Knowledge-Based Systems 138 (2017) 208–219

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

A multi-view OVA model based on decision tree for multi-classification tasks

Xiaoqiang Guan*, Jiye Liang, Yuhua Qian, Jifang Pang

Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, School of Computer and Information Technology, Shanxi University, Taiyuan, 030006, Shanxi, China

ARTICLE INFO

Article history:
Received 22 August 2016
Revised 15 August 2017
Accepted 2 October 2017
Available online 3 October 2017

Keywords:
Decision tree
Multi-class classification
OVA
Membership vector

ABSTRACT

Decision tree is a simple classification algorithm and has been widely used in knowledge discovery and pattern recognition fields, which can be used to deal with the multi-classification tasks. In this paper, we present a multi-view OVA model based on decision tree (MVDT) for multi-classification tasks to simplify the structure of the decision tree and improve the generalization ability. A multi-class classification task is divided into c multiple parallel sub-tasks, and MVDT builds c decision trees as base binary classifiers for each sub-task. Each decision tree gives membership vector for each leaf node to estimate the probabilities of the instances in the leaf node belonging to negative classes, as well as presents a precise classification for positive class. Thus, one can obtain more information about instances belonging to negative classes through membership vectors, which helps to achieve higher accuracy and better robustness for classification. As a general framework, MVDT algorithm can use any existing decision tree model as base classifier. To evaluate the performance of our algorithm, we choose C4.5, CART, TEIM, SCDT and NBTree as base classifiers in MVDT. The experiments on 22 data sets show that the proposed MVDT has excellent performance for multi-class classification problems and has excellent robustness to output noise.

1. Introduction

Data mining is used to find out important and meaningful knowledge in large scale data. Multi-class classification involves training of instances for different classes so as to further enable the identification of classes for various unknown instances, and widely exists in data mining and machine learning area. There are many classification algorithms, such as decision tree [1,2], neural networks [3,4], support vector machines [5] and Bayesian networks [6,7], etc. Among them, decision tree is a simple method and has been widely used in knowledge discovery and pattern recognition fields such as medical diagnosis [8,9] and credit risk assessments[10].

Decision tree is an inductive learning algorithm based on instances, which focuses on the classification rules in the form of decision trees from a set of non-order, non-regular instances. By using tree structure to make decision, decision tree is a natural mechanism for human beings to deal with the decision making problems. It has a high predictive performance for a relatively small computational effort and can produce a comprehensible classification/regression model. Decision tree has been proved to be one of the most powerful and popular approaches to discover useful patterns in data science.

In decision tree induction algorithms, ID3 [1] and its extension C4.5 [11] are very popular and successful. The split criterion of ID3 is founded on information entropy. ID3 has some deficiencies, including the tendency of selecting attributes with more values, poor anti-noise ability, and the inability to deal with miss values [12]. C4.5 is an extension version of ID3 to deal with continuous attributes and missing values. C4.5 uses information gain ratio instead of information gain as split criterion to obtain a finer partition, and uses a pessimistic error prunning method to reduce overfitting. Another popular decision tree algorithm is CART developed by Breiman et al. [13], which uses Gini index as its split criterion and creates binary trees from data with continuous and discrete attributes. In CART, cost complexity pruning method is adopted to reduce overfitting.

To further improve the performance of classical decision trees, many researchers then proposed a lot of improvements. Some algorithms have been proposed to improve the split criterion [14–25]. Chandra et al. [16] proposed a new node splitting measure named distinct class based splitting measure(DCSM) to provide

* Corresponding author.
E-mail addresses: gxq0079@sxu.edu.cn (X. Guan), ljy@sxu.edu.cn (J. Liang), ljinchengqyh@126.com (Y. Qian), purplepjf@sxu.edu.cn (J. Pang).

https://doi.org/10.1016/j.knosys.2017.10.004
0950-7051/

higher classification accuracy. Mantas and Abellán [19] showed that Credal Decision Tree (CDT) with Imprecise Information-Gain (IIG) and Complete Credal Decision Tree(CCDT) with Complete Imprecise Information-Gain(CIIG) can obtain better performance and stability than those based on the information entropy. Asadi and Ghatee [23] proposed a new decision tree (named as AGDT in this paper) where experts' perception is used to select branching variables at the primary levels in construction of decision tree to obtain more reasonable results. Wu et al. [24] proposed a Size Constrained Decision Tree (SCDT), in which the categorical attributes are divided into two groups before node splitting and the number of leaf nodes is constrained through a threshold. Wang et al. [25] proposed Tsallis Entropy Information Metric (TEIM) algorithm using a new split criterion based on two-term Tsallis conditional entropy, which has better stability and adaptability to data sets. In TEIM, the minimum leaf size is set to 5 to avoid overfitting. Some researchers also proposed improved decision tree algorithms to deal with ordinal classification, time series, gene expression data and continuous valued attributes etc. [26–34]. Qian et al. [27] proposed a fusing monotonic decision tree for ordinal classification, Yi et al. [31] proposed an AMDT algorithm for multi-valued and multi-labeled data, and Ludwig et al. [30] presented a fuzzy decision tree algorithm on gene expression data.

Multi-class classification problems can also be solved by decomposing the original problems into several binary classification tasks that can be solved efficiently using binary classifiers. Rifkin and Klautau [35] argued that One-versus-all (OVA) is one of the most efficient decomposition strategies for multi-class classification problems. In OVA, given c classes, c binary classifiers are built, each one considers one of the classes as the "positive" class while the remainders are combined into a "negative" class. For each binary classifier, if an instance is classified as negative, then all classes except the positive one get a vote. Wang et al. [36] proposed a multi-class multi-nomial naive Bayes tree (MMNBTree) for text classification. In MMNBTree, for each binary classifier, if the membership probability of an instance belonging to positive is p, then the membership probabilities of the instance belonging to all negative classes are assigned to $1 - p$. The minimum leaf size is set to $|D|$*40% to avoid overfitting in MMNBTree, where $|D|$ is the size of dataset.

Both OVA and MMNBTree assign equal membership probabilities of an instance belonging to negative classes, while the membership probabilities might be various in a binary classifier. If each binary classifier can provide not only the information about the positive class but also some useful information about the negative classes, then the accuracy of the multi-class classification algorithm might be promoted. In order to solve multi-class classification problems better, we propose a multi-view OVA model based on decision tree (MVDT), which adopts multiple decision trees as base classifiers. An instance can be classified as "positive" by a decision tree, or it can be classified as "negative" by the decision tree with various membership probabilities belonging to negative classes. For an instance, we combine the membership probabilities of the instance belonging to some class given by all the decision trees, and then classify the instance as the class with highest combined probability.

In MVDT, a multi-class classification task can be divided into multiple parallel sub-tasks, and a decision tree will be built for solving a sub-task. For the whole classification task, each decision tree presents a precise classification for positive class and gives various probability estimates for negative classes. It is showed that the proposed MVDT has excellent performance and good robustness to noise data for multi-class classification problems and can effectively avoid overfitting.

This paper is organized as follows. Section 2 briefly describes basic concepts of decision trees and multi-class classification problems. Section 3 introduces the proposed multi-view OVA model based on decision tree (MVDT). Section 4 presents and analyzes the compared experimental results with other multi-class classifications, such as C4.5, CART, TEIM, SCDT, MMNBTree etc. Section 5 gives an analysis and conclusion.

2. Previous knowledge

2.1. Multi-class classification problems

Multi-class classification problems exist widely in real life, such as fingerprints and medicine. Rifkin and Klautau[35] argued that One-versus-all (OVA) is one of the most efficient decomposition strategies for multi-class classification problems.

In OVA, a dataset D with c classes is divided into c binary classifiers. Each classifier considers one of the classes as the "positive" class while the remainders are combined into a "negative" class. Given a test instance, classifier that gives a positive indicates the output class. If the positive output is unique, then the final classification results of the test instance are marked as the corresponding class. In many cases, the positive output is not unique, the common approach uses the confidence of the classifier to decide the final output, predicting the class from the classifier with the largest confidence. OVA uses a score vector $R = (r_1, r_2, \ldots, r_c)$ [37] (where $r_i \in [0, 1]$ is probability estimate of the instance belonging to class i) to evaluate the outputs.

2.2. Decision trees

Decision tree models have been successfully used for multi-class classification problems. It is based on the tree structure to make decisions and is a natural mechanism for human beings to deal with the decision making problems.

A decision tree can be viewed as a classifier expressed as a recursive partition of the instance space, which is usually constructed top-down recursively from a training data set. In other words, each internal node splits the instance space into two or more sub-spaces according to the values of input attributes, so each path in the decision tree from root node to leaf node corresponds to a conjunction rule of attribute tests. The decision tree can be regarded as a disjunction of conjunctions of constraints on the attribute values of instances [38].

Once a decision tree has been constructed by the training set, it can be used to classify an unknown instance set based on the values of the attributes. When a decision tree is used to classify an instance, it begins to test the values of the attributes gradually from the root node, and then goes along the appropriate branch down until it reaches a leaf node. The class label represented by the leaf node is then assigned to the test instance. Every decision problem put forward in the decision-making process is a "test" for a certain attribute.

A decision tree itself can handle multi-class classification problems, in which all classes can be distinguished by the establishment of a fully grown tree. In order to divide the training set as pure as possible, a decision tree needs to select the best splitting attribute(s) for all classes. Then, the size of the decision tree might be very large. Therefore, in decision tree algorithms, the complexity of decision tree model is an important factor which needs to be considered. There might be enormous amount of leaf nodes in a fully grown tree, and only a few training instances in each leaf node. If the algorithm searches too long or concentrates too much on a few hard-to-learn instances, the problem of over-fitting can occur. And the performance of the decision tree might be also reduced when dealing with a noise data set.

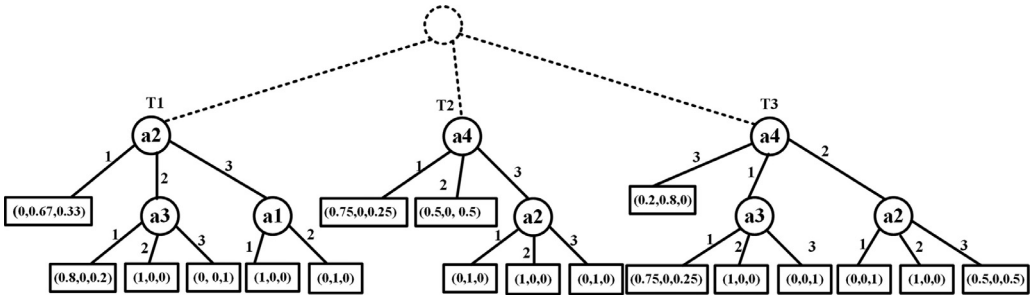

Fig. 1. MVDT-ID3 decision tree.

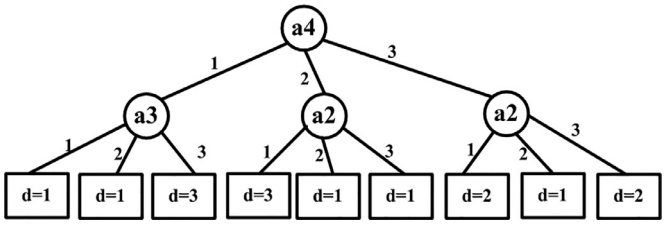

Fig. 2. ID3 decision tree.

Table 1
An artificial data set.

Instances	a_1	a_2	a_3	a_4	d
x_1	1	2	2	1	ω_1
x_2	1	2	2	2	ω_1
x_3	2	2	2	1	ω_1
x_4	2	3	2	3	ω_2
x_5	1	3	2	1	ω_1
x_6	1	2	3	1	ω_3
x_7	2	3	1	3	ω_2
x_8	1	2	2	3	ω_1
x_9	1	1	3	3	ω_2
x_{10}	2	1	2	3	ω_2
x_{11}	1	1	2	2	ω_3
x_{12}	1	3	3	1	ω_1
x_{13}	2	1	2	3	ω_2

Table 2
Results of ID3 and MVDT-ID3 on testing data in Table 1.

Instances	Real labels	labels by ID3	labels by MVDT
x_{12}	ω_1	ω_3	ω_1
x_{13}	ω_2	ω_2	ω_2

Table 3
Summary of the used data sets.

Dataset	#Ex.	#Atts.	#Cl.	IR
iris	150	4	3	1
texture	5500	40	11	1
mfeat-fac	2000	216	10	1
mfeat-fou	2000	76	10	1
mfeat-kar	2000	64	10	1
mfeat-mor	2000	6	10	1
mfeat-pix	2000	240	10	1
mfeat-zer	2000	47	10	1
wavform	5000	40	3	1.02
optdigits	5620	64	10	1.03
tae	151	5	3	1.06
letter	20,000	16	26	1.11
wine	178	13	3	1.48
cmc	1473	9	3	1.59
molecula	3190	60	3	2.16
satimage	6435	36	6	2.45
balance-scale	625	4	3	5.88
glass	214	9	7	8.44
car	1728	6	4	18.62
ecoli	336	7	8	71.5
yeast	1484	8	10	92.6
nursery	12,960	8	5	2160

3. A multi-view OVA model based on decision tree (MVDT)

We adopt multi-view OVA to the proposed MVDT model to improve the performance of decision tree. The main idea is to divide a multi-class classification task into c multiple parallel sub-tasks and build c decision trees for these sub-tasks. The number of decision trees is determined by the number of classes in the training set. Each decision tree is corresponding to a binary classification task for one class, which gives membership vector for each leaf node to estimate the probabilities of the instances in the leaf node belonging to negative classes, as well as presents a precise classification for positive class. Thus, we can obtain more information about negative instances through membership vectors, which helps to achieve higher accuracy for classification. For a minority class, one could obtain a more accurate prediction through the decision tree corresponding to the class. Hence, MVDT shows a significant performance on the imbalance data sets. When there is noise in training set, only a few of decision trees might be sensitive to noise data and the accuracy of these decision trees might be reduced. But most of the decision trees would be slightly affected by noise data. The combination result of these trees could still maintain a higher prediction accuracy. Hence, our algorithm is also robust to data set with output noise.

The proposed multi-view OVA model based on decision tree will first establish c decision trees as base classifiers for each class.

3.1. The construction of the base classifier

For a c-class problem, c decision trees will be established in MVDT algorithm, where the ith decision tree presents a precise classification of the ith class. That is, each instance in a leaf node of the ith decision tree will be classified as positive or as negative to the ith class.

For convenience, we first give some definitions and notations.

Given a training data set $\{(X_1, y_1), (X_2, y_2), \ldots, (X_n, y_n)\}$, where $X_j \in \mathbb{R}^m$ denotes the jth instance. Let $Y = \{y_1, y_2, \ldots, y_n\}^T$ be label vector for X, where $y_j \in \{\omega_1, \omega_2, \ldots, \omega_c\}$ is the class label of the jth instance.

Each decision tree is a binary classifier for the class under discussion. In order to build binary classifier, we need to define the binary label vector for each class, see Definition 1.

Definition 1. For an instance set X and its label vector $Y = \{y_1, y_2, \ldots, y_n\}^T$, the **binary label vector** $Y_i' = \{y_{i1}', y_{i2}', \ldots, y_{in}'\}^T$

Table 4
Performance analysis of algorithms with C4.5 as base classifier.

Dataset	Accuracy			F1-measures			Precision			AUC		
	C4.5	OVA-C4.5	MVDT-C4.5	C4.5	OVA-C4.5	MVDT-C4.5	C4.5	OVA-C4.5	MVDT-C4.5	C4.5	OVA-C4.5	MVDT-C4.5
iris	0.9620 (2)	0.9567 (3)	0.9633 (1)	0.9589 (2)	0.9537 (3)	0.9605 (1)	0.9622 (2)	0.9583 (3)	0.9637 (1)	0.9722 (3)	0.9799 (2)	0.9845 (1)
texture	0.9215 (2)	0.9049 (3)	0.9377 (1)	0.9210 (2)	0.9069 (3)	0.9369 (1)	0.9218 (2)	0.9211 (3)	0.9380 (1)	0.9646 (3)	0.9895 (2)	0.9957 (1)
mfeat-fac	0.8768 (1)	0.7970 (3)	0.8537 (2)	0.8745 (1)	0.8048 (3)	0.8514 (2)	0.8790 (1)	0.8493 (3)	0.8569 (2)	0.9763 (1)	0.9313 (3)	0.9608 (2)
mfeat-fou	0.7499 (2)	0.6463 (3)	0.7753 (1)	0.7468 (2)	0.6448 (3)	0.7677 (1)	0.7518 (2)	0.6967 (3)	0.7710 (1)	0.8802 (3)	0.9120 (2)	0.9618 (1)
mfeat-kar	0.8139 (2)	0.7695 (3)	0.8791 (1)	0.8093 (2)	0.7788 (3)	0.8756 (1)	0.8142 (3)	0.8329 (2)	0.8783 (1)	0.9006 (3)	0.9569 (2)	0.9844 (1)
mfeat-mor	0.6380 (2)	0.5855 (3)	0.6731 (1)	0.6328 (2)	0.5738 (3)	0.6661 (1)	0.6381 (2)	0.6019 (3)	0.6715 (1)	0.8676 (2)	0.8324 (3)	0.9522 (1)
mfeat-pix	0.8540 (2)	0.8413 (3)	0.8937 (1)	0.8510 (2)	0.8468 (3)	0.8909 (1)	0.8550 (3)	0.8812 (2)	0.8943 (1)	0.9222 (3)	0.9721 (2)	0.9856 (1)
mfeat-zer	0.6561 (2)	0.6103 (3)	0.7154 (1)	0.6559 (2)	0.6068 (3)	0.7081 (1)	0.6657 (2)	0.6472 (3)	0.7097 (1)	0.8533 (3)	0.9060 (2)	0.9591 (1)
wavform	0.7502 (2)	0.7326 (3)	0.7617 (1)	0.7500 (2)	0.7300 (3)	0.7584 (1)	0.7507 (3)	0.7599 (2)	0.7667 (1)	0.8200 (3)	0.8869 (2)	0.9108 (1)
optdigits	0.8467 (2)	0.8264 (3)	0.9021 (1)	0.8458 (2)	0.8337 (3)	0.9001 (1)	0.8468 (3)	0.8719 (2)	0.9018 (1)	0.9129 (3)	0.9717 (2)	0.9891 (1)
tae	0.5940 (3)	0.6197 (1)	0.6135 (2)	0.5780 (3)	0.5975 (1)	0.5924 (2)	0.6074 (3)	0.6299 (1)	0.6238 (2)	0.7024 (3)	0.7465 (2)	0.7866 (1)
letter	0.8521 (2)	0.8055 (3)	0.8752 (1)	0.8512 (2)	0.8235 (3)	0.8744 (1)	0.8522 (3)	0.8647 (2)	0.8796 (1)	0.9191 (3)	0.9590 (2)	0.9902 (1)
wine	0.8692 (3)	0.8917 (2)	0.9267 (1)	0.8637 (3)	0.8815 (2)	0.9202 (1)	0.8717 (3)	0.9032 (2)	0.9256 (1)	0.9008 (3)	0.9743 (2)	0.9784 (1)
cmc	0.4654 (3)	0.4718 (2)	0.4913 (1)	0.4430 (2)	0.4167 (3)	0.4651 (1)	0.4453 (2)	0.4389 (3)	0.4693 (1)	0.6045 (3)	0.6315 (2)	0.6728 (1)
molecula	0.8957 (2)	0.8898 (3)	0.9137 (1)	0.8839 (2)	0.8790 (3)	0.9023 (1)	0.8833 (3)	0.8902 (2)	0.9111 (1)	0.9135 (3)	0.9654 (2)	0.9735 (1)
satimage	0.8494 (2)	0.8210 (3)	0.8757 (1)	0.8246 (2)	0.8015 (3)	0.8440 (1)	0.8257 (2)	0.8256 (3)	0.8660 (1)	0.9176 (3)	0.9501 (2)	0.9763 (1)
balance-scale	0.7663 (2)	0.7311 (3)	0.8031 (1)	0.5687 (2)	0.5630 (3)	0.5797 (1)	0.5845 (2)	0.5957 (1)	0.5803 (3)	0.8317 (3)	0.8701 (2)	0.8857 (1)
glass	0.5553 (3)	0.6003 (2)	0.6583 (1)	0.4406 (3)	0.4737 (2)	0.5037 (1)	0.4619 (3)	0.5053 (2)	0.5078 (1)	0.6911 (3)	0.7976 (2)	0.8640 (1)
car	0.9785 (2)	0.9755 (3)	0.9789 (1)	0.9447 (1)	0.9423 (2)	0.9406 (3)	0.9521 (3)	0.9747 (1)	0.9605 (2)	0.9653 (3)	0.9911 (1)	0.9907 (2)
ecoli	0.8270 (2)	0.8158 (3)	0.8515 (1)	0.6578 (2)	0.6481 (3)	0.6908 (1)	0.6896 (2)	0.6871 (3)	0.7370 (1)	0.8716 (3)	0.8855 (2)	0.9470 (1)
yeast	0.4562 (2)	0.4494 (3)	0.5493 (1)	0.3207 (2)	0.2980 (3)	0.3868 (1)	0.3369 (3)	0.3612 (2)	0.4198 (1)	0.6589 (3)	0.7196 (2)	0.8272 (1)
nursery	0.9971 (2)	0.9871 (3)	0.9977 (1)	0.9516 (2)	0.8778 (3)	0.9587 (1)	0.9509 (2)	0.8760 (3)	0.9595 (1)	0.9632 (3)	0.9759 (2)	0.9948 (1)
Avg.Rank	2.136	2.773	1.091	2.045	2.773	1.182	2.455	2.318	1.227	2.864	20.45	1.091
Friedman $\alpha = 0.05$	✓	✓	–	✓	✓	–	✓	✓	–	✓	✓	–

relative to ith class is defined as follows:

$$y'_{ij} = \begin{cases} 1, & y_j = \omega_i \\ 0, & y_j \neq \omega_i \end{cases}, where\ 1 \le j \le n, and\ 1 \le i \le c \tag{1}$$

We can select any existing decision tree algorithm to build a decision tree T_i as a binary classifier for ith class from instance set X and its binary label vector Y'_i is defined by Definition 1. Thus, for a c-class problem, we can build c decision trees T_1, T_2, $\cdots$, T_c as base classifiers.

If the decision tree T could offer some useful information of negative classes for instances in a leaf node v when a leaf node v in T is identified as negative, we might obtain more accurate prediction result. Hence, we define the membership vector of a leaf node v in a decision tree T as Definition 2.

Definition 2. Let X be a training set and Y be the label vector for X, the **membership vector** of a leaf node v in a decision tree T is defined as $MS_v^{(T)} = (s_{v1}^{(T)}, s_{v2}^{(T)}, \ldots, s_{vc}^{(T)})$, where

$$s_{vk}^{(T)} = \frac{|B_v(k)|}{|B_v|}, k = 1, 2, \cdots, c \tag{2}$$

where B_v is the instance set of node v, and $B_v(k)$ is the instance set belonging to the kth class of node v. The superscripts can be omitted when without causing confusing.

Clearly, in a decision tree T, the membership vector MS_v of a leaf node v represents the membership probability of the instances in v belonging to classes $\{\omega_1, \omega_2, \ldots, \omega_c\}$.

The construction process of base classifiers are depicted as Algorithm 1.

3.2. The multi-View OVA model based on decision tree (MVDT)

In Algorithm MVDT, we first build c decision trees as base classifiers. Each decision tree T_i is a binary classifier for class ω_i. If a leaf node v of T_i outcomes positive, then we can definitely classify

Algorithm 1 Generation of base classifiers.

Input: Training set $\{(X_1, y_1), (X_2, y_2), \ldots, (X_n, y_n)\}$;
Let $X = \{X_1, X_2, \ldots, X_n\}$ be instance set in the training data set;
$Y = \{y_1, y_2, \ldots, y_n\}^T$ be label vector for X;
c be the number of classes;
Output: Decision trees $T_1, T_2, \ldots, T_c$

for $i = 1$ to c **do**
  Compute the binary class label for ith class $Y'_i = (y'_{i1}, y'_{i2}, \ldots, y'_{in})^T$ according to Definition 1;
  $T_i = CreatTree(X, Y'_i)$;
end for
procedure $CreatTree(X, Y'_i)$
  Create a node N for instance set X;
  if satisfying the stop condition of the selected decision tree algorithm **then**
    // *Any existing decision tree algorithm might be selected, such as ID3, C4.5, CART, TEIM, etc.*
    return N as a leaf node with the membership vector MS_N by using Y according to Definition 2;
  else
    splitting the node N to t child nodes $N_1, N_2, \ldots, N_t$ based on the selected decision tree algorithm,
      where Y'_i is used as the decision attribute;
    for $j = 1$ to t **do**
      Let $X[N_j]$ be instance subset corresponding to N_j,
        and $Y'_i[N_j]$ be the subset of Y'_i corresponding to $X[N_j]$;
      attach the node returned by CreatTree($X[N_j]$, $Y'_i[N_j]$) to node N;
    end for
  end if
  return N as the root node of (X, Y'_i).
end procedure

Table 5
Performance of algorithms with CART as base classifier.

Dataset	Accuracy			F1-measures			Precision			AUC		
	CART	OVA-CART	MVDT-CART	CART	OVA-CART	MVDT-CART	CART	OVA-CART	MVDT-CART	CART	OVA-CART	MVDT-CART
iris	0.9340 (1)	0.9333 (2)	0.9320 (3)	0.9274 (1)	0.9247 (3)	0.9252 (2)	0.9379 (1)	0.9345 (3)	0.9364 (2)	0.9685 (1)	0.9596 (3)	0.9653 (2)
texture	0.9389 (2)	0.9181 (3)	0.9409 (1)	0.9385 (2)	0.9196 (3)	0.9399 (1)	0.9389 (2)	0.9181 (3)	0.9409 (1)	0.9726 (3)	0.9922 (2)	0.9956 (1)
mfeat-fac	0.8894 (2)	0.8609 (3)	0.8914 (1)	0.8865 (2)	0.8668 (3)	0.8880 (1)	0.8896 (3)	0.8968 (1)	0.8904 (2)	0.9405 (3)	0.9763 (2)	0.9820 (1)
mfeat-fou	0.7591 (2)	0.6951 (3)	0.7882 (1)	0.7560 (2)	0.6936 (3)	0.7838 (1)	0.7617 (2)	0.7388 (3)	0.7885 (1)	0.8868 (3)	0.9316 (2)	0.9674 (1)
mfeat-kar	0.8399 (2)	0.8090 (3)	0.8787 (1)	0.8372 (2)	0.8143 (3)	0.8758 (1)	0.8413 (3)	0.8556 (2)	0.8791 (1)	0.9191 (3)	0.9656 (2)	0.9840 (1)
mfeat-mor	0.6498 (2)	0.6000 (3)	0.6785 (1)	0.6455 (2)	0.5876 (3)	0.6714 (1)	0.6500 (2)	0.6117 (3)	0.6747 (1)	0.8397 (3)	0.8709 (2)	0.9492 (1)
mfeat-pix	0.8740 (2)	0.8478 (3)	0.8927 (1)	0.8715 (2)	0.8550 (3)	0.8897 (1)	0.8750 (3)	0.8906 (2)	0.8929 (1)	0.9357 (3)	0.9762 (2)	0.9871 (1)
mfeat-zer	0.6858 (2)	0.6430 (3)	0.7198 (1)	0.6835 (2)	0.6376 (3)	0.7139 (1)	0.6904 (2)	0.6729 (3)	0.7168 (1)	0.8721 (3)	0.9171 (2)	0.9603 (1)
wavform	0.7559 (2)	0.7426 (3)	0.7641 (1)	0.7557 (2)	0.7403 (3)	0.7613 (1)	0.7566 (3)	0.7647 (2)	0.7686 (1)	0.8244 (3)	0.8876 (2)	0.9057 (1)
optdigits	0.9045 (2)	0.8718 (3)	0.9196 (1)	0.9039 (2)	0.8785 (3)	0.9187 (1)	0.9049 (3)	0.9062 (2)	0.9196 (1)	0.9464 (3)	0.9810 (2)	0.9922 (1)
tae	0.6472 (2)	0.6439 (3)	0.6530 (1)	0.6291 (2)	0.6240 (3)	0.6335 (1)	0.6570 (3)	0.6655 (1)	0.6611 (2)	0.7395 (3)	0.7627 (2)	0.7838 (1)
letter	0.8765 (2)	0.8357 (3)	0.8887 (1)	0.8757 (2)	0.8516 (3)	0.8883 (1)	0.8768 (3)	0.8878 (2)	0.8943 (1)	0.9321 (3)	0.9710 (2)	0.9896 (1)
wine	0.9115 (1)	0.8933 (3)	0.9053 (2)	0.9092 (1)	0.8861 (3)	0.8968 (2)	0.9140 (1)	0.9103 (3)	0.9115 (2)	0.9407 (3)	0.9816 (1)	0.9774 (2)
cmc	0.4838 (2)	0.4828 (3)	0.5056 (1)	0.4590 (1)	0.4284 (3)	0.4753 (2)	0.4614 (2)	0.4504 (3)	0.4805 (1)	0.6150 (3)	0.6400 (2)	0.6834 (1)
molecula	0.8964 (3)	0.8979 (2)	0.9269 (1)	0.8838 (3)	0.8884 (2)	0.9182 (1)	0.8818 (3)	0.8955 (2)	0.9215 (1)	0.9157 (3)	0.9678 (2)	0.9779 (1)
satimage	0.8562 (2)	0.8316 (3)	0.8749 (1)	0.8325 (2)	0.8140 (3)	0.8440 (1)	0.8328 (3)	0.8352 (2)	0.8673 (1)	0.9211 (3)	0.9532 (2)	0.9766 (1)
balance-scale	0.7750 (2)	0.7424 (3)	0.8027 (1)	0.5734 (2)	0.5721 (3)	0.5810 (1)	0.5888 (2)	0.6030 (1)	0.5830 (3)	0.8380 (3)	0.8680 (2)	0.8818 (1)
glass	0.6756 (1)	0.6458 (3)	0.6574 (2)	0.5903 (1)	0.5435 (2)	0.5049 (3)	0.6227 (1)	0.5897 (2)	0.5255 (3)	0.8825 (1)	0.8333 (2)	0.7740 (3)
car	0.9849 (1)	0.9728 (3)	0.9769 (2)	0.9632 (1)	0.9138 (3)	0.9186 (2)	0.9683 (1)	0.9481 (2)	0.9350 (3)	0.9777 (3)	0.9840 (1)	0.9808 (2)
ecoli	0.8190 (2)	0.8116 (3)	0.8508 (1)	0.6224 (3)	0.6255 (2)	0.6899 (1)	0.6462 (3)	0.6610 (2)	0.7314 (1)	0.8604 (3)	0.8833 (2)	0.9427 (1)
yeast	0.5145 (2)	0.5014 (3)	0.5581 (1)	0.3985 (2)	0.3739 (3)	0.4165 (1)	0.4137 (3)	0.4381 (2)	0.4511 (1)	0.7043 (3)	0.7610 (2)	0.8457 (1)
nursery	0.9985 (1)	0.9684 (3)	0.9984 (2)	0.9885 (1)	0.9396 (3)	0.9765 (2)	0.9886 (1)	0.9380 (3)	0.9765 (2)	0.9928 (2)	0.9840 (3)	0.9937 (1)
Avg.Rank	1.818	2.909	1.273	1.818	2.864	1.318	2.273	2.227	1.500	2.773	2.000	1.227
Friedman $\alpha = 0.05$	✓	✓	–	✓	✓	–	✓	✓	–	✓	✓	–

the instances in v as class ω_i. If a leaf node v of T_i outcomes negative, then we show the membership of the instances in v belonging to the other $c-1$ classes.

We can obtain the final membership of an instance x by following steps:

(1) Given an instance x, we can obtain the membership vector $MS^{(i)}(x) = \left(s_1^{(i)}(x), s_2^{(i)}(x), \ldots, s_c^{(i)}(x)\right)$ for the instance x respective to T_i by testing the attribute values of x gradually from the root node of T_i to a leaf node v. $MS^{(i)}(x)$ equals to the membership vector of a leaf node v in T_i, i.e., $MS^{(i)}(x) = MS_v^{(T_i)}$. Hence, $s_k^{(i)}(x) = s_{vk}^{(T_i)}$ $(1 \le k \le c)$ can be regarded as the probability of x belonging to kth class $(k = 1, 2, \ldots, c)$ determined by the classifier T_i ;

(2) We can obtain the probability of x belonging to the kth class ω_k by combining the probability of x belonging to kth class $(k = 1, 2, \ldots, c)$ determined by each classifier $T_1, T_2, \ldots, T_c$. Let $P(x) = (p_1(x), p_2(x), \ldots, p_c(x))$ be the final membership vector of x, where

$$p_k(x) = \frac{1}{c}\sum_{i=1}^{c} s_k^{(i)}(x) \quad (k = 1, 2, \ldots, c) \tag{3}$$

(3) Give x the class label ω_t, where

$$t = \arg\max\{p_k(x), 1 \le k \le c\} \tag{4}$$

Algorithm 2 shows the frame of proposed Multi-view OVA model based on decision tree.

We can choose any existing decision tree algorithm (such as ID3, C4.5, CART etc.) as the base classifier in MVDT, and the resulting models are simply named by MVDT-ID3, MVDT-C4.5, MVDT-CART respectively.

3.3. Examples

Next we will explain the working mechanism of the MVDT by introducing an illustrative example. We generate an artificial data set with 13 instances, 3 classes($\{\omega_1, \omega_2, \omega_3\}$) and 4 attributes$\{a_1, a_2, a_3, a_4\}$, as shown in Table 1. Instances x_1 to x_{11} are treated as the training set of constructing a decision tree, x_{12} and x_{13} are taken as the test set of evaluating the performance of a decision tree. We choose ID3 as base classifier.

Algorithm 2 MVDT algorithm.

Input: Training set $\{(X_1, y_1), (X_2, y_2), \ldots, (X_n, y_n)\}$, the class number c, and an instance $x \in \mathbb{R}^m$;

Output: the class label of x.

1: Generating c decision trees $\{T_1, T_2, \ldots, T_c\}$ as base classifiers by Algorithm 1.
2: **for** $i = 1$ to c **do**
3: Obtain the membership vector $MS^{(i)}(x) = \left(s_1^{(i)}(x), s_2^{(i)}(x), \ldots, s_c^{(i)}(x)\right)$ for the instance x respective to T_i;
4: **end for**
5: Calculate the final membership vector $P(x) = (p_1(x), p_2(x), \ldots, p_c(x))$ of x;
6: Return ω_t as the class label of x, where $t = \arg\max\{p_k(x), 1 \le k \le c\}$.

By Algorithm 1, MVDT-ID3 will establish three decision trees $\{T_1, T_2, T_3\}$ as base classifiers and output membership vectors (see Definition 2) of the leaf nodes respective to the related tree. The decision trees $\{T_1, T_2, T_3\}$ and the membership vectors of leaves obtained by MVDT-ID3 are shown in Fig. 1.

Fig. 2 shows the decision tree which applying ID3 algorithm directly to the training data in Table 1.

In what follows, we consider the class label of each instance in the test set. The decisions induced by *ID3* and those induced by the proposed method MVDT-ID3 in this paper are listed in Table 2.

Through computing, we have that:

(1) For x_{12}, T_1 outputs $MS^{(1)}(x_{12}) = (1, 0, 0)$, T_2 outputs $MS^{(2)}(x_{12}) = (0.75, 0, 0.25)$, and T_3 outputs $MS^{(3)}(x_{12}) = (0, 0, 1)$ by Algorithm 1. The final membership vector of x_{12} obtained by MVDT-ID3 is $P(x_{12}) = (0.58, 0, 0.42)$, where $p_1(x_{12}) = (1 + 0.75 + 0)/3 = 0.58$, $p_2(x_{12}) = (0 + 0 + 0)/3 =$

Table 6
Performance analysis of algorithms with TEIM as base classifier.

Dataset	Accuracy			F1-measures			Precision			AUC		
	TEIM	OVA-TEIM	MVDT-TEIM	TEIM	OVA-TEIM	MVDT-TEIM	TEIM	OVA-TEIM	MVDT-TEIM	TEIM	OVA-TEIM	MVDT-TEIM
iris	0.9533 (1)	0.9433 (3)	0.9487 (2)	0.9475 (1)	0.9348 (3)	0.9432 (2)	0.9538 (2)	0.9418 (3)	0.9483 (1)	0.9848 (3)	0.9802 (2)	0.9816 (1)
texture	0.9467 (1)	0.9273 (3)	0.9401 (2)	0.9461 (1)	0.9288 (3)	0.9394 (2)	0.9467 (1)	0.9381 (3)	0.9410 (2)	0.9994 (1)	0.9919 (3)	0.9958 (2)
mfeat-fac	0.8946 (2)	0.8677 (3)	0.9150 (1)	0.8930 (2)	0.8733 (3)	0.9128 (1)	0.8960 (3)	0.9021 (2)	0.9158 (1)	0.9443 (3)	0.9786 (2)	0.9877 (1)
mfeat-fou	0.7412 (2)	0.6704 (3)	0.7735 (1)	0.7366 (2)	0.6713 (3)	0.7668 (1)	0.7413 (2)	0.7235 (3)	0.7722 (1)	0.8815 (3)	0.9210 (2)	0.9614 (1)
mfeat-kar	0.8372 (2)	0.8024 (3)	0.8854 (1)	0.8337 (2)	0.8078 (3)	0.8820 (1)	0.8383 (3)	0.8521 (2)	0.8864 (1)	0.9277 (3)	0.9642 (2)	0.9854 (1)
mfeat-mor	0.6556 (2)	0.5897 (3)	0.6835 (1)	0.6490 (2)	0.5792 (3)	0.6753 (1)	0.6543 (2)	0.6167 (3)	0.6802 (1)	0.8737 (2)	0.8730 (3)	0.9509 (1)
mfeat-pix	0.8917 (2)	0.8681 (3)	0.9194 (1)	0.8897 (2)	0.8741 (3)	0.9175 (1)	0.8926 (3)	0.9044 (2)	0.9204 (1)	0.9462 (3)	0.9811 (2)	0.9902 (1)
mfeat-zer	0.6918 (2)	0.6609 (3)	0.7313 (1)	0.6853 (2)	0.6569 (3)	0.7242 (1)	0.6883 (3)	0.6924 (2)	0.7266 (1)	0.8881 (3)	0.9225 (2)	0.9650 (1)
wavform	0.7622 (2)	0.7314 (3)	0.7743 (1)	0.7618 (2)	0.7301 (3)	0.7738 (1)	0.7625 (2)	0.7599 (3)	0.7770 (1)	0.8546 (3)	0.8809 (2)	0.9210 (1)
optdigits	0.9114 (2)	0.8987 (3)	0.9391 (1)	0.9110 (2)	0.9041 (3)	0.9385 (1)	0.9121 (3)	0.9245 (2)	0.9393 (1)	0.9539 (3)	0.9865 (2)	0.9947 (1)
tae	0.6055 (2)	0.5828 (3)	0.6315 (1)	0.5817 (2)	0.5643 (3)	0.6130 (1)	0.6092 (3)	0.6227 (2)	0.6414 (1)	0.7640 (2)	0.7363 (3)	0.7872 (1)
letter	0.8795 (2)	0.8495 (3)	0.8951 (1)	0.8789 (2)	0.8657 (3)	0.8950 (1)	0.8802 (3)	0.9014 (1)	0.9006 (2)	0.9425 (3)	0.9728 (2)	0.9915 (1)
wine	0.9544 (1)	0.9179 (3)	0.9441 (2)	0.9520 (1)	0.9131 (3)	0.9427 (2)	0.9587 (1)	0.9308 (3)	0.9479 (2)	0.9989 (1)	0.9854 (3)	0.9936 (2)
cmc	0.4787 (3)	0.4856 (2)	0.5085 (1)	0.4510 (2)	0.4224 (3)	0.4776 (1)	0.4556 (3)	0.4631 (2)	0.4851 (1)	0.6438 (3)	0.6467 (2)	0.6889 (1)
molecula	0.9176 (2)	0.9088 (3)	0.9255 (1)	0.9080 (2)	0.8991 (3)	0.9162 (1)	0.9069 (2)	0.9021 (3)	0.9188 (1)	0.9363 (3)	0.9668 (2)	0.9759 (1)
satimage	0.8590 (2)	0.8309 (3)	0.8729 (1)	0.8345 (2)	0.8131 (3)	0.8398 (1)	0.8354 (3)	0.8359 (2)	0.8691 (1)	0.9282 (3)	0.9535 (2)	0.9758 (1)
balance-scale	0.7851 (2)	0.7392 (3)	0.8210 (1)	0.6231 (1)	0.6037 (2)	0.5853 (3)	0.6366 (1)	0.6305 (2)	0.5750 (3)	0.8762 (3)	0.8779 (2)	0.9114 (1)
glass	0.7024 (1)	0.6448 (2)	0.6439 (3)	0.6125 (1)	0.5442 (2)	0.4973 (3)	0.6341 (1)	0.5986 (2)	0.5202 (3)	0.8735 (1)	0.8282 (2)	0.8131 (3)
car	0.9710 (1)	0.9335 (3)	0.9555 (2)	0.9179 (1)	0.7799 (3)	0.8635 (2)	0.9339 (1)	0.8690 (3)	0.8951 (2)	0.9828 (1)	0.9444 (3)	0.9654 (2)
ecoli	0.8209 (2)	0.8051 (3)	0.8322 (1)	0.6113 (3)	0.6318 (2)	0.6474 (1)	0.6429 (3)	0.6730 (2)	0.6864 (1)	0.9287 (2)	0.8765 (3)	0.9455 (1)
yeast	0.5196 (2)	0.4943 (3)	0.5742 (1)	0.4328 (2)	0.3828 (3)	0.4448 (1)	0.4582 (3)	0.4585 (2)	0.4787 (1)	0.7384 (3)	0.7521 (2)	0.8554 (1)
nursery	0.9958 (2)	0.9844 (3)	0.9959 (1)	0.8811 (2)	0.8020 (3)	0.9566 (1)	0.8782 (2)	0.8033 (3)	0.9578 (1)	0.9930 (2)	0.9574 (3)	0.9977 (1)
Avg.Rank	1.818	2.909	1.273	1.773	2.864	1.364	2.273	2.364	1.364	2.455	2.318	1.227
Friedman $\alpha = 0.05$	✓	✓	–	✓	✓	–	✓	✓	–	✓	✓	–

Table 7
Performance analysis of algorithms with SCDT as base classifier.

Dataset	Accuracy			F1-measures			Precision			AUC		
	SCDT	OVA-SCDT	MVDT-SCDT	SCDT	OVA-SCDT	MVDT-SCDT	SCDT	OVA-SCDT	MVDT-SCDT	SCDT	OVA-SCDT	MVDT-SCDT
iris	0.9475 (1)	0.9383 (3)	0.9408 (2)	0.9423 (1)	0.9327 (3)	0.9352 (2)	0.9464 (1)	0.9380 (3)	0.9401 (2)	0.9782 (1)	0.9735 (3)	0.9780 (2)
texture	0.9311 (2)	0.9067 (3)	0.9321 (1)	0.9306 (2)	0.9087 (3)	0.9310 (1)	0.9313 (2)	0.9218 (3)	0.9319 (1)	0.9694 (3)	0.9903 (2)	0.9949 (1)
mfeat-fac	0.8723 (1)	0.7924 (3)	0.8482 (2)	0.8703 (1)	0.8017 (3)	0.8453 (2)	0.8740 (1)	0.8472 (3)	0.8494 (2)	0.9757 (1)	0.9298 (3)	0.9619 (2)
mfeat-fou	0.7250 (2)	0.6671 (3)	0.7704 (1)	0.7221 (2)	0.6652 (3)	0.7656 (1)	0.7274 (2)	0.7130 (3)	0.7705 (1)	0.8735 (3)	0.9195 (2)	0.9628 (1)
mfeat-kar	0.8056 (2)	0.7739 (3)	0.8657 (1)	0.8026 (2)	0.7809 (3)	0.8626 (1)	0.8088 (1)	0.8306 (3)	0.8664 (2)	0.9006 (3)	0.9573 (2)	0.9821 (1)
mfeat-mor	0.6400 (2)	0.5739 (3)	0.6686 (1)	0.6356 (2)	0.5622 (3)	0.6604 (1)	0.6401 (2)	0.5898 (3)	0.6650 (1)	0.8334 (3)	0.8638 (2)	0.9511 (1)
mfeat-pix	0.8736 (2)	0.8478 (3)	0.8929 (1)	0.8718 (2)	0.8539 (3)	0.8905 (1)	0.8775 (3)	0.8863 (2)	0.8941 (1)	0.9296 (3)	0.9738 (2)	0.9858 (1)
mfeat-zer	0.6750 (2)	0.6396 (3)	0.7096 (1)	0.6724 (2)	0.6360 (3)	0.7032 (1)	0.6801 (2)	0.6719 (3)	0.7056 (1)	0.8624 (3)	0.9173 (2)	0.9611 (1)
wavform	0.7514 (2)	0.7358 (3)	0.7642 (1)	0.7512 (2)	0.7336 (3)	0.7610 (1)	0.7521 (3)	0.7607 (2)	0.7695 (1)	0.8236 (3)	0.8845 (2)	0.9065 (1)
optdigits	0.8847 (2)	0.8651 (3)	0.9194 (1)	0.8842 (2)	0.8717 (3)	0.9183 (1)	0.8854 (3)	0.9008 (2)	0.9193 (1)	0.9373 (3)	0.9803 (2)	0.9918 (1)
tae	0.5751 (3)	0.6091 (2)	0.6267 (1)	0.5474 (3)	0.5826 (2)	0.5982 (1)	0.5785 (3)	0.6268 (1)	0.6266 (2)	0.7083 (3)	0.7305 (2)	0.7732 (1)
letter	0.8634 (2)	0.8248 (3)	0.8879 (1)	0.8625 (2)	0.8411 (3)	0.8877 (1)	0.8636 (3)	0.8784 (2)	0.8936 (1)	0.9257 (3)	0.9677 (2)	0.9897 (1)
wine	0.9070 (1)	0.9058 (2)	0.8996 (3)	0.8955 (2)	0.8971 (1)	0.8916 (3)	0.9061 (3)	0.9107 (1)	0.9067 (2)	0.9373 (3)	0.9668 (1)	0.9578 (2)
cmc	0.4784 (3)	0.4795 (2)	0.4942 (1)	0.4556 (2)	0.4272 (3)	0.4654 (1)	0.4597 (2)	0.4462 (3)	0.4706 (1)	0.6089 (3)	0.6365 (2)	0.6749 (1)
molecula	0.9171 (2)	0.9087 (3)	0.9291 (1)	0.9076 (2)	0.8990 (3)	0.9203 (1)	0.9074 (2)	0.9051 (3)	0.9244 (1)	0.9311 (3)	0.9694 (2)	0.9784 (1)
satimage	0.8468 (2)	0.8242 (3)	0.8707 (1)	0.8214 (2)	0.8062 (3)	0.8375 (1)	0.8220 (3)	0.8268 (2)	0.8603 (1)	0.9155 (3)	0.9497 (2)	0.9745 (1)
balance-scale	0.7842 (2)	0.7450 (3)	0.8010 (1)	0.5824 (2)	0.5753 (3)	0.5829 (1)	0.5961 (2)	0.6063 (1)	0.5871 (3)	0.8452 (3)	0.8733 (2)	0.8933 (1)
glass	0.6226 (2)	0.6102 (3)	0.6444 (1)	0.5466 (1)	0.5222 (2)	0.5054 (3)	0.5684 (1)	0.5601 (2)	0.5167 (3)	0.7434 (3)	0.7917 (2)	0.8729 (1)
car	0.9685 (3)	0.9725 (1)	0.9720 (2)	0.9177 (2)	0.9173 (3)	0.9215 (1)	0.9507 (2)	0.9550 (1)	0.9471 (3)	0.9746 (3)	0.9861 (1)	0.9811 (2)
ecoli	0.8252 (2)	0.8140 (3)	0.8513 (1)	0.6627 (2)	0.6471 (3)	0.7140 (1)	0.7021 (2)	0.6781 (3)	0.7553 (1)	0.9344 (2)	0.8784 (3)	0.9444 (1)
yeast	0.4919 (2)	0.4733 (3)	0.5441 (1)	0.3999 (2)	0.3636 (3)	0.4206 (1)	0.4201 (3)	0.4343 (2)	0.4763 (1)	0.6960 (3)	0.7330 (2)	0.8463 (1)
nursery	0.9982 (1)	0.9731 (3)	0.9981 (2)	0.9868 (1)	0.9042 (3)	0.9703 (2)	0.9861 (1)	0.9042 (3)	0.9703 (2)	0.9912 (2)	0.9761 (3)	0.9942 (1)
Avg.Rank	1.955	2.773	1.273	1.864	2.818	1.318	2.136	2.318	1.545	2.727	2.091	1.182
Friedman $\alpha = 0.05$	✓	✓	–	✓	✓	–	✓	✓	–	✓	✓	–

0 and $p_3((x_{12}) = (0 + 0.25 + 1)/3 = 0.42$. Hence, x_{12} is classified as class ω_1.

(2) For x_{13}, T_1 outputs $MS^{(1)}(x_{13}) = (0, 0.67, 0.33)$, T_2 outputs $MS^{(2)}(x_{13}) = (0, 1, 0)$, and T_3 outputs $MS^{(3)}(x_{13}) = (0.2, 0.8, 0)$ by Algorithm 1. The final membership vector of x_{13} obtained by MVDT-ID3 is $P(x_{13}) = (0.07, 0.82, 0.11)$. Hence, x_{13} is classified as class ω_2.

It is clear that the corresponding class labels of test instances induced by MVDT-ID3 are equivalent to their real class labels. However, the class labels for x_{12} and x_{13} induced by ID3 are ω_3 and ω_2, respectively, which does not coincide with the fact.

4. Experimental analysis

In this section, we provide experimental analysis of the proposed MVDT algorithm. The performance enhancement is exhibited in Section 4.3. The robustness of MVDT algorithm to output noise is demonstrated in Section 4.4.

4.1. Data sets

We employed 22 data sets from UCI repository of machine learning data sets including some widely used imbalance data sets (such as satimage, glass, car, etc.) to test the performance of the

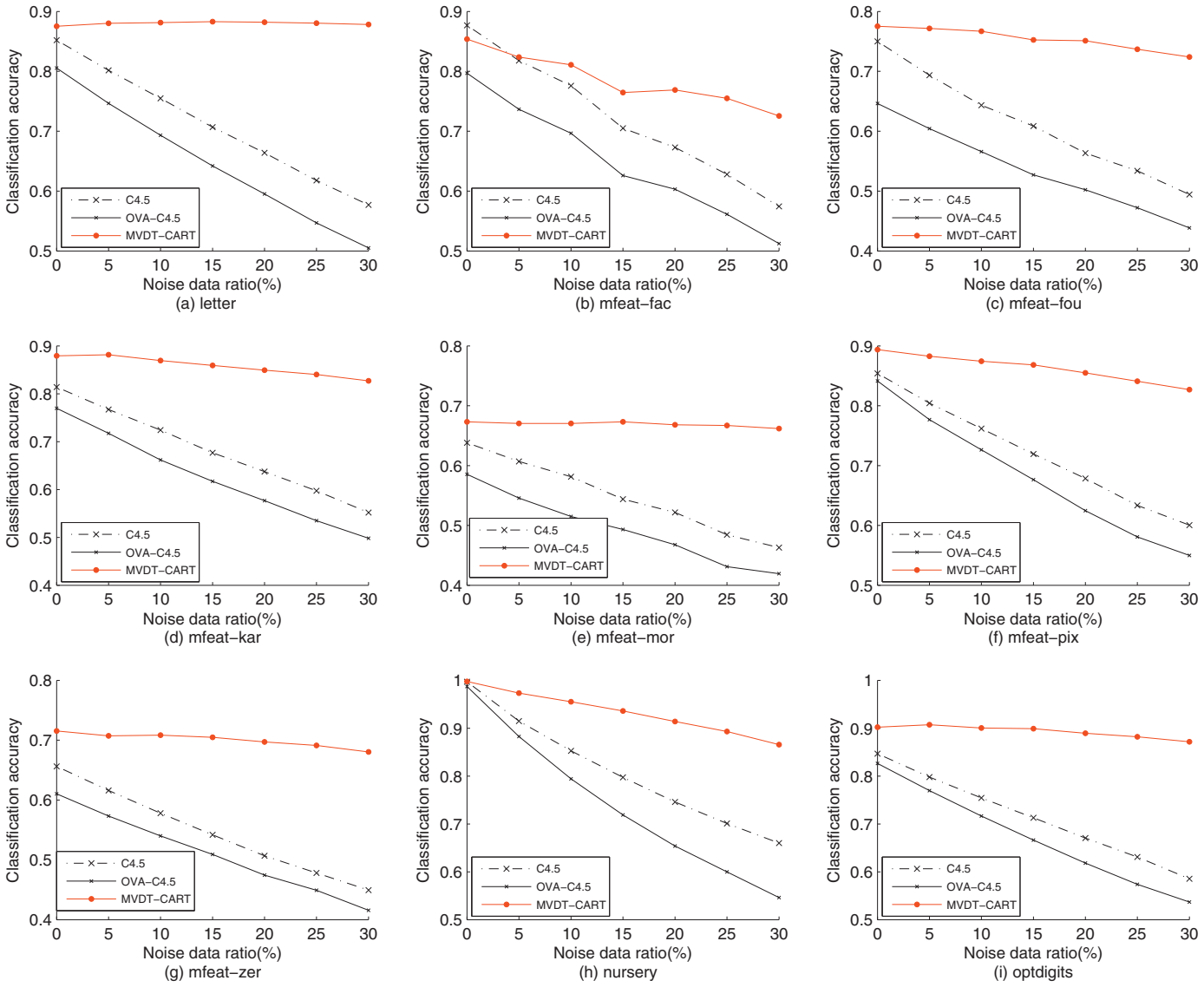

Fig. 3. Accuracy changes of C4.5, OVA-C4.5 and MVDT-C4.5 on noise ratio.

proposed MVDT algorithm. Essential information about the 22 data sets are shown in Table 3, where the number of examples (#Ex.), the number of attributes (#Atts.), the number of classes (#Cl.) and the imbalance ratio (IR) are listed for each data set. The data sets are exhibited in non-decreasing order according to imbalance ratio, which is computed as the proportion of the number of majority class examples to the number of minority class examples [39]. All these data sets can be downloaded from http://archive.ics.uci.edu/ml.

4.2. Evaluation metric

In order to compare the performance of different classifiers, we employ Accuracy, Precision, F1-measure, Area Under the ROC Curve(AUC) to evaluate the effectiveness of the trees. Accuracy is the most commonly used evaluation metric, which is calculated by the percentage of successful predictions. Another popular evaluation metric is *F*-measure. *F*-measure is a class of measures which has been considered as the harmonic mean of the precision and recall of a classifier. In this paper, we consider the *F*1-measure, where equal importance is given to both precision and recall. *F*1-measure is defined as Eq.(5), where P is precision and R is recall.

$$F1 = \frac{2PR}{(P+R)} \tag{5}$$

Area Under the ROC Curve (AUC) measures the probability of ranking a random positive class example over a random negative class example. For a multi-class data set, we average *AUC* over all pairs of classes [40], which is defined as:

$$AUC = \frac{2}{c(c-1)} \sum_{i<j} AUC(i,j) \quad (1 \le i, j \le c) \tag{6}$$

where c is the number of classes and $AUC(i, j)$ is the *AUC* of class i and j.

4.3. The MVDT model performance analysis

In this section, we will compare the classification Accuracy, Precision, F1-measure, AUC of MVDT with the other methods to show the effectiveness of MVDT. As a general framework, MVDT algorithm can use any existing decision tree model as base classifier. We choose 5 classical decision tree models, such as C4.5, CART, TEIM, SCDT and NBTree to evaluate the performance of our MVDT algorithm on different base classifiers.

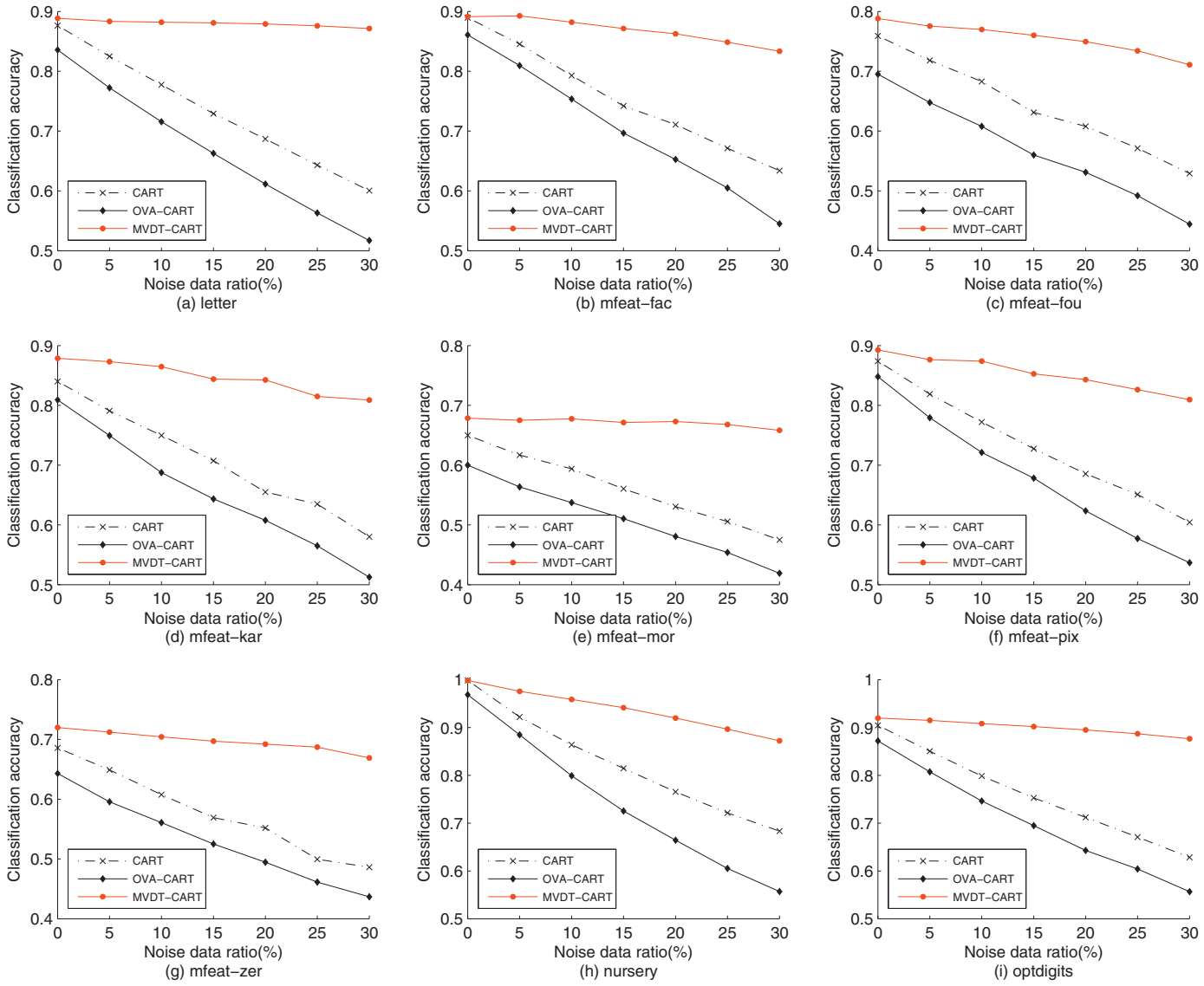

Fig. 4. Accuracy changes of CART, OVA-CART and MVDT-CART on noise ratio.

All of the algorithms are implemented in Matlab. In order to verify the validity of MVDT model, we perform five group of contrast experiments. Each group compares the performance of one base classifier, OVA based on the base classifier and MVDT model based on the base classifier. To ensure the stability of the experimental results, 10-fold cross-validation technique is performed on 22 data sets. We repeat the experiments 10 times on each data set.

Demšar [41] suggests that the best way to consider the performance of multiple classifiers across multiple data sets is through a comparative analysis of averaged performance ranks. We follow this recommendation and rank the performance of each classifier, where rank 1 denotes the best method. Since we seek to determine whether or not the MVDT model are statistically significantly better than the other methods, the Friedman test is then used to compare the ranks at 95% confidence interval as recommended by Demšar [41]. The relative ranking for each classifier is indicated in parenthesis with the average rank applied for all ties. Using the Friedman test for comparing the ranking across all the 22 data sets and 3 classifiers, a ✓in the bottom row indicates that MVDT statistically significantly improved over that classifier.

The first group uses C4.5 as the base classifier, Table 4 reports the performance comparisons on Accuracy, F1-measure, Precision and AUC of MVDT model against other algorithms on various data sets, respectively. It can be easily seen that our MVDT-C4.5 statistically significantly improved over the C4.5 and OVA-C4.5 on these evaluation metrics.

The second group uses CART as the base classifier, the experimental results of each evaluation metric are shown in Table 5. It can be easily seen that our MVDT-CART statistically significantly improved over the CART and OVA-CART on all evaluation metrics.

The third group uses TEIM as the base classifier, the experimental results are shown in Table 6. It can be seen that MVDT-TEIM statistically significantly improved over TEIM and OVA-TEIM on all evaluation metrics.

The fourth group uses SCDT as the base classifier, the experimental results are shown in Table 7. In SCDT, the number of leaf nodes is constrained to 80%**TotalMaxleaves*, where *TotalMaxleaves* is the number of leaf nodes produced by C4.5. It can be seen that MVDT-SCDT statistically significantly improved over SCDT and OVA-SCDT on all evaluation metrics.

The fifth group uses NBTree as the base classifier, and the experimental results are shown in Table 8. In NBTree [42], the data is

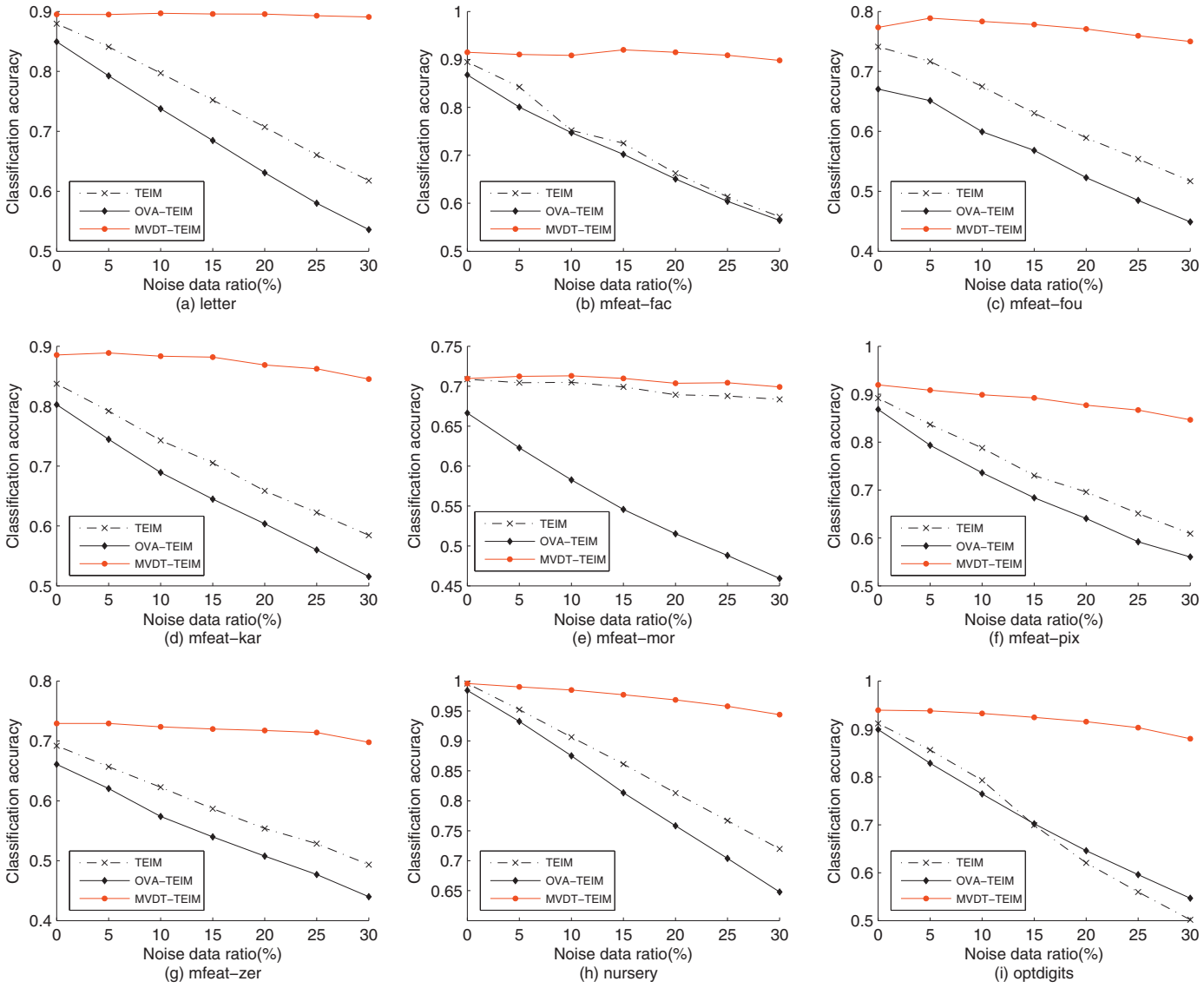

Fig. 5. Accuracy changes of TEIM, OVA-TEIM and MVDT-TEIM on noise ratio.

pre-discretized using an entropy-based algorithm [43] and the leaf size is set to 30. What's more, since MMNBTree [36] is a successful improvement of NBTree using OVA strategy, we compare our MVDT-NBTree with MMNBTree instead of NBTree in Table 8. It is shown that our MVDT-NBTree outperforms over the OVA-NBTree and MMNBTree on Accuracy, Precision and AUC but slightly lower than all algorithms except OVA-NBTree on F1-measure.

Based on the experimental results, it is easy to draw the conclusion that for multi-class classification tasks the classification performance of the new proposed MVDT algorithms based on C4.5, CATR, TEIM, SCDT and NBTree respectively(i.e., MVDT-C4.5, MVDT-CART, MVDT-TEIM MVDT-SCDT and MVDT-NBTree) are significantly higher than that of the base algorithms and OVA methods. Furthermore, since we build c decision trees for c classes including minority classes in MVDT model and the decision tree corresponding to a minority class could provide a relatively accurate prediction for the class, our MVDT shows a significant performance on imbalance data sets, as is shown in Tables 4–8.

4.4. The effects of output noise data

In MVDT, a c-class classification task is divided to c sub-tasks, each decision tree is corresponding to a sub-task respective to the class discussed. When there is noise in training set, only a few of decision trees might be sensitive to noise data and the accuracy of these decision trees might be reduced. Most of decision trees would be slightly affected by noise data, however, the combination result of these trees could still maintain a higher prediction accuracy. Hence, our algorithm is robust to data set with output noise. In order to test the robustness of the proposed algorithm, the following experiments are done.

For each data set in the experiment, 10% instances at random is split off as a test set. Two rounds are made on the remaining training set. The first round is on the training set. The second round is on a noisy version of the training set. The noisy version is gotten by changing 5% of the class labels randomly into an alternate class label chosen uniformly from the other labels. This is repeated 10 times 10-fold cross-validation using C4.5, CART, TEIM, SCDT, MVDT-C4.5, MVDT-CART, MVDT-TEIM and MVDT-SCDT. Experiment results

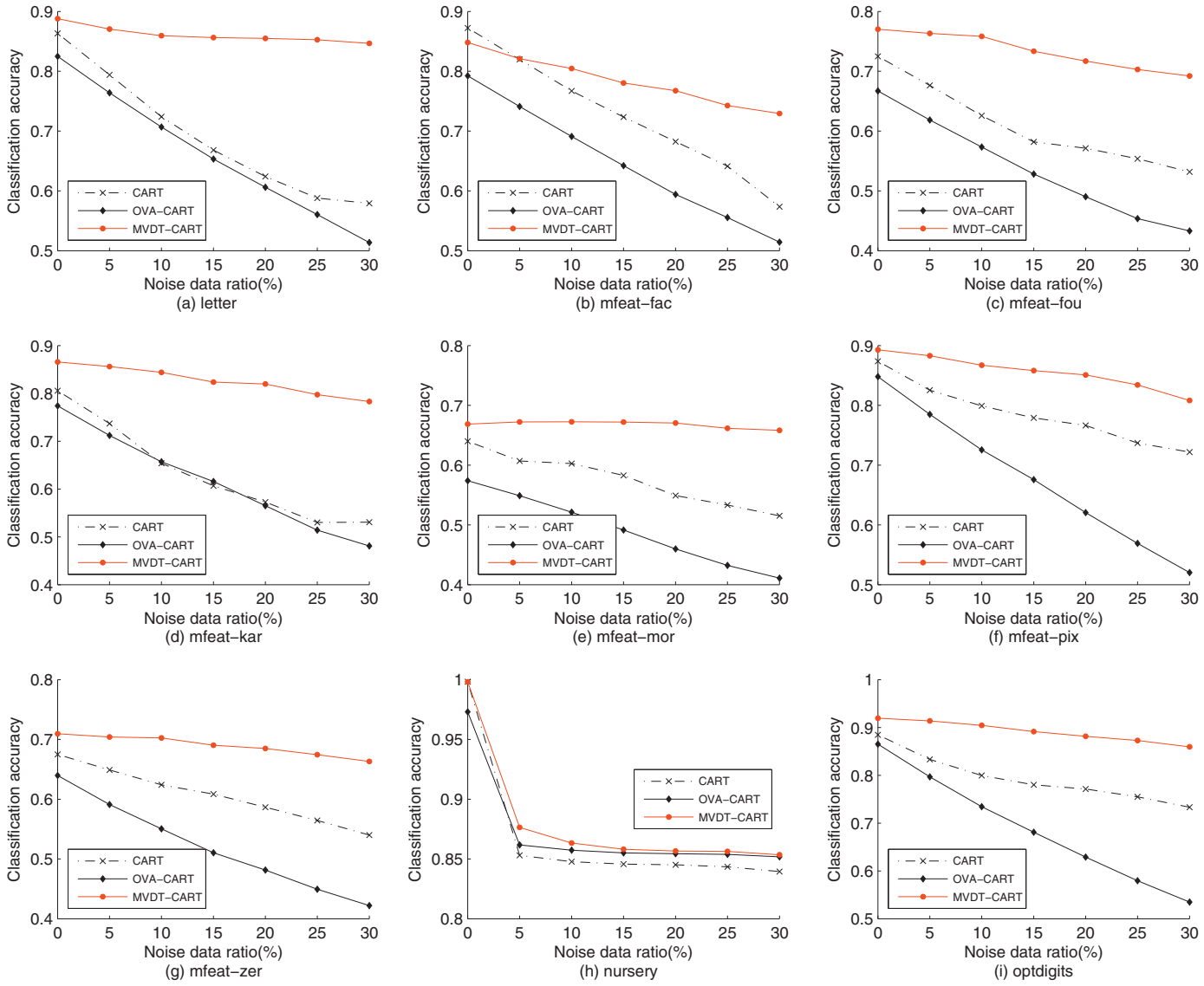

Fig. 6. Accuracy changes of SCDT, OVA-SCDT and MVDT-SCDT on noise ratio.

on accuracy rates in the 5% noise are exhibited in Table 9. After adding 5% noise data, the average accuracy of all algorithms will be reduced. Our MVDT algorithm is the most robust to output noise.

Furthermore, we gradually increase output noise rate to the training set at 10%, 15%, 20%, 25%, 30% respectively. These are also repeated 10 times 10-fold cross-validation. Figs. 3–6 exhibit the effect of output noise on the accuracy of classification. It can be seen that the classification accuracy of all algorithms has decreased with the increase of noise rate, while our algorithm MVDT still shows excellent robustness to output noise.

4.5. The effects of experts' perception

Experts' perception might provide effective guidance in decision process by defining important splitting attributes in decision trees. In [23] a new decision tree(AGDT) is proposed to select branching variables at the primary levels according to the experts' perception, and is used to compare different feasible solutions for routing, scheduling and assignment of drivers to carriers and roads for Hazmat transportation. We also try to introduce experts' perception into the MVDT model to build base decision trees. We use "Wife's age" as the first splitting attribute in the construction of decision trees in "cmc" data set, "Mg" in "glass" data set, "Course" in "tae" data set and "gvh" in "yeast" data set. The experimental results show that if the effective expert experience could be introduced in MVDT model, the classification performance might be improved, as shown in Table 10.

5. Conclusion

Multi-class classification problems exist widely in knowledge discovery and pattern recognition. And decision tree is a useful tool to deal with multi-class classification problems. The robustness to noise data and generalization are the major issues of decision trees.

In order to improve the robustness to noise and the generalization of classical decision tree, we propose a multi-view OVA model based on decision trees (MVDT) in this paper. MVDT introduces "Divide and Conquer" into solving multi-class classification problems, in which we first divide a multi-class classification task into c multiple parallel sub-tasks and build c decision trees for these sub-tasks. We define a membership vector for leaf nodes in a decision tree to represent the membership probabilities of the instance belonging to some class determined by the

Table 8
Performance analysis of algorithms with NBTree as base classifier.

Dataset	Accuracy			F1-measures			Precision			AUC		
	OVA-NBTree	MMNBTree	MVDT-NBTree	OVA-NBTree	MMNBTree	MVDT-NBTree	OVA-NBTree	MMNBTree	MVDT-NBTree	OVA-NBTree	MMNBTree	MVDT-NBTree
iris	0.9367 (1)	0.9293 (3)	0.9307 (2)	0.9239 (1)	0.9158 (3)	0.9179 (2)	0.9303 (1)	0.9220 (3)	0.9229 (2)	0.9726 (3)	0.9854 (1)	0.9844 (2)
texture	0.9027 (3)	0.9229 (2)	0.9402 (1)	0.9028 (3)	0.9220 (2)	0.9385 (1)	0.9132 (3)	0.9245 (2)	0.9390 (1)	0.9835 (3)	0.9896 (2)	0.9948 (1)
mfeat-fac	0.8780 (3)	0.8985 (2)	0.9365 (1)	0.8828 (3)	0.8991 (2)	0.9352 (1)	0.9095 (3)	0.9100 (2)	0.9379 (1)	0.9837 (3)	0.9898 (2)	0.9936 (1)
mfeat-fou	0.6865 (3)	0.7765 (2)	0.7835 (1)	0.6636 (3)	0.7715 (2)	0.7753 (1)	0.7126 (3)	0.7759 (2)	0.7773 (1)	0.9348 (3)	0.9495 (2)	0.9634 (1)
mfeat-kar	0.8640 (3)	0.9085 (2)	0.9155 (1)	0.8679 (3)	0.9057 (2)	0.9128 (1)	0.8980 (3)	0.9084 (2)	0.9143 (1)	0.9835 (3)	0.9916 (2)	0.9935 (1)
mfeat-mor	0.6955 (3)	0.7035 (2)	0.7055 (1)	0.6524 (3)	0.6817 (1)	0.6815 (2)	0.6870 (3)	0.7033 (2)	0.7141 (1)	0.9157 (3)	0.9493 (2)	0.9551 (1)
mfeat-pix	0.8596 (3)	0.8838 (2)	0.9258 (1)	0.8650 (3)	0.8830 (2)	0.9239 (1)	0.8949 (2)	0.8916 (3)	0.9256 (1)	0.9805 (3)	0.9865 (2)	0.9916 (1)
mfeat-zer	0.6743 (3)	0.7291 (2)	0.7381 (1)	0.6411 (3)	0.7154 (2)	0.7301 (1)	0.6551 (3)	0.7208 (2)	0.7340 (1)	0.9331 (3)	0.9487 (2)	0.9623 (1)
wavform	0.8457 (2)	0.8456 (3)	0.8490 (1)	0.8443 (3)	0.8456 (2)	0.8491 (1)	0.8493 (2)	0.8489 (3)	0.8519 (1)	0.9360 (3)	0.9691 (1)	0.9671 (2)
optdigits	0.8937 (3)	0.9312 (1)	0.9294 (2)	0.8958 (3)	0.9310 (1)	0.9289 (2)	0.9121 (3)	0.9326 (1)	0.9307 (2)	0.9816 (3)	0.9902 (2)	0.9915 (1)
tae	0.5304 (3)	0.5430 (2)	0.5436 (1)	0.5043 (3)	0.5204 (2)	0.5225 (1)	0.5461 (3)	0.5516 (2)	0.5552 (1)	0.6737 (3)	0.7072 (2)	0.7099 (1)
letter	0.7128 (3)	0.8020 (1)	0.7997 (2)	0.7356 (3)	0.8019 (1)	0.7995 (2)	0.7950 (3)	0.8089 (1)	0.8083 (2)	0.9441 (3)	0.9730 (2)	0.9829 (1)
wine	0.9776 (2)	0.9747 (3)	0.9798 (1)	0.9774 (2)	0.9752 (3)	0.9791 (1)	0.9797 (2)	0.9778 (3)	0.9807 (1)	0.9947 (3)	0.9974 (2)	0.9987 (1)
cmc	0.5201 (2)	0.5192 (3)	0.5291 (1)	0.4660 (3)	0.5024 (1)	0.4964 (2)	0.5223 (1)	0.5082 (3)	0.5195 (2)	0.6808 (3)	0.7173 (1)	0.7169 (2)
molecula	0.9267 (3)	0.9454 (1)	0.9424 (2)	0.9178 (3)	0.9399 (1)	0.9368 (2)	0.9206 (3)	0.9416 (2)	0.9430 (1)	0.9734 (3)	0.9892 (2)	0.9893 (1)
satimage	0.8135 (3)	0.8469 (1)	0.8263 (2)	0.7926 (2)	0.8272 (1)	0.7756 (3)	0.8045 (3)	0.8331 (1)	0.8122 (2)	0.9465 (3)	0.9598 (2)	0.9639 (1)
balance-scale	0.9204 (1)	0.9147 (2)	0.9142 (3)	0.8408 (1)	0.6339 (2)	0.6334 (3)	0.8240 (1)	0.6094 (2)	0.6088 (3)	0.9818 (1)	0.9555 (2)	0.9465 (3)
glass	0.6681 (3)	0.7315 (1)	0.7230 (2)	0.5419 (3)	0.6552 (1)	0.6200 (2)	0.5626 (3)	0.6692 (1)	0.6304 (2)	0.8317 (3)	0.8655 (2)	0.8681 (1)
car	0.8406 (3)	0.9055 (1)	0.8418 (2)	0.6557 (2)	0.7928 (1)	0.6049 (3)	0.8317 (2)	0.8359 (1)	0.7846 (3)	0.9403 (3)	0.9801 (2)	0.9825 (1)
ecoli	0.8296 (3)	0.8403 (2)	0.8415 (1)	0.6921 (3)	0.7174 (1)	0.7058 (2)	0.7426 (3)	0.7528 (2)	0.7551 (1)	0.9210 (3)	0.9712 (2)	0.9758 (1)
yeast	0.5576 (2)	0.5891 (1)	0.5409 (3)	0.4638 (2)	0.5181 (1)	0.4431 (3)	0.5381 (1)	0.5314 (2)	0.5110 (3)	0.7539 (3)	0.8211 (2)	0.8224 (1)
nursery	0.9441 (3)	0.9705 (1)	0.9628 (2)	0.9265 (1)	0.8900 (2)	0.8156 (3)	0.9509 (1)	0.9415 (2)	0.9263 (3)	0.9914 (2)	0.9902 (3)	0.9958 (1)
Avg.Rank	2.636	1.818	1.545	2.545	1.636	1.818	2.364	2.000	1.636	2.864	1.909	1.227
Friedman $\alpha = 0.05$	✓		–	✓		–	✓		–	✓	✓	–

Table 9
Accuracies of algorithms on all data sets introducing 5% output noise.

Dataset	C4.5 as base classifier			CART as base classifier			TEIM as base classifier			SCDT as base classifier		
	C4.5	OVA-C4.5	MVDT-C4.5	CART	OVA-CART	MVDT-CART	TEIM	OVA-TEIM	MVDT-TEIM	SCDT	OVA-SCDT	MVDT-SCDT
iris	0.9200 (2)	0.9053 (3)	0.9233 (1)	0.8873 (2)	0.8820 (3)	0.8913 (1)	0.9173 (2)	0.9293 (3)	0.9307 (1)	0.9417 (2)	0.9250 (3)	0.9383 (1)
texture	0.8701 (2)	0.8385 (3)	0.9399 (1)	0.8820 (2)	0.8506 (3)	0.9315 (1)	0.8830 (2)	0.8533 (3)	0.9419 (1)	0.8601 (2)	0.8314 (3)	0.9235 (1)
mfeat-fac	0.8180 (2)	0.7366 (3)	0.8238 (1)	0.8455 (2)	0.8095 (3)	0.8926 (1)	0.8427 (2)	0.8006 (3)	0.9104 (1)	0.8196 (2)	0.7411 (3)	0.8210 (1)
mfeat-fou	0.6936 (2)	0.6044 (3)	0.7717 (1)	0.7181 (2)	0.6477 (3)	0.7755 (1)	0.7166 (2)	0.6513 (3)	0.7889 (1)	0.6764 (2)	0.6186 (3)	0.7632 (1)
mfeat-kar	0.7670 (2)	0.7174 (3)	0.8814 (1)	0.7908 (2)	0.7493 (3)	0.8729 (1)	0.7918 (2)	0.7447 (3)	0.8888 (1)	0.7369 (2)	0.7119 (3)	0.8560 (1)
mfeat-mor	0.6070 (2)	0.5456 (3)	0.6703 (1)	0.6169 (2)	0.5636 (3)	0.6749 (1)	0.7043 (2)	0.6228 (3)	0.7123 (1)	0.6070 (2)	0.5489 (3)	0.6721 (1)
mfeat-pix	0.8044 (2)	0.7767 (3)	0.8826 (1)	0.8188 (2)	0.7791 (3)	0.8764 (1)	0.8367 (2)	0.7933 (3)	0.9082 (1)	0.8254 (2)	0.7848 (3)	0.8830 (1)
mfeat-zer	0.6160 (2)	0.5733 (3)	0.7072 (1)	0.6490 (2)	0.5957 (3)	0.7122 (1)	0.6569 (2)	0.6204 (3)	0.7291 (1)	0.6490 (2)	0.5911 (3)	0.7041 (1)
wavform	0.7205 (2)	0.7063 (3)	0.7672 (1)	0.7331 (2)	0.7067 (3)	0.7690 (1)	0.7384 (2)	0.7114 (3)	0.7885 (1)	0.7282 (2)	0.7015 (3)	0.7632 (1)
optdigits	0.7979 (2)	0.7697 (3)	0.9073 (1)	0.8505 (2)	0.8071 (3)	0.9149 (1)	0.8561 (2)	0.8285 (3)	0.9377 (1)	0.8333 (2)	0.7967 (3)	0.9136 (1)
tae	0.5950 (3)	0.6040 (2)	0.6085 (1)	0.6262 (3)	0.6277 (2)	0.6422 (1)	0.5778 (2)	0.5746 (3)	0.6225 (1)	0.5629 (3)	0.6058 (2)	0.6086 (1)
letter	0.8017 (2)	0.7465 (3)	0.8802 (1)	0.8249 (2)	0.7722 (3)	0.8834 (1)	0.8409 (2)	0.7926 (3)	0.8949 (1)	0.7940 (2)	0.7639 (3)	0.8706 (1)
wine	0.6699 (3)	0.6775 (2)	0.7152 (1)	0.8810 (1)	0.8592 (3)	0.8664 (2)	0.9027 (2)	0.8761 (3)	0.9196 (1)	0.8850 (2)	0.8819 (3)	0.8917 (1)
cmc	0.4631 (3)	0.4655 (2)	0.4878 (1)	0.4776 (2)	0.4745 (3)	0.4981 (1)	0.4787 (3)	0.4856 (2)	0.5085 (1)	0.4802 (2)	0.4708 (3)	0.4942 (1)
molecula	0.8538 (2)	0.8404 (3)	0.8908 (1)	0.8520 (2)	0.8467 (3)	0.9017 (1)	0.8755 (2)	0.8523 (3)	0.9036 (1)	0.8773 (2)	0.8457 (3)	0.9026 (1)
satimage	0.8002 (2)	0.7744 (3)	0.8783 (1)	0.8122 (2)	0.7835 (3)	0.8731 (1)	0.8199 (2)	0.7854 (3)	0.8787 (1)	0.8189 (2)	0.7734 (3)	0.8752 (1)
balance-scale	0.7362 (2)	0.6938 (3)	0.7861 (1)	0.7443 (2)	0.6963 (3)	0.7810 (1)	0.7521 (2)	0.7153 (3)	0.8152 (1)	0.7482 (2)	0.7030 (3)	0.7868 (1)
glass	0.5380 (3)	0.5706 (2)	0.6666 (1)	0.6674 (1)	0.6323 (3)	0.6613 (2)	0.6946 (1)	0.6094 (3)	0.6464 (2)	0.5903 (2)	0.6051 (3)	0.6561 (1)
car	0.9090 (3)	0.9365 (2)	0.9458 (1)	0.9124 (3)	0.9279 (2)	0.9391 (1)	0.9524 (1)	0.9192 (3)	0.9491 (2)	0.9150 (3)	0.9275 (2)	0.9314 (1)
ecoli	0.7308 (2)	0.6916 (3)	0.7743 (1)	0.7628 (1)	0.7572 (2)	0.7422 (3)	0.8268 (2)	0.8057 (3)	0.8334 (1)	0.8236 (2)	0.8071 (3)	0.8392 (1)
yeast	0.4311 (3)	0.4328 (2)	0.5350 (1)	0.4877 (2)	0.4848 (3)	0.5641 (1)	0.5008 (2)	0.4833 (3)	0.5823 (1)	0.4859 (2)	0.4628 (3)	0.5585 (1)
nursery	0.9521 (2)	0.9325 (3)	0.9901 (1)	0.9220 (2)	0.8846 (3)	0.9756 (1)	0.9521 (2)	0.9325 (3)	0.9901 (1)	0.8530 (3)	0.8617 (2)	0.8764 (1)
Avg.Rank	2.273	2.727	1.000	1.955	2.864	1.182	1.955	2.955	1.091	2.136	2.864	1.000
Friedman $\alpha = 0.05$	✓	✓	–	✓	✓	–	✓	✓	–	✓	✓	–
$\alpha = 0.05$												

Table 10
Performance analysis of algorithms with AGDT as base classifier.

Dataset	Accuracy			F1-measures			Precision			AUC		
	AGDT	OVA-AGDT	MVDT-AGDT	AGDT	OVA-AGDT	MVDT-AGDT	AGDT	OVA-AGDT	MVDT-AGDT	AGDT	OVA-AGDT	MVDT-AGDT
cmc	0.4740 (2)	0.4706 (3)	0.4938 (1)	0.4497 (2)	0.4166 (3)	0.4662 (1)	0.4523 (2)	0.4394 (3)	0.4709 (1)	0.5948 (3)	0.6143 (2)	0.6552 (1)
glass	0.6283 (2)	0.6266 (3)	0.6757 (1)	0.4826 (3)	0.5106 (2)	0.5206 (1)	0.5042 (3)	0.5520 (1)	0.5302 (2)	0.7238 (3)	0.8082 (2)	0.8984 (1)
tae	0.6155 (2)	0.6073 (3)	0.6340 (1)	0.5930 (2)	0.5837 (3)	0.6115 (1)	0.6166 (3)	0.6200 (2)	0.6356 (1)	0.7098 (3)	0.7375 (2)	0.7844 (1)
yeast	0.4714 (2)	0.4671 (3)	0.5618 (1)	0.3525 (2)	0.3231 (3)	0.4043 (1)	0.3730 (3)	0.3922 (2)	0.4396 (1)	0.6560 (3)	0.7056 (2)	0.8170 (1)
Avg.Rank	2.000	3.000	1.000	2.250	2.750	1.000	2.75	2.000	1.250	3.000	2.000	1.000
Friedman $\alpha = 0.05$	✓	✓	–	✓	✓	–	✓		–	✓	✓	–

decision tree. We combine the membership probabilities of the instance given by all the decision trees, and then classify the instance as the class with highest combined probability. Due to rich information obtained from membership vectors, MVDT could achieve higher accuracy for classification. As one could obtain a more accurate prediction through decision trees corresponding to minority classes, MVDT also shows a significant performance on imbalance data sets. When there is noise in training set, most of decision trees would be slightly affected by noise data, and the combination result of these trees could still maintain a higher prediction accuracy. Hence, our algorithm is robust to data set with output noise. To evaluate the performance of our algorithm, we choose C4.5, CART, TEIM, SCDT and NBTree as base classifiers in MVDT. The experiments on 22 data sets show that the proposed MVDT has excellent performance for multi-class classification problems and has excellent robustness to output noise. As a general framework, MVDT algorithm can use any existing decision tree model as base classifier.

In future works, we will further develop theoretical research and experimental analysis on multi-label data sets. We intend to use multi-view methods to multi-label problems, where membership vector defined in this paper might be effective to represent the correlations among the labels. The challenge we face is how to learn a well constructed classification model by exploiting label correlations to predict a set of possible labels for unseen examples more accurately.

Acknowledgements

This work was supported by the National Natural Science Fund of China (No. 61432011, No. U1435212, No. 61322211), the National Key Basic Research and Development Program of China (973) (No. 2013CB329404), the Natural Science Found of Shanxi (No. 2015021101).

References

[1] J.R. Quinlan, Induction of decision trees, Mach. Learn. 1 (1) (1986) 81–106.
[2] E. Alpaydin, Introduction to Machine Learning, MIT press, 2014.
[3] R. Rojas, Neural Networks: A Systematic Introduction, Springer Science & Business Media, 2013.
[4] J. Schmidhuber, Deep learning in neural networks: an overview, Neural Netw. 61 (2015) 85–117.
[5] Q. He, Z. Xie, Q. Hu, C. Wu, Neighborhood based sample and feature selection for svm classification learning, Neurocomputing 74 (10) (2011) 1585–1594.
[6] N. Quadrianto, Z. Ghahramani, A very simple safe-Bayesian random forest, Pattern Anal. Mach. Intell. IEEE Trans. 37 (6) (2015) 1297–1303.
[7] F.V. Jensen, An Introduction to Bayesian Networks, Vol. 210, UCL press London, 1996.
[8] A.T. Azar, S.M. El-Metwally, Decision tree classifiers for automated medical diagnosis, Neural Comput. Appl. 23 (7–8) (2013) 2387–2403.
[9] E. Pashaei, M. Ozen, N. Aydin, Improving medical diagnosis reliability using boosted c5. 0 decision tree empowered by particle swarm optimization, in: Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE, IEEE, 2015, pp. 7230–7233.
[10] S.Y. Sohn, J.W. Kim, Decision tree-based technology credit scoring for start-up firms: Korean case, Expert Syst. Appl. 39 (4) (2012) 4007–4012.
[11] J.R. Quinlan, C4. 5: Programming for Machine Learning, Morgan Kauffmann, 1993. 38.
[12] Y.L.C. Jin, F. Li, A generalized fuzzy id3 algorithm using generalized information entropy, Knowl.-Based Syst. 64 (1) (2014) 13–21.
[13] L. Breiman, J. Friedman, C.J. Stone, R.A. Olshen, Classification and Regression Trees, CRC press, 1984.
[14] X.-Z. Wang, L.-C. Dong, J.-H. Yan, Maximum ambiguity-based sample selection in fuzzy decision tree induction, IEEE Trans. Knowl. Data Eng. 24 (8) (2012) 1491–1505.
[15] S. Su, X. Wang, J. Zhai, An improved cluster oriented fuzzy decision trees, in: International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing, Springer, 2009, pp. 447–454.
[16] B. Chandra, R. Kothari, P. Paul, A new node splitting measure for decision tree construction, Pattern Recognit. 43 (8) (2010) 2725–2731.
[17] C. Olaru, L. Wehenkel, A complete fuzzy decision tree technique, Fuzzy Sets Syst. 138 (2) (2003) 221–254.
[18] X. Li, H. Zhao, W. Zhu, A cost sensitive decision tree algorithm with two adaptive mechanisms, Knowl.-Based Syst. 88 (2015) 24–33.
[19] C.J. Mantas, J. Abellán, Analysis and extension of decision trees based on imprecise probabilities: application on noisy data, Expert Syst. Appl. 41 (5) (2014) 2514–2525.
[20] C.J. Mantas, J. Abellán, Credal-c4. 5: decision tree based on imprecise probabilities to classify noisy data, Expert Syst. Appl. 41 (10) (2014) 4625–4637.
[21] S. Shah, P.S. Sastry, New algorithms for learning and pruning oblique decision trees, IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.) 29 (4) (1999) 494–505.
[22] R. Weber, Fuzzy-id3: a class of methods for automatic knowledge acquisition, in: The Second International Conference on Fuzzy Logic and Neural Networks, 1992, pp. 265–268.
[23] R. Asadi, M. Ghatee, A rule-based decision support system in intelligent hazmat transportation system, IEEE Trans. Intell. Transp. Syst. 16 (5) (2015) 2756–2764.
[24] C.-C. Wu, Y.-L. Chen, Y.-H. Liu, X.-Y. Yang, Decision tree induction with a constrained number of leaf nodes, Appl. Intell. 45 (3) (2016) 673–685.
[25] Y. Wang, S.-T. Xia, J. Wu, A less-greedy two-term Tsallis entropy information metric approach for decision tree classification, Knowl.-Based Syst. 120 (2017) 34–42.
[26] Q. Hu, X. Che, L. Zhang, D. Zhang, M. Guo, D. Yu, Rank entropy-based decision trees for monotonic classification, IEEE Trans. Knowl. Data Eng. 24 (11) (2012) 2052–2064.
[27] Y. Qian, H. Xu, J. Liang, B. Liu, J. Wang, Fusing monotonic decision trees, IEEE Trans. Knowl. Data Eng. 27 (10) (2015) 2717–2728.
[28] H. Zhu, J. Zhai, S. Wang, X. Wang, Monotonic decision tree for interval valued data, in: International Conference on Machine Learning and Cybernetics, Springer, 2014, pp. 231–240.
[29] R. Wang, S. Kwong, X.-Z. Wang, Q. Jiang, Segment based decision tree induction with continuous valued attributes, IEEE Trans. Cybern. 45 (7) (2015) 1262–1275.
[30] S.A. Ludwig, D. Jakobovic, S. Picek, Analyzing gene expression data: fuzzy decision tree algorithm applied to the classification of cancer data, in: Fuzzy Systems (FUZZ-IEEE), 2015 IEEE International Conference on, IEEE, 2015, pp. 1–8.
[31] W. Yi, M. Lu, Z. Liu, Multi-valued attribute and multi-labeled data decision tree algorithm, Int. J. Mach. Learn. Cybern. 2 (2) (2011) 67–74.
[32] A. Douzal-Chouakria, C. Amblard, Classification trees for time series, Pattern Recognit. 45 (3) (2012) 1076–1091.
[33] J. Liu, X. Li, W. Zhong, Ambiguous decision trees for mining concept-drifting data streams, Pattern Recognit. Lett. 30 (15) (2009) 1347–1355.
[34] L. Rutkowski, M. Jaworski, L. Pietruczuk, P. Duda, The cart decision tree for mining data streams, Inf. Sci. 266 (2014) 1–15.
[35] R. Rifkin, A. Klautau, In defense of one-vs-all classification, J. Mach. Learn. Res. 5 (1) (2004) 101–141.
[36] S. Wang, L. Jiang, C. Li, Adapting naive Bayes tree for text classification, Knowl. Inf. Syst. 44 (1) (2015) 77–89.
[37] M. Galar, A. Ndez, E. Barrenechea, H. Bustince, F. Herrera, An overview of ensemble methods for binary classifiers in multi-class problems: experimental study on one-vs-one and one-vs-all schemes, Pattern Recognit. 44 (8) (2011) 1761–1776.
[38] T.M. Mitchell, et al., Machine learning, 1997, Wcb.
[39] A. Orriols-Puig, E. Bernadó-Mansilla, Evolutionary rule-based systems for imbalanced data sets, Soft Comput. Fusion Found. Methodol. Appl. 13 (3) (2009) 213–225.
[40] D.J. Hand, R.J. Till, A simple generalisation of the area under the roc curve for multiple class classification problems, Mach. Learn. 45 (2) (2001) 171–186.
[41] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (1) (2006) 1–30.
[42] R. Kohavi, Scaling up the accuracy of naive-Bayes classifiers: a decision-tree hybrid, KDD 96 (1996) 202–207.
[43] U.M. Fayyad, K.B. Irani, Multi-interval discretization of continuous-valued attributes for classification learning, in: International Joint Conference on Artificial Intelligence, 1993, pp. 1022–1027.