-up	dull	clear	dented	hard-slippery	good
5	shallow-white	curled-up	voiced	clear	dented	hard-slippery	good
6	dark-green	slightly-curled	voiced	clear	slightly-dented	soft-sticky	good
7	jet-black	slightly-curled	voiced	slightly-blurry	slightly-dented	soft-sticky	good
8	jet-black	slightly-curled	voiced	clear	slightly-dented	hard-slippery	good
9	jet-black	slightly-curled	dull	slightly-blurry	slightly-dented	hard-slippery	bad
10	dark-green	stiff	crisp	clear	flat	soft-sticky	bad
11	shallow-white	stiff	crisp	blurry	flat	hard-slippery	bad
12	shallow-white	curled-up	voiced	blurry	flat	soft-sticky	bad
13	dark-green	slightly-curled	voiced	slightly-blurry	dented	hard-slippery	bad
14	shallow-white	slightly-curled	dull	slightly-blurry	dented	hard-slippery	bad
15	jet-black	slightly-curled	voiced	clear	slightly-dented	soft-sticky	bad
16	shallow-white	curled-up	voiced	blurry	flat	hard-slippery	bad
17	dark-green	curled-up	dull	slightly-blurry	slightly-dented	hard-slippery	bad

Based on the above factors, we can compute the information gain of D in a using Eq.(9). $Gain(D,a)$ is equal to ρ multiplying $Gain(\tilde{D},a)$ and Eq.(2) is applied to compute $Gain(\tilde{D},a)$ using the substitutions of $Ent(\tilde{D})$, $Ent(\tilde{D}_v)$ and $\tilde{r}_v$. In Eq.(10), Eq.(1) is applied to compute $Ent(\tilde{D})$ using the substitutions of $\tilde{p}_k$.

$$\begin{aligned} Gain(D,a) &= \rho \times Gain(\tilde{D},a) \\ &= \rho \times \left[Ent(\tilde{D}) - \sum_{v=1}^{V} \tilde{r}_v Ent(\tilde{D}^v) \right] \end{aligned} \tag{9}$$

$$Ent(\tilde{D}) = -\sum_{k=1}^{|Y|} \tilde{p}_k \log_2 \tilde{p}_k \tag{10}$$

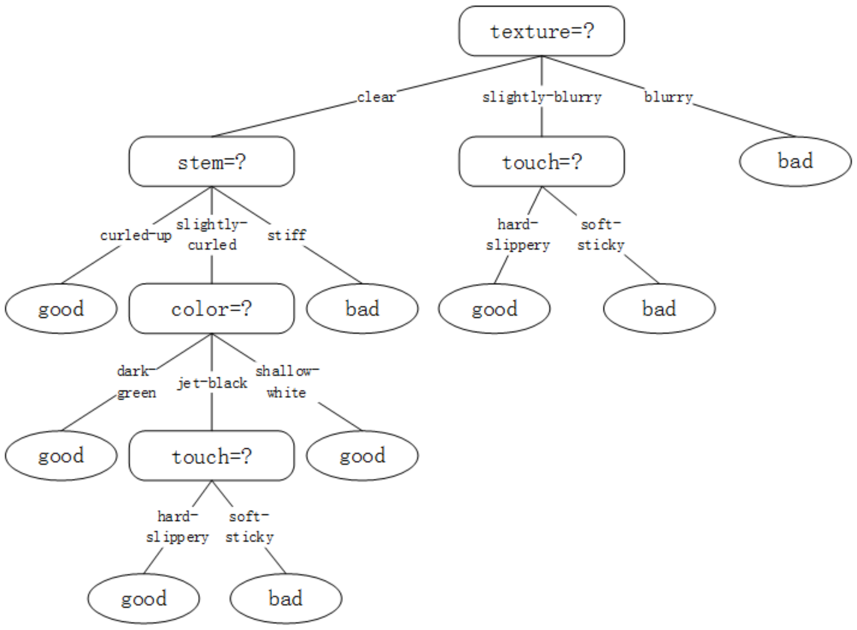

FIGURE 1. Experimental result for basic algorithm

If a value of a_v in a best split attribute is known, the sample will walk to v-th branch and the weight of w_x will keep. For Problem II, if a value of a_v in a best split attribute is unknown, the sample will split into V branches. The weight for the split sample in v-th branch is equal to $\tilde{r}_v \times w_x$ for $v = 1, 2, ..., V$.

In testing phase, if a sample splits into several branches due to unknown value, the situation might cause multiple classification results when the sample's travel of the decision tree reaches two more leaf nodes. The final classification result of the sample depends on these split weights in the leaf nodes. One of most common determination is that these split weights can be regarded as probabilities and these probabilities can form a probability distribution, so that a random process based on the probability distribution is used to decided the final classification result.

3. **Experimental result.** To evaluate C4.5 algorithm, we prepare a training dataset which collects 17 samples and their attribute values for melon classification. The attributes are composed of six discrete attributes and two continuous attributes. The six discrete attributes are about melon's color, stem (shape), knock (sound), texture, navel

(shape) and touch. The two continuous attributes contain melon's density and sugar content. These samples are labeled good or bad melon. The purpose of learning algorithm is to build a decision tree to classify training samples into good or bad labels. Table 1 lists the six discrete attributes and their attribute values. The following subsections describe experimental results for basic algorithm, handling continuous attribute value and handling unknown attribute value.

3.1. **Experimental result for basic algorithm.** We input Melon dataset I containing six discrete attributes as listed in Table 2, and use basic algorithm to generate a decision tree as shown in Fig. 1. We found that four attributes such as texture, stem, color and touch, are selected to be best split attributes during the training phase. The decision tree are concluded out two classification rules. One is that while a sample's texture is clear, the sample is classified into good melon except for two attribute values of stiff stem and soft-sticky touch. The other one is that while a sample's texture is not clear, the sample is classified into bad melon except for soft-sticky touch. It is note that touch attribute is selected twice, but touch attribute is just applied once in one of travelling paths from root to left nodes. That still does not violate a principle of one attribute only applied once during testing phase.

Table 3. Melon dataset II

No	density	sugar content	good/bad
1	0.697	0.46	good
2	0.774	0.376	good
3	0.634	0.264	good
4	0.608	0.318	good
5	0.556	0.215	good
6	0.403	0.237	good
7	0.481	0.149	good
8	0.437	0.211	good
9	0.666	0.091	bad
10	0.243	0.267	bad
11	0.245	0.057	bad
12	0.343	0.099	bad
13	0.639	0.161	bad
14	0.657	0.198	bad
15	0.36	0.37	bad
16	0.593	0.042	bad
17	0.719	0.103	bad

3.2. **Experimental result for handling continuous attribute value.** We input Melon dataset II containing two continuous attributes of density and sugar content, as listed in Table 2, and the basic algorithm combine continuous value handling to generate a decision tree as shown in Fig. 2. The decision tree are also concluded out three classification rules. One is that while a sample's sugar content< 0.126 or it's density< 0.381, the sample is classified into bad melon. The second one is that while a sample's sugar content> 0.205, the sample is classified into good melon. The third one is that while a sample of sugar content between 0.126 and 0.205 and its density< 0.56, the sample is classified into good melon, but its $density > 0.56$ to bad melon. It is note that the two attribute are both selected twice, and the two attributes are possibly applied twice in several travelling paths from root to left nodes. These twice applications in the same attribute are but under different partition thresholds, therefore the decision tree generation is still reasonable.

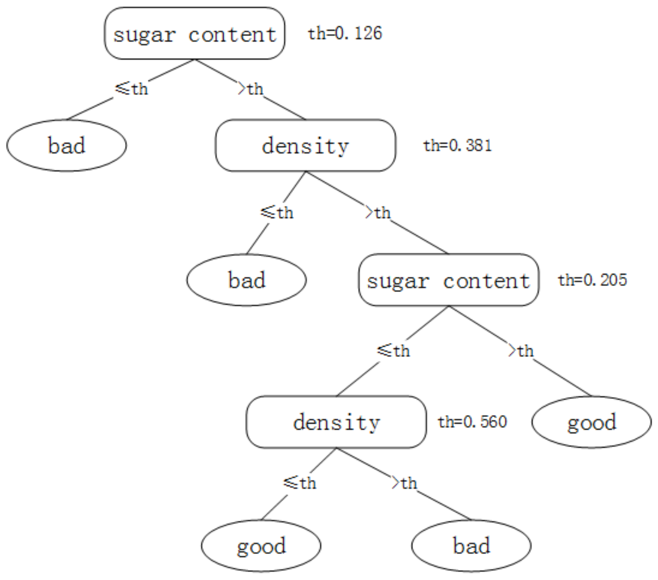

Figure 2. Experimental result for handling continuous attribute value

3.3. **Experimental result for handling unknown attribute value.** We input Melon dataset III containing six discrete attributes and several unknown attribute values as listed in Table 4, and the basic algorithm combines unknown value handling to generate a decision tree as shown in Fig. 3. The numbers in blue mean the weighting factors, $\tilde{r_v}$ in Eq.(7), for every branches. We found that all six attributes are selected to be best split attributes during the training phase. The decision tree are concluded out three classification rules. One is that while a sample's texture is clear, the sample is classified into good melon except for soft-sticky touch. The second one is that while a sample's texture is slightly-blurry, the sample is classified into bad melon except for slightly-dented and flat navels. The third one is that while a sample's texture is blurry, the sample is classified into bad melon except for jet-black color.

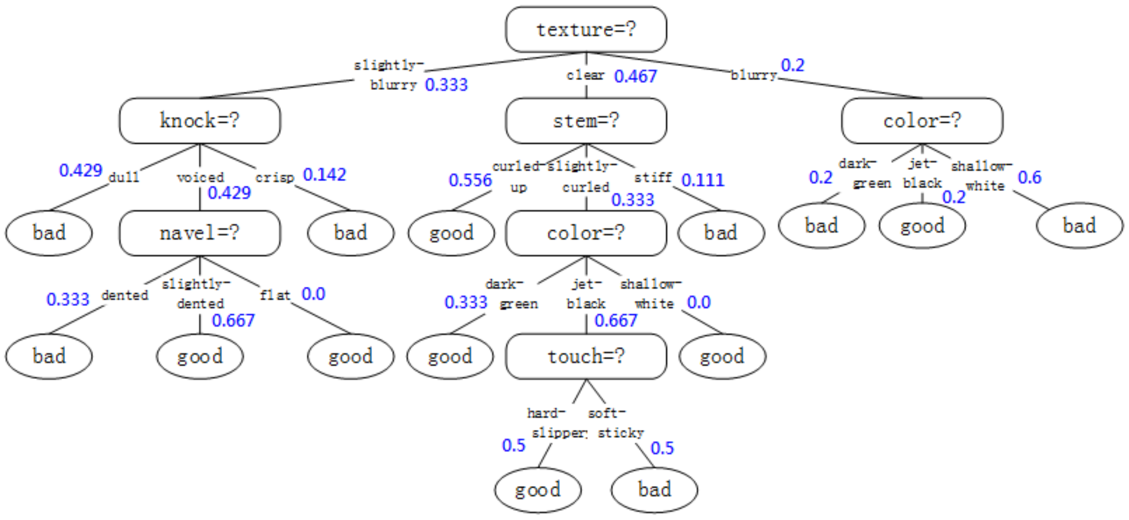

Figure 3. Experimental result for handling unknown attribute value

TABLE 4. Melon dataset III

No	color	stem	knock	texture	navel	touch	good/bad
1	–	curled-up	voiced	clear	dented	hard-slippery	good
2	jet-black	curled-up	dull	clear	dented	hard-slippery	good
3	jet-black	curled-up	–	clear	dented	hard-slippery	good
4	dark-green	curled-up	dull	clear	dented	hard-slippery	good
5	–	curled-up	voiced	clear	–	hard-slippery	good
6	dark-green	slightly-curled	voiced	clear	slightly-dented	soft-sticky	good
7	jet-black	slightly-curled	voiced	slightly-blurry	slightly-dented	soft-sticky	good
8	jet-black	slightly-curled	voiced	–	slightly-dented	hard-slippery	good
9	jet-black	–	dull	slightly-blurry	slightly-dented	hard-slippery	bad
10	dark-green	stiff	crisp	–	flat	soft-sticky	bad
11	shallow-white	stiff	crisp	blurry	flat	–	bad
12	shallow-white	curled-up	–	blurry	flat	soft-sticky	bad
13	–	slightly-curled	voiced	slightly-blurry	dented	hard-slippery	bad
14	shallow-white	slightly-curled	dull	slightly-blurry	dented	hard-slippery	bad
15	jet-black	slightly-curled	voiced	clear	–	soft-sticky	bad
16	shallow-white	curled-up	voiced	blurry	flat	hard-slippery	bad
17	dark-green	–	dull	slightly-blurry	slightly-dented	hard-slippery	bad

4. **Conclusions.** This paper has presented a review of handling continuous and unknown attribute values of C4.5 algorithm. Continuous value discretization and sample weighting are respectively applied to handle continuous and unknown attribute values. The two handles cooperate with the basic algorithm, so that the generalization of C4.5 receives a large extension to a various of datasets. In this paper, a mathematical and systematic description is proposed and meanwhile the experimental results of a melon dataset are demonstrated good rule conclusion of C4.5 algorithm. The review will contribute to the deeper studies and more understanding for C4.5 algorithm.

Acknowledgment. This work was supported in part by Fujian Provincial Department of Science and Technology, Granted No.2017J01729 and by Fujian University of Technology, Granted No.GY-Z15087.

REFERENCES

[1] Z. H. Zhou, *Machine Learning*, Tsinghua University Press, Beijing, Jan, 2016.
[2] J. R. Quinlan, *Induction of decision trees*, Machine Learning, vol.1, no.1, pp.81–106, 1986.
[3] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, CA, Jan, 1992.
[4] U. M. Fayyad, K. B. Irani. *On the handling of continuous-valued attributes in decision tree generation*", Machine Learning, vol.8, no.1, pp.87–102, 1992.
[5] D. Kim, J. Lee. *Handling continuous-valued attributes in decision tree using neural network modeling*, Lecture Notes in Computer Science vol. 1810, pp.211–219, 2000.
[6] J. Yang, C. Ye, P. Recog. *Tree Based on Fuzzy Discretization*, Computer Simulation, 2000.
[7] Y. F. Qiu, X. Y. Zhang, X. Li, L. S. Shao *Research on the missing attribute value data-oriented for decision tree*, Computer Science, vol.38, no.10, pp.174–176, 2010.
[8] S. Garca, J. Luengo, F. Herrera *Dealing with Missing Values. Data Preprocessing in Data Mining*, Springer International Publishing, pp.59–105, 2015.