Computers and Electrical Engineering 65 (2018) 474–487

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

An improved Id3 algorithm for medical data classification☆

Shuo Yang, Jing-Zhi Guo*, Jun-Wei Jin

Department of Computer and Information Science, Faculty of Science and Technology, University of Macau, Macao

ARTICLE INFO

Article history:
Received 15 November 2016
Revised 8 August 2017
Accepted 8 August 2017
Available online 12 August 2017

Keywords:
Id3 algorithm
Decision tree
Balance function
Numeric attribute discretization
Rule representation of classifier model

ABSTRACT

Data mining techniques play an important role in clinical decision making, which provides physicians with accurate, reliable and quick predictions through building different models. This paper presents an improved classification approach for the prediction of diseases based on the classical Iterative Dichotomiser 3 (Id3) algorithm. The improved Id3 algorithm overcomes multi-value bias problem when selecting test/split attributes, solves the issue of numeric attribute discretization and stores the classifier model in the form of rules by using a heuristic strategy for easy understanding and memory savings. Experiment results show that the improved Id3 algorithm is superior to the current four classification algorithms (J48, Decision Stump, Random Tree and classical Id3) in terms of accuracy, stability and minor error rate.

1. Introduction

In the medical and health fields, researchers have been trying different data mining techniques in an attempt to improve the precision of medical diagnosis. Methods with better precision and reliability would provide more supportive data for the identification of prospective patients via accurate disease prediction. Until now, techniques such as support vector machines [1–3], neural networks [4–6], logistic regression [7,8] and clustering algorithms [9,10] have been applied in medical field. Experiment results show that these approaches give high accuracy in prediction [11,12]. These techniques support physicians to make accurate diagnostic decisions in clinical diagnosis. However, among them, the decision tree as a kind of classification method is still preferred in engineering applications, since tree models as classifiers can be easily applied. Moreover, their usages in bio-informatics have been reported in numerous research papers [2,5,7]. For example, based on historical patient data, decision tree classifiers are used to predict whether a patient has a kind of disease (e.g., heart diseases, lung cancers or breast cancers). The internal nodes of a decision tree are test attributes including properties such as symptoms and signs of patients. Based on the decision tree, an illness can be predicted by passing a new example down from the root to a leaf, testing the appropriate attribute at each internal node and following the branch corresponding to the attribute's value. If the accurate rate is high, then the patient can be suggested to receive proper treatments as early as possible.

Iterative Dichotomiser 3 (Id3) is an important classification algorithm which was proposed by Quinlan [13] in 1986 and has been applied in fields such as economics, medicine and science. In a situation where large data sets are used and the information for classification is complex, Id3 provides a useful solution. However, it has problems including multi-value bias, only handling nominal attributes as well as loosing relations among attributes in the classifier model. To resolve these problems, many approaches have been proposed, such as Gain ratio [13–15] and Gini index [16–18] for multi-value bias,

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Guest Editor Dr. S. Sioutas.

* Corresponding author.
E-mail addresses: yb37416@umac.mo (S. Yang), jzguo@umac.mo (J.-Z. Guo), jinjunwei24@163.com (J.-W. Jin).

http://dx.doi.org/10.1016/j.compeleceng.2017.08.005
0045-7906/

partition number [19] for numerical attribute discretization. However, these approaches have both pros and cons. Gain ratio tends to prefer unbalanced splits in which one partition is much small than the other. Gini index is biased towards multi-value attributes. It also has difficulties when the number of classes is large and tends to favor tests that result in equal-sized partitions and purity in both partitions. For numeric attribute discretization, pre-assigning partition number is a common way [19]. However, its drawback is that it may be difficult to choose an optimal partition number for any numeric attribute or the threshold for an interval partition. More often, when the number of numeric attributes is large, it is time-consuming to set an optimal partition number for each of them.

This paper proposes an improved Id3 algorithm. The new algorithm uses a balance function to offset the increase of the information gain when an attribute has many different values. Besides, it is capable to handle numeric attributes by using a novel discretization method. The final classifier model is represented in the form of rules rather than a tree structure for the sake of easy interpretation and memory saving. Through the experiments with five data sets from UCI [20], the improved algorithm achieves better accuracy and reliability compared to other four decision-tree classification algorithms. Therefore, it can be used to improve the performance of disease prediction. The main contributions of this paper are:

- This paper proposes an improved Id3 algorithm for disease prediction with a novel balance function for test attribute selection, a novel numeric attribute discretization strategy as well as a new rule-based heuristic method for classifier representation.
- This paper mathematically proves that the classical Id3 algorithm is biased to multi-variated attributes and how the proposed balance function can address this issue.

The rest of the paper is organized as follows: Section 2 discusses related works. In Section 3, the fundamental theory of Id3 algorithm and its problems are discussed. Section 4 proposes improvement strategies. Section 5 shows experimental results based on five benchmark data sets. Finally, a conclusion is given with summary, contributions and future work.

2. Related work

Decision trees (DT) are becoming popular with the growth of data mining in the field of bioinformatics. DT-based classification algorithms have tree structures consisting of internal nodes, branches and leaves. The tree structure is equivalent to a set of decision rules. Each internal node in the tree represents a decision on a property (or an attribute), and the branch of each internal node represents the output of a decision result. The purpose of DT is to search for a set of decision rules to predict an outcome for a set of input instances. A leaf node of the tree represents a sample group or a sample classification. The number of nodes of the tree has an important influence on the accuracy of the classification and the size of the tree. Common decision tree construction methods include classification and regression trees [21,22], chi-square automatic interaction detector (CHAID [23]), Random Forest and Random Tree [24]. Compared to Boosted Decision Tree (BDT) methods and Decision Tree Forest (DTF) techniques, Id3 as a single decision tree classification algorithm provides a rapid and effective method to categorize data sets.

To resolve the issue of multi-variated attribute and select the most discriminatory attribute as splitting node, several methods have been proposed. Gain ratio [13–15] and Gini index [16–18] are two famous indices designed to address this problem. By computing these indices, the fitness of an attribute can be determined, and the attribute with the best fitness is chosen as the test attribute for the current node when building a decision tree. Specifically, Gain ratio normalizes information gain through dividing information gain by splitting information amount that represents the potential information generated by splitting the training data set D into v partitions. The strategy of Gini index chooses the attribute whose Gini Index is minimum after splitting. Besides, some researches [19,25] focus on multi-valued and multi-labeled data where the traditional decision tree algorithms have been proved to be not applicable. However, the issue of multi-value attribute is a special case of multi-variated attribute, since multiple values of a certain attribute in a sample can be seen as a special attribute value. Since multi-value attributes have different combination forms of values, this kind of attribute has high information gain and thus it is easily chosen to be test/split attribute.

To address the issue of numeric attribute discretization, some studies (e.g., C4.5 [15]) assign attribute A with two possible outcomes: $A \leq t$ and $A > t$, where t is called threshold. The threshold is calculated by means of a linear search in the whole training set. Thus, it needs to compute information gain many times with the change of t. Some researchers propose improved methods to update the procedure of numeric attribute discretization. For example, in [19], a partition number needs to be pre-assigned, and then the sorted data records are segmented proportionally to the partition number. The problem is that it may be difficult for users to set an optimal partition number for any numeric attribute and the threshold for an interval partition.

3. Classical Id3 algorithm and its problems

The core part of the Id3 algorithm is that it computes the information gain as a selection criteria of test attributes for hierarchical levels of non-leaf nodes in a decision tree. The aim of the algorithm is to acquire the largest class information about the sub-dataset when making a decision on an internal node. Specifically, its first step is to compute the information gain for each attribute and select the one that holds the largest information gain as the root node of the decision tree. Based on the different values of the attribute, the node generates different branches. Then, for the sub-dataset assigned to each

branch, the algorithm iteratively calls information computing function to create new sub-nodes until the sub-dataset in each branch have the same class label. Finally, the built-up decision tree is used to classify new examples.

3.1. Basic principles of Id3

The Id3 algorithm chooses test attributes by calculating and comparing their information gains. Let S be the collection of data samples. Suppose class attribute C has m different values which represent m different class labels $C_i (i = 1, 2, \ldots, m)$. Let S_i be the number of samples in class $C_i (i = 1, 2, \ldots, m)$. The expected information amount required to classify S is given by (1):

$$I(S_1, S_2, \ldots, S_m) = -\sum_{i=1}^{m} p_i \log_2 p_i \tag{1}$$

where p_i means the probability of instances in S belonging to class C_i. $I(S_1, S_2, \ldots, S_m)$ means the average information amount required to identify the class labels for all instances in S.

Assume attribute A has v different values $\{a_1, a_2, \ldots, a_v\}$ in the training dataset S. If A is a nominal attribute, then the attribute splits S into v subsets such that $\{S_1, S_2, \ldots, S_v\}$, in which S_j is a subset of S where instances in S_j have the same attribute value a_j on A. However, instances in S_j may have different class labels. Let S_{ij} be the set of instances whose class labels are C_i in the subset of $\{S_j | A = a_j, j \in 1, 2, \ldots, v, S_j \in S\}$ where attribute $A = a_j$. The required information amount (i.e., entropy) of attribute A to split the training dataset S is measured by (2):

$$E(A) = \sum_{j=1}^{v} \left(\frac{(s_{1j} + s_{2j} + \ldots + s_{mj})}{s} \times I(s_{1j}, s_{2j}, \ldots, s_{mj}) \right) \tag{2}$$

The less information amount required, the more purity of a sub-dataset is.

$$I(s_{1j}, s_{2j}, \ldots, s_{mj}) = -\sum_{i=1}^{m} p_{ij} \log_2 (p_{ij}) \tag{3}$$

where p_{ij} means the probability of instances in S_j belonging to class C_i. $I(s_{1j}, s_{2j}, \ldots, s_{mj})$ means the average information amount required to identify the class labels for all instances in S_j. The information gain of A is defined as:

$$InfoGain(A) = I(S_1, S_2, \ldots, S_m) - E(A) \tag{4}$$

namely, the amount of original information requirement (only based on class) minus the amount of new information requirement (based the splitting on attribute A). Choosing the attribute with the maximum $InfoGain(A)$ as a test attribute which is assigned to an internal node in a decision tree. In this way, the needed information amount for classifying instances is the minimum.

3.2. Problems of classical Id3 algorithm

The classical Id3 algorithm has problems including bias on multi-variated attributes, helplessness with numeric attributes and loosing relations among attributes. In the following sections, these problems are discussed in detail.

3.2.1. Bias on multi-variated attributes

Id3 algorithm selects test attributes based on the information theory [13] which tends to assign those that have more different attribute values with larger information gains. However, in fact, this kind of attribute is not always the best one to distinguish training datasets. In other words, an attribute with the minimum entropy is not definitely the optimal for splitting datasets in practical applications. In the following, strict mathematical proof is given to validate whether classical Id3 algorithm has bias on multi-variated attributes.

Proof. First, assume A as a test attribute in a training dataset, its value is in the range of $\{a_1, a_2, \ldots, a_n\}$. A' is another test attribute in the same training data set, whose value is in the range of $\{a_1', a_2', \ldots, a_n', a_{n+1}'\}$ and $a_i = a_i' (i = 1, 2, \ldots, n-1)$. The attribute value a_n of A is the combination of the attribute values a_n' and a_{n+1}' of A'. By this way, the two attribute A and A' are consistent at the priority level.

Then, calculate the information gain of the two attributes A and A'. If the information gain of A' is always greater than the information gain of A, i.e., $InfoGain(A) < InfoGain(A')$. Then, it is concluded that Id3 algorithm has multi-value bias problem. If the difference between the two information gains is uncertainty, then the above problem may not exist.

Because the two attributes A and A' are in the same training dataset S, the expected information amounts of them to classify S are the same. Thus, the problem of proving $InfoGain(A) < InfoGain(A')$ is transformed to prove whether this inequality $E(A) > E(A')$ is always satisfied.

Next, calculate the conditional entropies of the two attributes, respectively:

$$E(A) = -\sum_{j=1}^{n} P(a_j) \sum_{i=1}^{m} \left[P(C_i/a_j) \log_2 P(C_i/a_j) \right] \tag{5}$$

and

$$E(A') = -\sum_{j=1}^{n+1} P(a_j') \sum_{i=1}^{m} \left[P(C_i/a_j') \log_2 P(C_i/a_j') \right] \tag{6}$$

Let (5) minus (6), we have

$$\begin{aligned} E(A) - E(A') &= \sum_{j=1}^{n+1} P(a_j') \sum_{i=1}^{m} \left[P(C_i/a_j') \log_2 P(C_i/a_j') \right] \\ &- \sum_{j=1}^{n} P(a_j) \sum_{i=1}^{m} \left[P(C_i/a_j) \log_2 P(C_i/a_j) \right] \end{aligned} \tag{7}$$

Due to $a_i = a_i'(i = 1, 2, \ldots, n-1)$, (7) is further simplified to (8).

$$\begin{aligned} & E(A) - E(A') \\ &= \sum_{j=n}^{n+1} P(a_j') \sum_{i=1}^{m} [P(C_i/a_j') \log_2 P(C_i/a_j')] - P(a_n) \sum_{i=1}^{m} [P(C_i/a_n) \log_2 P(C_i/a_n)] \\ &= P(a_n') \sum_{i=1}^{m} [P(C_i/a_n') \log_2 P(C_i/a_n')] + P(a_{n+1}') \sum_{i=1}^{m} [P(C_i/a_{n+1}') \log_2 P(C_i/a_{n+1}')] \\ &- P(a_n) \sum_{i=1}^{m} [P(C_i/a_n) \log_2 P(C_i/a_n)] \end{aligned} \tag{8}$$

$\because 0 < P(a_n) \le 1$, divide by $P(a_n)$ at both sides of (8) and get the (9) as follows:

$$\begin{aligned} \frac{E(A) - E(A')}{P(a_n)} &= \frac{P(a_n')}{P(a_n)} \sum_{i=1}^{m} \left[P(C_i/a_n') \log_2 P(C_i/a_n') \right] + \frac{P(a_{n+1}')}{P(a_n)} \\ \sum_{i=1}^{m} [P(C_i/a_{n+1}') \log_2 P(C_i/a_{n+1}')] &- \sum_{i=1}^{m} [P(C_i/a_n) \log_2 P(C_i/a_n)] \end{aligned} \tag{9}$$

For simplicity, assume that the class attribute contains two different values, i.e., $m = 2$ and we introduce the following parameters:

$$\begin{aligned} & P(C_1/a_n) = x, \quad P(C_1/a_n') = t, \\ & P(C_1/a_{n+1}') = q, \ P(a_n')/P(a_n) = r \end{aligned}$$

Through the analysis, we can get the following relations:

$$\begin{aligned} & P(C_2/a_n) = 1 - x, \quad P(C_2/a_n') = 1 - t, \\ & P(C_2/a_{n+1}') = 1 - q, P(a_{n+1}')/P(a_n) = 1 - r \end{aligned}$$

Take the above parameters into (9), then (10) is obtained.

$$\begin{aligned} \frac{E(A) - E(A')}{P(a_n)} &= r[t \log_2 t + (1-t) \log_2 (1-t)] + (1-r) \\ & [q \log_2 q + (1-q) \log_2 (1-q)] - [x \log_2 x + (1-x) log_2 (1-x)] \end{aligned} \tag{10}$$

Take $f(x) = x \log_2 x + (1-x) \log_2 (1-x)$, then (10) is simplified to (11).

$$\frac{E(A) - E(A')}{P(a_n)} = r f(t) + (1-r) f(q) - f(x) \tag{11}$$

According to the calculation formula of the derivative, take the derivative of $f(x)$. One order and two order derivatives of $f(x)$ on the interval of $x \in (0, 1)$ are:

$$f'(x) = \log_2 x - \log_2 (1-x), \quad f''(x) = \frac{1}{ln2} \left(\frac{1}{x} + \frac{1}{1-x} \right)$$

Table 1
Problem of numeric attributes.

Dataset	No. of samples	RMAE (%) for training data	RMAE (%) for percentage split = 50%
Diabetes	384	0	66.95
Dermatology database	366	0	45.05
SPECTF heart data set	187	0	40.82

$\because x \in (0,1), \therefore 1/\ln 2 > 0, 1/x > 0, 1/(1-x) > 0, \therefore f(x)'' > 0$. Thus $f(x)$ is a convex function on $x \in (0, 1)$. Based on the properties of convex function and $x = rt + (1-r)q$, $f(x) = f(rt + (1-r)q) \le rf(t) + (1-r)f(q)$. $\therefore rf(t) + (1-r)f(q) - f(x) > 0$, $(0 < r, t, q < 1)$ and (11) is transformed to (12).

$$\frac{E(A) - E(A')}{P(a_n)} = \mathrm{rf(t)} + (1-\mathrm{r})\mathrm{f(q)} - \mathrm{f(x)} > 0 \tag{12}$$

Multiply $P(a_n)$ at both sides of (12), and $0 < P(a_n) < 1$, then $E(\mathrm{A}) - E(\mathrm{A}') > 0$. $\therefore E(\mathrm{A}) > E(\mathrm{A}')$ and $InfoGain(\mathrm{A}) < InfoGain(\mathrm{A}')$. When the range of i in $a_i = a_i'(i = 1, 2, \ldots, n-1)$ is continuously decreasing, it is easy to prove $InforGain(\mathrm{A}) < InfoGain(\mathrm{A}')$ by following the nearly the same proof procedure. When $i = 0$, it can be concluded that if attribute A takes n values while attribute A' takes totally different n' $(n' > n)$ values, $InforGain(A) < InfoGain(A')$ is satisfied. Thus, the classical Id3 algorithm has the multi-value bias problem. □

3.2.2. Helplessness with numeric attributes

Classical Id3 algorithm can only deal with nominal attributes, leaving numeric attributes unprocessable. Thus, for numeric values in the dataset, data pre-processing need to be done. However, if the default filter of *NumericToNominal* in WEKA [26] is used to discretize numeric attributes in the dataset into nominal attributes, the performance of Id3 algorithm degrades dramatically. Table 1 shows that classical Id3 algorithm has 0 root mean absolute error (RMAE) rate on training datasets, but increases to a large extend on test datasets, by using 10-fold cross validation as the test option. The datasets used in Table 1 come from the installment file of the software of Weka [26] and the UCI machine learning repository [20].

For numeric attributes, since not all the attribute values are discovered during the training phase of model construction (due to attributes with continuous values), unseen values in the test dataset may result in improper classification of the instances. In order to overcome this problem, the improved algorithm divides the continuous range of numeric values into discrete ranges according to the number of different values in the class attribute.

3.2.3. Loosing relations among attributes

The classical Id3 algorithm uses a tree structure to represent a classifier model. However, the hierarchical arrangement of nodes in a tree are not well suitable for the reading habit of human beings. During the construction of a decision tree by Id3 algorithm, each internal node contains only one attribute. Thus, it is not evident to reveal the relation among attributes except the parent-children and sibling relationships between them. Although attributes are linked to each other through branches, the relationship between them is still loose and opaque. Moreover, the tree representation of classifier models is neither an optimal method for storage nor a good way for interpretation.

Many of the existing data mining studies have focused on representing the discovered knowledge as a set of If-Then rules. However, one of the major problems is that they often produce too many rules, which makes manual inspection and analysis difficult [27]. This problem has been regarded as a major obstacle in practical applications of data mining techniques. It represents a major gap between the mining results and their understanding and using. This paper aims to bridge the gap through simplifying the mining results via a heuristic method. It follows that they can be easily analyzed and understood by the human user.

4. Improvements on Id3

To resolve the problems of the classical Id3 algorithm mentioned in Section 3, this section proposes three strategies including a balance function for multi-variated attributes, a new discretization method for numeric attributes and a new representation for classifier models.

4.1. Balance function

Due to the fact that Id3 favors attributes that have more different values as test attributes, classifier models may not meet practical demands. This section proposes the Balance Function (BF) which is a monotonically increasing function of the number of attribute values.

The selection of BF needs two requirements. First, the function must be a monotonically increasing function of the number of different values (e.g., n) of the attribute. Let $f(n)$ be a monotonically increasing function of n, then information gain formula is transformed to (13) in which the molecule is a monotonically increasing function of n. The denominator must

also be a monotonically increasing function. Thus, the function $f(n)$ serves a trade-off role. Moreover, the monotonic trend of a balance function $f(n)$ should be comparatively flat in the range of n, which guarantees that the balance function is not over-balancing. This paper takes $f(n) = |\log_2(\sqrt{n+1})\frac{\cos(3.5n-1.5)}{1.8}|,\ (n > 0)$ as a balance function.

$$\text{Gain}(A) = \text{InfoGain}(A)/f(n) \tag{13}$$

Balance function plays a similar functionality to *splitInfo* in gain ratio in C4.5. The formula for computing gain ratio is defined as:

$$GainRatio(A) = \frac{Gain(A)}{splitInfo_A(D)} \tag{14}$$

$$splitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{D} \log_2 \frac{|D_j|}{D} \tag{15}$$

where *splitInfo* means how much information are needed to tell which branch an instance belongs to. In other words, the split information (*splitInfo*) value represents the potential information generated by splitting the training data set into v partitions, corresponding to v values of attribute A. High *splitInfo* means that a high amount of information is needed to know which partition an instance belongs to. This is because all partitions have almost the same number of samples, which means v is relatively large and the attribute is multi-variated. Low *splitInfo* means that a low amount of information is needed to know which partition an instance belongs to. At this point, very few partitions cover the vast majority of samples, which means v is relatively small and the attribute A is less multi-variated. Therefore, *splitInfo* can be seen as a function of penalty, that is, more punishment for multi-variated attributes and less punishment for less-variated attributes. A simple theoretical proof of balance function is as follows.

Proof. $\because$ Obviously, $f(n) = |\log_2(\sqrt{n+1})\frac{\cos(3.5n-1.5)}{1.8}|$ is not strictly monotonous in the range of $[0, +\infty)$.
$\therefore \frac{infoGain}{f(n)}$ is not strictly monotonous.
$\therefore Gain = \frac{infoGain}{f(n)}$ is no longer a monotonically increasing function of the number of attribute values n.
$\because \lim_{x\to+\infty} f(n) = +\infty, n \in \left(\frac{3\pi x}{7}, \frac{4\pi x}{7}\right), n \in Z$, Z is the natural number set.
$\therefore$ In general, attribute A with more multiple values will get a higher penalty and vice versa.
$\therefore f(n)$ plays a role of balance to address the issue of multi-variated attributes. □

In Section 4, the practical effect of $f(n)$ is evaluated by experiments.

4.2. Discretization of numeric attributes

For convenient description, the definitions and properties of convexity and concavity of functions are pointed out.

Definition 1. Suppose $f(x)$ is continuous and has first-order and second-order derivatives in range of $[a, b]$, then
1) If $f''(x) > 0$ on (a, b), then the graph of $f(x)$ is convex on [a, b];
2) If $f''(x) < 0$ on (a, b), then the graph of $f(x)$ is concave on [a, b].

Property 1: Suppose $f(x)$ is a concave function on the interval I, $\forall x_1, x_2 \in I, \lambda \in [0, 1]$, then

$$\lambda f(x_1) + (1-\lambda) f(x_2) \le f(\lambda x_1 + (1-\lambda) x_2) \tag{16}$$

Property 2: Suppose $f(x)$ is a concave function on the interval I, $\forall x_1, x_2, \ldots, x_n \in I,\ \lambda_1, \lambda_2, \ldots, \lambda_n > 0,\ \lambda_1 + \lambda_2 + \ldots + \lambda_n = 1$, then

$$\lambda_1 f(x_1) + \ldots + \lambda_n f(x_n) \le f(\lambda_1 x_1 + \ldots + \lambda_n x_n) \tag{17}$$

If $x_1 = x_2 = \ldots = x_n$ or $\lambda_1 = \ldots = \lambda_{i-1} = \lambda_{i+1} = \ldots = \lambda_n$ and $\lambda_i = 1$, then the above formula takes the equality. According to the formula of information gain, the logarithmic function $f(p) = \log_2 p_i,\ (0 < p_i < 1)$ in $I(s_1, s_2, \ldots, s_m) = -\sum_{i=1}^{m} p_i \log_2 p_i$ is a concave function. Therefore, based on Property 2, we have

$$I(s_1, s_2, \ldots, s_m) = -\sum_{i=1}^{m} p_i \log_2 p_i \le -\log_2 \sum_{i=1}^{m} {p_i}^2 \tag{18}$$

where $p_i = s_i/s$ and the equality is satisfied if $p_i = p_j$, where $i, j \in [1, m]$. In a real application, if $p_i - p_j = \Delta p \to 0$, then formula (18) is converted into (19)

$$I(s_1, s_2, \ldots, s_m) = -\sum_{i=1}^{m} p_i \log_2 p_i \cong -\log_2 \sum_{i=1}^{m} {p_i}^2 \tag{19}$$

Thus, the expected information amount has an upper bound if $p_i \cong p_j$ for $i, j \in [1, m]$. We have two methods to fulfill the condition of $p_i \cong p_j$ for $i, j \in [1, m]$. One is to average the distribution of all class labels in sub-datasets. The other is to enlarge the number of bins and make fewer numbers of instances in each sub-dataset. When p_i is very close to p_j, (18) takes the

discretize(Ins, Att, k): *Ins - training dataset; Att - a numeric attribute; k - the number of different labels in a class attribute.*

Input: *Att, Ins, k*

Output: *Intervals of Att*

```
1.  Inss[][] = Partition(Ins, k); // function Partition divides Ins in to k parts, stored in 2-dimensional array Inss.
2.  Sort(Inss, Att); // function Sort sorts samples in all sub-datasets of Inss based on the value of Att in
    ascending order.
3.  For i ∈ |Inss| // |Inss| is the number of sub-datasets in Inss.
4.     Inters[i] = Generate interval [Min(Inss[i], Att), Max(Inss[i], Att)] for each sub-dataset Inss[i]; //
    Min(Inss[i]) is the minimum value of Att in Inss[i]; Max(Inss[i]) is the maximum value of Att in Inss[i];
    Inters[i] is the corresponding interval of Inss[i].
5.  End_for
6.  For i ∈ |Inters| // |Inters| is the number of intervals.
7.     subInts[i][]= Split(Inters[i]); // function Split divides each Inters[i] into independent sub-intervals based
    on numerical continuity and stores them in array subInts[i][].
8.  End_for
9.  For i ∈ |subInts| // |subInts| is the number of total sub-intervals.
10.    Merge(subInts); // function Merge combines adjacent sub-intervals based on the three basic rules.
11. End_for
```

Fig. 1. Algorithm to discretize a numeric attribute.

largest value as its approximate value. However, it is needed to get an attribute whose entropy is small by taking the purity of the sub-dataset into account for classification. Thus, compared to average the distribution of class labels, enlarging the number of bins for splitting datasets is easier to get a small entropy. This paper takes the number of class labels as the number of bins when discretizing numeric attributes.

In our approach, we first create one interval for each label in a class attribute in the training dataset. Each interval is to choose the minimum and maximum values of the numeric attribute (*e.g.*, A) in a sub-dataset (*e.g.*, s_1) as two endpoints of the interval. The purpose is as far as possible to guarantee the purity of the data, which results in, hopefully, the largest information gain after discretization. However, for an example in other sub-datasets (*e.g.*, s_2), although its attribute value of A in s_2 belongs to the corresponding interval in s_1, their class labels are possibly different. Thus, it is necessary to divide the interval further. The first step is to sort data records in all sub-datasets (*e.g.*, s_1, s_2) according to the value of the numeric attribute. Then, let any continuous part (*e.g.*, 1, 2, 3 is continuous) as an independent interval and any discontinuous part (*e.g.*, 6, 8 is discontinuous) as several single points. After that, we merge these sub-intervals based on three basic rules:

(1) Record single points and combine them with the adjacent interval (according to their class distributions);
(2) Identify the two closest adjacent intervals and combine them (according to their class distributions);
(3) Identify overlapping intervals with different class labels and compare the ratio of the records in each overlapping interval over all data records, and then remove the overlapping part in the interval with the small ratio. For example, given interval [1, 5] is labeled as class Y and interval [3, 8] is marked as class *N*. The overlapping part of the two intervals is sub-interval [3, 5]. Assume the number of data records within sub-interval [3, 5] in [1, 5] and [3, 8] are n_1 and n_2, respectively, and $\frac{n_1}{n} > \frac{n_2}{n}$ where n represents the number of total data records. Then interval [3, 8] shrinks to interval (5, 8]. Thus, records in the interval [3, 5] are tagged with class *Y*. The algorithm is shown in Fig. 1.

The key point of our discretization approach is to set the number of class labels (cn) as an upper bound to the number of branches that an internal node of a numeric attribute can fan out. That is, for each internal node $v_i \in V$, where v_i corresponds to a decision making on a numeric attribute, it must satisfy $2 \le Degree(v_i) \le cn$, where $Degree(v_i)$ is the number of outgoing arcs of node v_i. The parameter *cn* can help us to avoid too many rules of discretization. In addition, when creating branches from a node, we adopt the definition of an interval in [28]: each branch corresponds to a range of values for a numeric

Table 2
Improved Id3 algorithm.

Id3_improved algorithm: Generate_Decision_tree(S, attribute list). S is a training dataset.
Input: training dataset; numeric class attribute is discretized before constructing a classifier model.
Output: a rule-based classifier model.
(1) Create a node M
(2) If all instances in a sub-dataset are in the same class (e.g. C), execute (3)
(3) Return M as a leaf node and mark it with class C
(4) If attribute list is empty, then execute (5)
(5) Return M as a leaf node and mark it with the majority class in the dataset
(6) If attribute list is not empty, then select an attribute with the greatest InfoGain/f(n) from the attribute list as a test attribute
(7) Mark the node M as the test attribute
(8) If it is a numeric attribute, discretize it based on the class attribute values; Then execute (9)
(9) For each value a_i of the test attribute (e.g., A)
(10) Regard $A = a_i$ as a testing condition and generate a corresponding branch from the node M
(11) Let S_i be the sub-dataset when $A = a_i$
(12) If S_i is empty, then execute (13)
(13) Add a leaf node and mark it with the majority class in the dataset
(14) Else mark it with the return value of Generate_Decision_tree(S_i, {new attribute list|{attribute list}−A})
(15) Return rule-based decision list.

attribute and corresponds to a single value for a categorical attribute. Our approach allows users to build a decision tree without setting the tree branching factor beforehand.

4.3. Rule representation of classifier model

Any decision tree can be transformed to an equivalent set of rules. These kinds of rules are called decision list. For example, C4.5 contains a mechanism to re-express a decision tree as an ordered list of If-Then rules. Each path from the root of the tree to a leaf gives the conditions that must be satisfied if a case is to be classified by that leaf. However, when the tree model is very large, the number of rules generated is also very large. Some of the rules are even redundant and less actionable because they are often too specialized which may obscure the essential relationships in the domain or special cases (e.g., exceptions) [27]. If rules are extracted in a proper order, some test conditions in the head part of the rules can be deleted, which makes the rules simple and clear and saves space for storage.

This paper proposes a *TransformTreeToRules* approach to generate rule-based classifier models according to tree-based classifier models and representing rules in the form of "*IF...THEN ...*". The *TransformTreeToRules* creates a rule for each path from the root to a leaf. Conjunction of all attribute-value pairs in the path constructs the IF part of the rule, while the class label in a leaf is the THEN part. After that, a heuristic strategy of four steps for rule pruning are used to shrink the size of the rulebase:

- **Step 1**: Supplement each rule with irrelevant values. Irrelevant values are the values of irrelevant attributes which can be deleted or replaced by any values from the same domain values without affecting the correctness of the rule.
- **Step 2**: If a rule (*e.g.*, r_1) has n conditions in the head part and the former i conditions are different from other rules, then the latter $(n - i + 1)$ conditions can be deleted from r_1.
- **Step 3**: All rules are assigned to different groups in a decedent order according to the number of same conditions in the head part.
- **Step 4**: The set of rules is further reduced based on the MDL principle [29] finding the shortest description of rules of the same group by dropping repeated conditions in their head parts.

Based on the above improvements on the classical Id3, the improved algorithm iteratively constructs a decision tree as a classifier model, as shown in Table 2.

5. Implementation and experiments

5.1. Implementation, pre-processing and test option

The improved Id3 algorithm proposed in this paper is implemented by using Java programming language. The training datasets as the input file is either in .arff or .csv format, which includes the names of the attributes and a list of instances. The last attribute is always taken as the class attribute. The algorithm is executed on different datasets taken from the UCI Machine Learning Repository [20]. Since the classical Id3 algorithm cannot handle numeric attributes, the filter *NumericToNominal* from WEKA is applied, transforming numeric attributes into nominal ones. Unlike our discretization method, it takes all numerical values as nominal values of the corresponding transformed category attribute [26]. Other algorithms (e.g., id3 improved, J48, Random Tree and Decision Stump) enable to handle numeric attributes by their own internal discretization mechanism except for numeric class attributes which are pre-processed by the filter *NumericToNominal*. For missing

Table 3
Experiment datasets.

Name of dataset	No. of samples	No. of attributes	No. of classes
Acute inflammations	120	6	2
Wisconsin breast cancer	699	10	1
Lung cancer	32	56	2
Mammographic mass	961	6	1
Statlog (Heart)	270	13	1

Table 4
Comparison on dataset acute inflammations.

Acute inflammations							Instance number			120
	Accuracy	Error	Kappa	RMSE	TPR	TNR	FPR	FNR	F-measure	ACC * KS
Id3	0.7583	0.1083	0.7292	0.3536	0.8521	0.8685	0.1315	0.1479	0.8591	0.5530
Id3′	1.0000	0.0000	1.0000	0.0000	1.0000	1.0000	0.0000	0.0000	1.0000	1.0000
J48	1.0000	0.0000	1.0000	0.0000	1.0000	1.0000	0.0000	0.0000	1.0000	1.0000
DS	0.9167	0.0833	0.8333	0.2669	0.9170	0.9400	0.0600	0.0830	0.9170	0.7639
RT	0.9917	0.0083	0.9829	0.0913	0.9920	0.9940	0.0060	0.0080	0.9930	0.9747

Table 5
Comparison on dataset breast cancer Wisconsin.

Breast cancer Wisconsin							Instance number			699
	Accuracy	Error	Kappa	RMSE	TPR	TNR	FPR	FNR	F-measure	ACC * KS
Id3	0.9056	0.0401	0.9028	0.2058	0.9580	0.9497	0.0503	0.0420	0.9541	0.8176
Id3′	0.9528	0.0472	0.8970	0.2197	0.9738	0.9575	0.0425	0.0262	0.9659	0.8547
J48	0.9442	0.0558	0.8769	0.2180	0.9440	0.9350	0.0650	0.0560	0.9398	0.8280
DS	0.9242	0.0758	0.8368	0.2587	0.9240	0.9365	0.0635	0.0760	0.9250	0.7734
RT	0.9285	0.0715	0.8426	0.2675	0.9280	0.9190	0.0810	0.0720	0.9238	0.7823

values, *ReplaceMissingValues* was used which is also a filter in WEKA, replacing all missing values for nominal and numeric attributes with the modes and means from the training data [26]. Besides, the evaluation method for all the experiments is ten-folds cross validation. The output of the program includes rules in a decision list and multiple statistical parameters describing the performance of the classifier model such as accuracy and error rate.

5.2. Experiment datasets

Table 3 describes all the datasets used in the experiment.

(1) Acute inflammations data set [20]: The data were collected by a medical expert to test an expert system which performed the presumptive diagnosis of two diseases of the urinary system. It is the example of diagnosing of the acute inflammation of the urinary bladder and the acute nephritis. The number of instances and attributes is 120 and 6, respectively.

(2) Wisconsin breast cancer data set [20]: This breast cancer data set was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg. The number of instances and attributes is 699 and 10, respectively.

(3) Lung cancer data set [20]: This data was used by Hong and Young to illustrate the power of the optimal discriminant plane even in ill-posed settings. The number of instances and attributes is 32 and 56, respectively.

(4) Mammographic mass data set [20]: Mammography is the most effective method for breast cancer screening available today. This data set was used to predict the severity (benign or malignant) of a mammographic mass lesion from BI-RADS attributes and the patient's age. The number of instances and attributes is 961 and 6, respectively.

(5) Statlog (Heart) data set [20]: It is a heart disease database. This database contains 13 attributes (which was extracted from a larger set of 75) with 270 observations.

5.3. Experiment results and analysis

Based on the five datasets above, the experiment is designed to compare the performances of five classification algorithms (i.e., classical Id3, improved Id3 (*Id3′*), J48, Decision Stump (DS) and Random Tree(RT)) on 9 evaluation metrics, which is shown in Tables 4–8. The nine evaluation metrics include accuracy (ACC), error, kappa statistic (KS), root mean absolute error (RMSE), true positive rate (TPR), true negative rate (TNR), false positive rate (FPR), false negative rate (FNR), F-measure and ACC*KS. Their comparison on ACC, KS, RMSE and ACC * KS are taken out for further analysis, as shown in Figs. 2–5. Note that since there may exist unclassified instances, the sum of accuracy and error may not be one.

Table 6
Comparison on dataset lung cancer.

Lung cancer							Instance number			32
	Accuracy	Error	Kappa	RMSE	TPR	TNR	FPR	FNR	F-measure	ACC * KS
Id3	0.5000	0.4375	0.3046	0.5578	0.5330	0.7570	0.2430	0.4670	0.5160	0.1523
Id3′	0.5313	0.4687	0.2962	0.5590	0.5313	0.5313	0.4687	0.4687	0.5313	0.1574
J48	0.5000	0.5000	0.2289	0.5005	0.5000	0.7160	0.2840	0.5000	0.5020	0.1145
DS	0.4375	0.5625	0.1504	0.4797	0.4380	0.7000	0.3000	0.5620	0.4320	0.0658
RT	0.4375	0.5625	0.1628	0.5848	0.4380	0.7110	0.2890	0.5620	0.4220	0.0712

Table 7
Comparison on dataset mammographic mass.

Mammographic mass							Instance number			961
	Accuracy	Error	Kappa	RMSEr	TPR	TNR	FPR	FNR	F-measure	ACC * KS
Id3	0.6545	0.1977	0.5317	0.4648	0.7680	0.7620	0.2380	0.2320	0.7680	0.3480
Id3′	0.8262	0.1738	0.6499	0.3645	0.8460	0.8277	0.1774	0.1540	0.8362	0.5370
J48	0.8210	0.1790	0.6381	0.3662	0.8210	0.8130	0.1870	0.1790	0.8200	0.5239
DS	0.8189	0.1811	0.6309	0.3825	0.8190	0.8030	0.1970	0.1810	0.8160	0.5167
RT	0.7659	0.2341	0.5262	0.4604	0.7660	0.7570	0.2430	0.2340	0.7650	0.4030

Table 8
Comparison on dataset statlog heart.

Statlog heart							Instance number			270
	Accuracy	Error	Kappa	RMSE	TPR	TNR	FPR	FNR	F-measure	ACC * KS
Id3	0.3519	0.2000	0.2580	0.6020	0.6380	0.6230	0.3770	0.3620	0.6390	0.0908
Id3′	0.7778	0.2222	0.5530	0.4699	0.8020	0.7789	0.2211	0.1980	0.7928	0.4301
J48	0.7667	0.2333	0.5271	0.4601	0.7670	0.7600	0.2400	0.2330	0.7670	0.4041
DS	0.7259	0.2741	0.4450	0.4477	0.7260	0.7190	0.2810	0.2740	0.7260	0.3230
RT	0.7407	0.2593	0.4732	0.5092	0.7410	0.7310	0.2690	0.2590	0.7400	0.3505

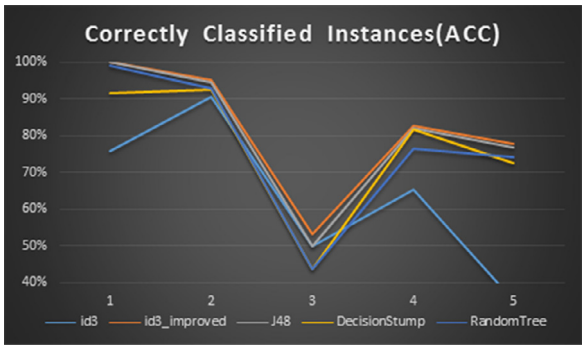

Fig. 2. Correctly classified instances (ACC).

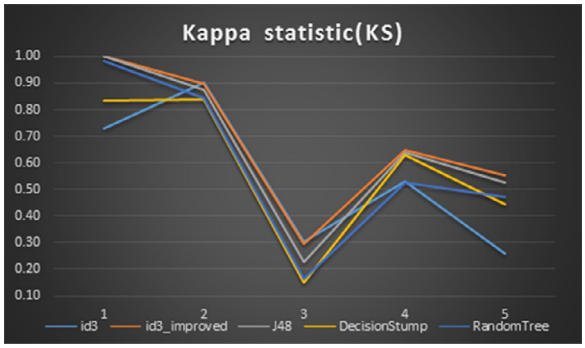

Fig. 3. Kappa statistic (KS).

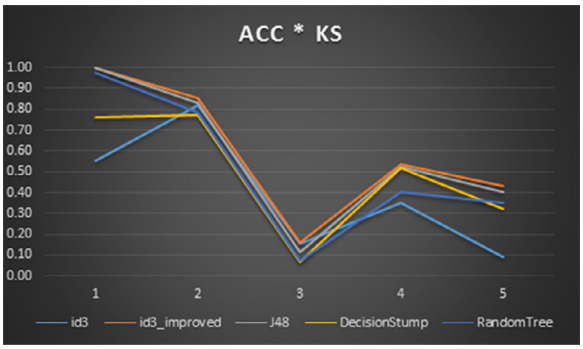

Fig. 4. ACC * KS.

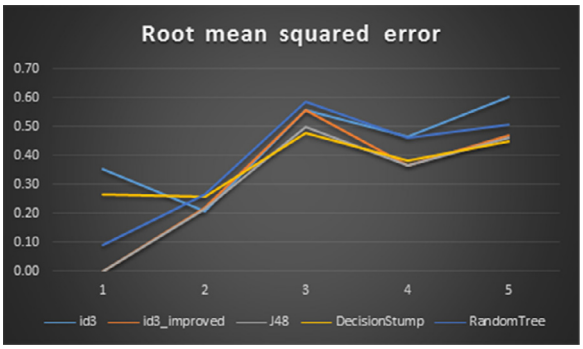

Fig. 5. Root mean squared error (RMSE).

From the experiment results, the five algorithms perform differently on different datasets. This is because the classification results partially depend on how the distributions of class attributes are in each dataset and how relevant the non-class attribute with the class attribute is. Specifically, in term of accuracy (Fig. 2) and reliability (Figs. 3 and 4), improved Id3, J48 and RT perform better than classical Id3 and DS. The performance of the five algorithms degrades largely on the third dataset. As to the fifth dataset, the improved Id3 is better than the other methods and Id3 is the worst followed by J48, RT and DS. RMSE means the average difference between the true value of interest (η) and the value estimated (η') using the algorithms. Compared with mean absolute error (MAE), RMSE presents the influence of the extreme value on the result more clearly. As shown in Fig. 5, the improved Id3 and J48 are better than the other algorithms with almost zero RMSE on the first dataset, while classical Id3 and DS have 28% and 35% error rate, respectively. All the five methods have about 45% – 60% error rate on the third dataset. As to the fifth one, DS is the best, while J48, improved Id3 and RT are nearly the same, leaving the classical Id3 behind.

This paper takes rule-based representation of classifier models. Take Wisconsin Breast Cancer Dataset for example. The constructed tree-based classifier model has 8 layers with 42 internal nodes and 42 leaf nodes. Fig. 6 shows the corresponding rule-based classifier model. For each rule in Fig. 6, there is a probability which represents the likelihood of the occurrence of the result if the condition is satisfied, which is calculated by (20).

$$\text{Probability}(rule) = \frac{The\ number\ of\ the\ instances\ meeting\ the\ rule}{The\ number\ of\ the\ total\ instances} \tag{20}$$

For example, in Fig. 6, the first rule in the decision list is "IF Uniformity_of_Cell_Size = '(-inf-1.5]' + Bare Nuclei = '(-inf-1.5]' THEN benign". The number of instances that satisfy the rule is 404 and the total number of instances is 669. Thus, the probability of this rule is 0.6039, which means if a person whose uniformity of cell size and bare nuclei are in the range of (-inf-1.5], his/her breast cancer is benign with the probability of 0.6039. Fig. 7 shows the comparison of the average memory cost of the tree-based classifier models generated by other four decision-tree classification algorithms with the one of our rule-based classifier model.

6. Conclusions

In this paper, we propose an improved Id3 classification algorithm to facilitate accurate and reliable clinical diagnosis. The classical Id3 algorithm is improved in three aspects. First, a theory of balance function is introduced to reduce the weight of attributes with multiple different values. It follows that an attribute with a higher distinctive degree is more likely to be taken as a split attribute. Second, a novel discretization algorithm is implemented to transform numeric attributes to

```
== Rule-based Classifier Model ==

IF Uniformity_of_Cell_Size = '(-inf-1.5]' + Bare Nuclei = '(-inf-1.5]' THEN benign
IF Bare Nuclei = '(1.5-inf)' + Normal_Nucleoli = '(-inf-1.5]' + Marginal_Adhesion = '(-inf-1.5]' THEN benign
IF Marginal_Adhesion = '(1.5-inf)' + Clump_Thickness = '(-inf-4.5]' + Single_Epithelial_Cell_Size = '(-inf-2.5]'
+ Mitoses = '(-inf-1.5]' THEN benign
IF Mitoses = '(1.5-inf)' THEN benign
IF Single_Epithelial_Cell_Size = '(2.5-inf)' THEN benign
IF Clump_Thickness = '(4.5-inf)' THEN benign
IF Normal_Nucleoli = '(1.5-inf)' + Clump_Thickness = '(-inf-4.5]' THEN benign
... ...
```

Fig. 6. Rule-based classifier model.

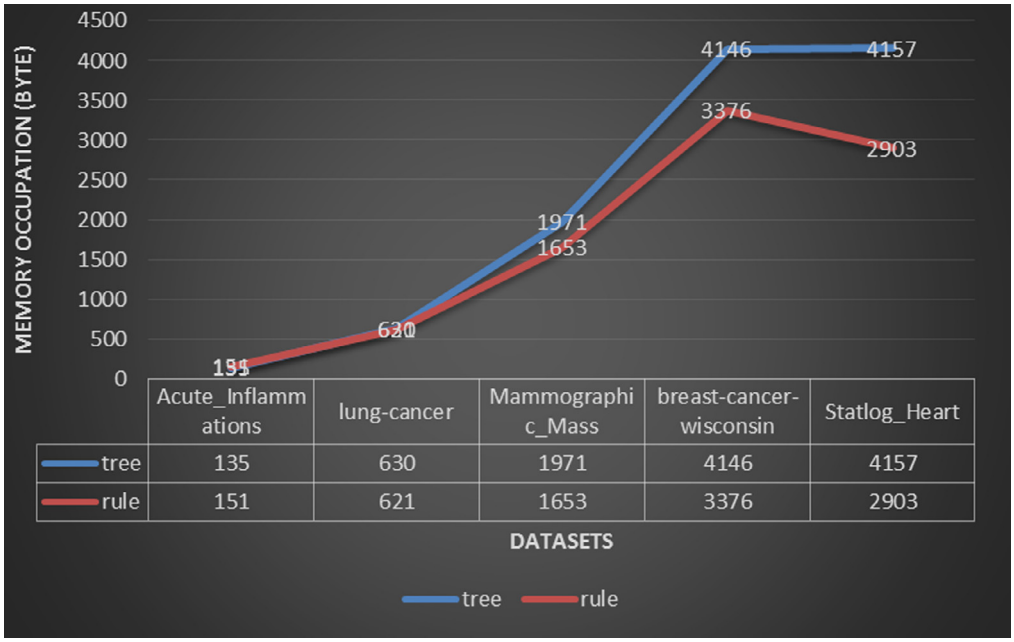

Fig. 7. Comparison in memory saving.

nominal ones. This new approach avoids randomly selecting an optimal partition number manually. Finally, we propose a rule-based heuristic strategy to represent the classifier model for easy understanding and memory saving. The experiments on medical and health areas show that the improved Id3 algorithm enhances the performance in term of accuracy, reliability and minor error rate compared with the classical Id3 and other three classification algorithms (J48, Decision Stump and Random Tree).

This paper has three core contributions to the academic field. First, it proposes an improved Id3 algorithm for accurate and reliable disease prediction. Second, it mathematically proves that the classical Id3 algorithm is biased to multi-variated attributes and how the newly proposed balance functions can address this issue. Thus, an attribute that has many different values but does not have the highest distinctive degree will not be chosen as a split attribute. Third, it mathematically proves the best partition number for numeric attribute discretization.

In future, the classification algorithm should be further improved to meet the big data computational demand. For example, in data stream mining, the tree-based classifier model should not be allowed for model re-construction but for model update. This is because there is no upper bound on the number of instances that arrive sequentially.

Acknowledgements

This research is supported by University of Macau Research Committee [grant numbers MYRG2015-00043-FST and MYRG2017-00091-FST].

References

[1] Berikol GB, Yildiz O, Özcan IT. Diagnosis of acute coronary syndrome with a support vector machine. J Med Syst 2016;40(4):1–8.
[2] Shen L, Chen H, Yu Z, Kang W, Zhang B, Li H, et al. Evolving support vector machines using fruit fly optimization for medical data classification. Knowl Based Syst 2016.
[3] Mahendran G, Dhanasekaran R. Investigation of the severity level of diabetic retinopathy using supervised classifier algorithms. Comput Electr Eng 2015;45:312–23.
[4] Ozkan O, Yildiz M, Arslan E, Yildiz S, Bilgin S, Akkus S, et al. A study on the effects of sympathetic skin response parameters in diagnosis of fibromyalgia using artificial neural networks. J Med Syst 2016;40(3):1–9.
[5] Raposo LM, Arruda MB, de Brindeiro RM, Nobre FF. Lopinavir resistance classification with imbalanced data using probabilistic neural networks. J Med Syst 2016;40(3):1–7.
[6] Er O, Tanrikulu AC, Abakay A, Temurtas F. An approach based on probabilistic neural network for diagnosis of mesotheliomas disease. Comput Electr Eng 2012;38(1):75–81.
[7] Chao C-M, Yu Y-W, Cheng B-W, Kuo Y-L. Construction the model on the breast cancer survival analysis use support vector machine, logistic regression and decision tree. J Med Syst 2014;38(10):1–7.
[8] Bilge U, Bozkurt S, Durmaz S. Application of data mining techniques for detecting asymptomatic carotid artery stenosis. Comput Electr Eng 2013;39(5):1499–505.
[9] Cerquitelli T, Chiusano S, Xiao X. Exploiting clustering algorithms in a multiple-level fashion: a comparative study in the medical care scenario. Expert Syst Appl 2016;55:297–312.
[10] Amami R, Smiti A. An incremental method combining density clustering and support vector machines for voice pathology detection. Comput Electr Eng 2017;57:257–65.
[11] Luo S-T, Cheng B-W. Diagnosing breast masses in digital mammography using feature selection and ensemble methods. J Med Syst 2012;36(2):569–77.
[12] Haseena HH, Mathew AT, Paul JK. Fuzzy clustered probabilistic and multi layered feed forward neural networks for electrocardiogram arrhythmia classification. J Med Syst 2011;35(2):179–88.
[13] Quinlan JR. Induction of decision trees. Mach Learn 1986;1(1):81–106.
[14] Agrawal R, Imielinski T, Swami A. Database mining: a performance perspective. IEEE Trans Knowl Data Eng 1993;5(6):914–25.
[15] Salzberg SL. C4. 5: Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993. Mach Learn 1994;16(3):235–40.
[16] Mehta M, Agrawal R, Rissanen J. Sliq: A fast scalable classifier for data mining. In: International conference on extending database technology. Springer; 1996. p. 18–32.
[17] Shafer J, Agrawal R, Mehta M. Sprint: A scalable parallel classi er for data mining. In: Proc. 1996 int. conf. very large data bases. Citeseer; 1996. p. 544–55.
[18] Wang H, Zaniolo C. Cmp: A fast decision tree classifier using multivariate predictions. In: Data engineering, 2000. Proceedings. 16th international conference on. IEEE; 2000. p. 449–60.
[19] Chen Y-L, Hsu C-L, Chou S-C. Constructing a multi-valued and multi-labeled decision tree. Expert Syst Appl 2003;25(2):199–209.
[20] Uci machine learning repository. 2007. URL http://archive.ics.uci.edu/ml.
[21] Seera M, Lim CP, Tan SC, Loo CK. A hybrid fam–cart model and its application to medical data classification. Neural Comput Appl 2015;26:1799–811.
[22] Budnik M, Krawczyk B. On optimal settings of classification tree ensembles for medical decision support. Health Inform J 2013;19(1):3–15.
[23] Kobayashi D, Takahashi O, Arioka H, Koga S, Fukui T. A prediction rule for the development of delirium among patients in medical wards: chi-square automatic interaction detector (chaid) decision tree analysis model. Am J Geriatr Psychiatry 2013;21:957–62.
[24] Pauly O. Random forests for medical applications. Ph.D. thesis; Technische Universität München; 2012.
[25] Chou S, Hsu C-L. Mmdt: a multi-valued and multi-labeled decision tree classifier for data mining. Expert Syst Appl 2005;28(4):799–812.
[26] Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH. The weka data mining software: an update. ACM SIGKDD Explor Newslett 2009;11(1):10–18.
[27] Liu B, Hu M, Hsu W. Intuitive representation of decision trees using general rules and exceptions. In: AAAI/IAAI; 2000. p. 615–20.
[28] Agrawal R, Ghosh S, Imielinski T, Iyer B, Swami A. An interval classi er for database mining applications. In: Proc. of the VLDB conference; 1992. p. 560–73.
[29] Robinet V, Lemaire B, Gordon MB. Mdlchunker: a mdl-based cognitive model of inductive learning. Cogn Sci 2011;35(7):1352–89.

Shuo Yang is pursuing his Ph.D. in the Department of Computer and Information Science at the University of Macau. His research interests include data mining, semantic interoperability and semantic inference.

Jingzhi Guo is currently working as an associate professor in the Department of Computer and Information Science, University of Macau. His research interests include concept representation, semantic integration, and collaboration systems.

Junwei Jin is pursuing his Ph.D. in the Department of Computer and Information Science at the University of Macau. His research interests are sparse representation, computer vision and machine learning.