Available online at www.sciencedirect.com
SCIENCE DIRECT®
Information Sciences 172 (2005) 91–129
INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL
www.elsevier.com/locate/ins

Decision tree learning with fuzzy labels

Zengchang Qin [a,*], Jonathan Lawry [a]

[a] *Artificial Intelligence Group, Department of Engineering Mathematics, University of Bristol, Bristol BS8 1TR, UK*

Accepted 12 December 2004

Abstract

Label semantics is a random set based framework for "Computing with Words" that captures the idea of computation on linguistic terms rather than numerical quantities. Within this new framework, a decision tree learning model is proposed where nodes are linguistic descriptions of variables and leaves are sets of appropriate labels. In such decision trees, the probability estimates for branches across the whole tree is used for classification, instead of the majority class of the single branch into which the examples fall. By empirical experiments on real-world datasets it is verified that our algorithm has better or equivalent classification accuracy compared to three well known machine learning algorithms. By applying a new forward branch merging algorithm, the complexity of the tree can be greatly reduced without significant loss of accuracy. Finally, a linguistic interpretation of trees and classification with linguistic constraints are introduced.

Keywords: Label semantics; LID3; Linguistic decision tree; Mass assignment; Random set; Linguistic constraint; Transparency

* Corresponding author. Tel.: +44 1179289798; fax: +44 1179251154.
E-mail addresses: Z.Qin@bristol.ac.uk (Z. Qin), J.Lawry@bristol.ac.uk (J. Lawry).

0020-0255/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2004.12.005

1. Introduction

Machine learning and data mining research has developed rapidly in recent decades. As one of the most successful branches of Artificial Intelligence, it is playing a more and more important role in real-world applications ranging from gene expression to flood prediction. Traditionally machine learning and data mining research has focused on learning algorithms with high classification or prediction accuracy. From another perspective, however, this is not always sufficient for some real world applications that require good algorithm transparency. By the latter we mean that models need to be easily understood and provide information regarding underlying trends and relationships that can be used by practitioners in the relevant fields. Also, uncertainty and imprecision are often inherent in modelling these real-world applications and it is desirable that these should be incorporated into learning algorithms.

Here we present a high-level knowledge representation framework centered on the Computing with Words (CW) paradigm proposed by Zadeh [20], although the underlying semantics of our approach will be quite different. Label semantics [11] is a random set based semantics for modelling imprecise concepts where the degree of appropriateness of a linguistic expression as a description of a value is measured in terms of how the set of appropriate labels for that value varies across a population. It provides us with a framework for modelling uncertainty with good transparency. Based on this semantics, a new tree-structured model, *Linguistic Decision Tree (LDT)* is proposed. Linguistic expressions such as *small*, *medium* and *large* are used to learn from data and build a linguistic decision tree guided by information based heuristics. For each branch, instead of labeling it with a certain class (such as positive or negative in binary classification) the probability of members of this branch belonging to a particular class is evaluated from a given training dataset. Unlabeled data is then classified by using probability estimation of classes across the whole decision tree.

This paper is organized as follows. Section 2 presents the fundamentals of label semantics as well as the methods for data analysis and data mining based on this framework. Section 3 gives a detailed description of the proposed linguistic decision tree model, and an algorithm of building such a decision tree is outlined. We also investigate the effect of using different discretization methods with empirical results given for comparisons. Experimental results for real-world datasets are given, comparing the new algorithm with three well-known machine learning algorithms: C4.5, Naive Bayes and Back Propagation Neural networks. In Section 4, a forward branch merging algorithm is introduced in order to build a compact decision tree without significant loss of accuracy. Experimental results show that the trees with forward merging have better transparency and equivalent or better accuracy than the original LDT. In Section 5, a formal linguistic interpretation of LDT and a method for classification

Table 1
Notation for this paper

Ω	Continuous universe	LA	Set of linguistic labels
L	Linguistic label	$\mu_L(x)$	Appropriate degree of using L to label x
$\boldsymbol{D}$	Numerical database		
$\boldsymbol{LD}$	Linguistic database from linguistic translation of $\boldsymbol{D}$	N	Number of instances
		n	Number of attribute
$\boldsymbol{C}$	Set of target attributes	m	Number of classes
N_F	Number of fuzzy sets	s	Number of branches
m_x	Mass assignment on x	$\mathscr{F}$	Focal set
F	Focal element	$m_x(F)$	Associated mass of F
$\mathscr{M}$	A set of focal elements for merging	k	The length of a branch
		LDT	Linguistic decision tree
MB	Merged branch	B	Branch of a LDT
T	Threshold for termination	T_m	Threshold for merging
θ	Linguistic expressions	pm	Prior mass assignment

given linguistic constraints is proposed and the results from applying it to a toy problem are given. Prior to the introduction of the formal framework we introduce some important notation used throughout this paper as shown in Table 1.

2. Label semantics

Label semantics, proposed by Lawry [11], is a framework for modelling with linguistic expressions, or labels such as *small*, *medium* and *large*. Such labels are defined by overlapping fuzzy sets which are used to cover the universe of continuous variables. Some recent research has applied this framework to machine learning systems, and in particular to a fuzzy Bayesian model [18]. Here we will apply label semantics to the problem of decision tree induction. Initially, however, a brief overview of label semantics is given in the following section.

2.1. Random set based framework

The fundamental question posed by label semantics is how to use linguistic expressions to label numerical values. The basic idea is that when individuals make assertions, such as 'x is small', they are essentially providing information about what labels are appropriate for the values of x. For example, for a variable x into a domain of discourse denoted by Ω we identify a finite set of linguistic labels $LA = \{L_1, \ldots, L_n\}$ with which to label the values of x. Then, for a specific value $x \in \Omega$, an individual I identifies a subset of LA, denoted D_x^I to stand for the description of x given by I, as the set of labels with which it is

appropriate to label x. If we allow I to vary across a population V with prior distribution P_V, then D_x^I will also vary and generate a random set denoted D_x into the power set of LA. By evaluating the probability of occurrence of a particular set of labels say S, for D_x across the population then we obtain a distribution on D_x referred to as a mass assignment (see [1] for details on Mass Assignment theory). We can view the random set D_x as a description of the variable x in terms of the labels in LA. More formally,

Definition 1 (*Label description*). For $x \in \Omega$ the label description of x is a random set from V into the power set of LA, denoted D_x, with associated distribution m_x, given by

$$\forall S \subseteq LA, \quad m_x(S) = P_V(\{I \in V | D_x^I = S\})$$

In this framework, *appropriateness degrees* are used to evaluate how appropriate a label is for describing a particular value of variable x. This measure can be defined based on mass assignments as follows:

Definition 2 (*Appropriateness degrees*).

$$\forall x \in \Omega, \ \forall L \in LA, \quad \mu_L(x) = \sum_{S \subseteq LA: L \in S} m_x(S)$$

This definition provides a relationship between mass assignments and appropriateness degrees. Clearly μ_L is a function mapping from Ω into [0, 1] and therefore can technically be viewed as the membership function of a fuzzy set. In simple terms, given a particular value x, the appropriateness degree of L as a label for x where L is represented by fuzzy set F, is the membership value of x belonging to F. The reason we use the new term 'appropriateness degree' is partly because it more accurately reflects the underlying semantics and partly to highlight the quite distinct calculus based on this framework.

For example, an expression such as '*the score on a dice is small*', as asserted by individual I, is interpreted to mean $D_{SCORE}^I = \{small\}$, where $SCORE$ denotes the value of the score given by a single throw of a particular dice. When I varies across a population V, different sets of labels could be given to describe the variable $SCORE$, so that we obtain the random set of D_{SCORE} into the power set of LA.

Example 1. Suppose the variable $SCORE$ with universe $\{1, 2, 3, 4, 5, 6\}$ gives the outcome of a single throw of a particular dice. Let $LA = \{small, medium, large\}$ and $V = \{I_1, I_2, I_3\}$ then a possible definition of D_{SCORE} is given as follows:

$$D_1^{I_1} = D_1^{I_2} = D_1^{I_3} = \{small\}$$

$$D_2^{I_1} = \{small, medium\}, \quad D_2^{I_2} = D_2^{I_3} = \{small\}$$

$$D_3^{I_1} = D_3^{I_2} = \{medium\}, \quad D_3^{I_3} = \{small, medium\}$$

$$D_4^{I_1} = \{medium, large\}, \quad D_4^{I_2} = D_4^{I_3} = \{medium\}$$

$$D_5^{I_2} = \{large\}, \quad D_5^{I_1} = D_5^{I_3} = \{medium, large\}$$

$$D_6^{I_1} = D_6^{I_2} = D_6^{I_3} = \{large\}$$

Assuming a uniform prior distribution on V, so that $P_V = 1/|V|$, then the mass assignment of D_x can be represented according to Definition 1 as follows:

$$\forall S \subseteq LA, \quad m_x(S) = \frac{|\{I \in V | D_x^I = S\}|}{|V|} \tag{1}$$

We can determine mass assignments on D_{SCORE} according to Eq. (1). For example, if $SCORE = 4$ we have

$$m_4(\{medium, large\}) = \frac{|\{I \in V | D_4^I = \{medium, large\}\}|}{|V|} = \frac{|\{I_1\}|}{|V|} = \frac{1}{3}$$

$$m_4(\{medium\}) = \frac{|\{I \in V | D_4^I = \{medium\}\}|}{|V|} = \frac{|\{I_2, I_3\}|}{|V|} = \frac{2}{3}$$

We then have the mass assignment for $SCORE = 4$ as follows:

$$m_4 = \{medium, large\} : \frac{1}{3}, \{medium\} : \frac{2}{3}$$

In the sequel 1/3 and 2/3 are also referred to as the associated mass for {*medium*, *large*} and {*medium*}, respectively. Similarly, we can obtain

$$m_1 = \{small\} : 1, m_2 = \{small\} : \frac{2}{3}, \{small, medium\} : \frac{1}{3}$$

$$m_3 = \{small, medium\} : \frac{1}{3}, \{medium\} : \frac{2}{3}$$

$$m_5 = \{medium, large\} : \frac{2}{3}\{large\} : \frac{1}{3}, m_6 = \{large\} : 1$$

The values of $m_1, \ldots, m_6$ are not dependent on the distribution of the data (i.e. the distribution on $\{1, \ldots, 6\}$. Rather they are dependent on the sets of appropriate labels assigned to each label by each voter and on the distribution across voters. The latter in this case is assumed to be uniform.

2.2. Label semantics for data mining

Based on this underlying semantics, we can translate numeric data into a set of appropriate labels with associated masses. It is certainly true that a mass assignment on D_x determines a unique appropriateness degree for any function but generally the converse does not hold. For example, given $\mu_{L_1}(x) = 0.3$ and $\mu_{L_2}(x) = 1$, then possible sets of appropriate labels with associated masses are as follows:

$$\{L_2\} : 0.7, \{L_1, L_2\} : 0.3$$
$$\{L_1\} : 0.1, \{L_2\} : 0.8, \{L_1, L_2\} : 0.2$$
$$\{L_1\} : 0.2, \{L_2\} : 0.9, \{L_1, L_2\} : 0.1$$
$$\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots$$

In fact, there are potentially an infinite family of random sets satisfying the given constraints [5]. This problem can be overcome by making the *consonance assumptions*, according to which we can determine the mass assignment uniquely from the appropriateness degrees as follows.

Definition 3 (*Consonant mass assignments on labels*). Let $\{\beta_1, \ldots, \beta_k\} = \{\mu_L(x) \,|\, L \in LA, \mu_L(x) > 0\}$ ordered such that $\beta_t > \beta_{t+1}$ for $t = 1,2, \ldots, k-1$ then:

$$m_x = M_t : \beta t - \beta_{t-1}, \quad \text{for } t = 1, 2, \ldots, k-1$$
$$M_k : \beta_k, \quad M_0 : 1 - \beta_1$$

where $M_0 = \emptyset$ and $M_t = \{L \in LA \,|\, \mu_L(x) \geqslant \beta_t\}$ for $t = 1,2 \ldots,k$.

For the previous example, given $\mu_{L_1}(x) = 0.3$ and $\mu_{L_2}(x) = 1$, we can calculate the consonant mass assignments as follows: The appropriateness degrees are ordered as $\{\beta_1, \beta_2\} = \{1, 0.3\}$ and $M_1 = \{L_2\}$, $M_2 = \{L_1, L_2\}$. We then can obtain

$$m_x = \{L_2\} : \beta_1 - \beta_2, \{L_1, L_2\} : \beta_2 = \{L_2\} : 0.7, \{L_1, L_2\} : 0.3$$

Because the appropriateness degrees are sorted in Definition 3 the resulting mass assignments are "nested" (see Fig. 1). Clearly then, there is a unique consonant mapping to mass assignments for a given set of appropriateness degree values. The justification of the consonance assumption can be found in [1,10]. Notice that in some cases we may have non-zero mass associated with the empty set (left-hand diagram of Fig. 1). This means that some voters believe that x cannot be described by any labels in LA. This property would add to the complexity of our learning algorithms and hence we avoid it by introducing a full fuzzy covering defined as follows:

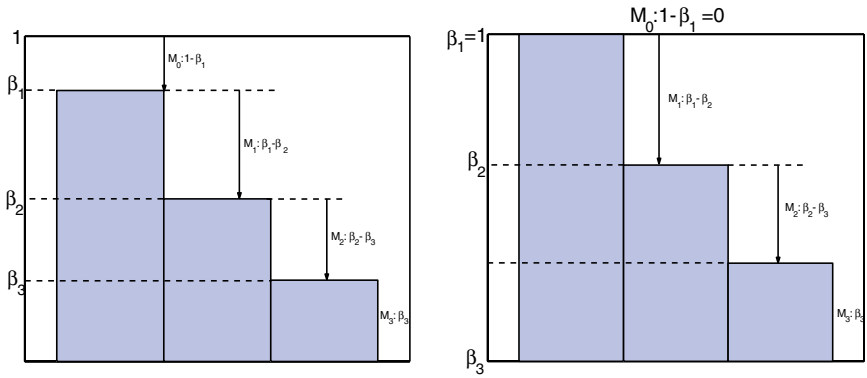

Fig. 1. Calculating mass assignments given the consonance assumption. The right-hand figure is with a full fuzzy covering while the left-hand figure is not.

Definition 4 (*Full fuzzy covering*). Given a continuous discourse Ω, LA is called a full fuzzy covering of Ω if:

$$\forall x \in \Omega,\ \exists L \in LA \quad \mu_L(x) = 1$$

In another words, the full fuzzy covering assumes that, for any element, there always exists a particular label which all the voters agree is appropriate to describe this data, though the voters may have different opinions on other labels. Unless otherwise stated, we will use N_F fuzzy sets with 50% overlap to cover a continuous universe (see Fig. 2), so that the appropriateness degrees satisfy: $\forall x \in \Omega,\ \exists i \in \{1, \ldots, N_F - 1\}$ such that $\mu_{L_i}(x) = \alpha,\ \mu_{L_i+1}(x) = \beta$ and

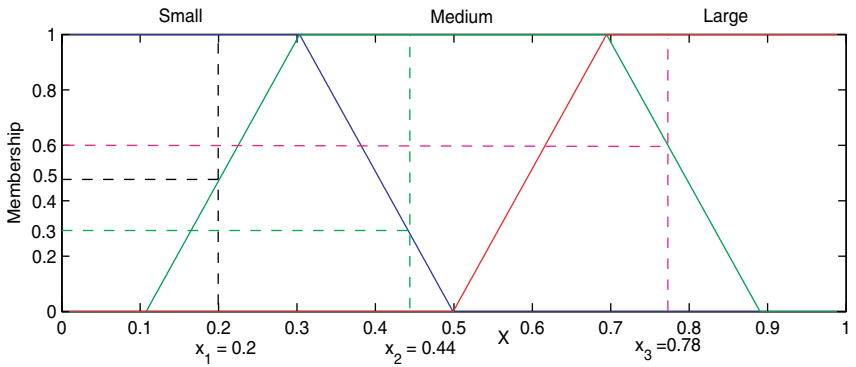

Fig. 2. An example of a full fuzzy covering with three uniformly distributed trapezoidal fuzzy sets with 50% overlap.

$\mu_{L_j}(x) = 0$ for $j < i$ or $j > i + 1$ and where $\max(\alpha, \beta) = 1$. In the case that $\alpha = 1$ according to the full fuzzy covering assumption, then m_x has the following form:

$$m_x = \{L_i\} : 1 - \beta, \{L_i, L_{i+1}\} : \beta \tag{2}$$

It is also important to note that, given definitions for the appropriateness degrees on labels, we can isolate a set of subsets of LA with non-zero masses. These are referred to as *focal sets* and the appropriate labels with non-zero masses as *focal elements*, more formally,

Definition 5 (*Focal set*). The focal set of LA is a set of focal elements defined as

$$\mathscr{F} = \{S \subseteq LA | \exists x \in \Omega, m_x(S) > 0\}$$

Based on the above assumptions (consonant, full fuzzy covering with 50% overlap) defined on a particular universe, we can then always find the unique and consistent translation from a given data element to a mass assignment on focal elements, specified by the function $\mu_L : L \in LA$. We call this the *linguistic translation (LT)*, which provides us with a way of transforming real-valued data into a set of associated masses for appropriate labels.

Definition 6 (*Linguistic translation*). Suppose we are given a numerical data set $\boldsymbol{D} = \{\langle x_1(i), \ldots, x_n(i)\rangle \,|\, i = 1, \ldots, N\}$ and focal set on attribute $j : \mathscr{F}_j = \{F_j^1, \ldots, F_j^{h_j} \,|\, j = 1, \ldots, n\}$, from the linguistic translation, we then obtain linguistic data set $\boldsymbol{LD}$ defined as follows:

$$\boldsymbol{LD} = \{A_1(i), \ldots, A_n(i) \,|\, i = 1, \ldots, N\}$$
$$A_j(i) = \{\langle m_{x_j(i)}(F_j^1), \ldots, m_{x_j(i)}(F_j^{h_j})\rangle\}$$

where $m_{x_j(i)}(F_j^r)$ is the associated mass of focal element F_j^r as appropriate labels for data element $x_j(i)$ where $r = 1, \ldots, h_j$ and $j = 1, \ldots, n$.

For a particular attribute with an associated focal set, linguistic translation is a process of replacing its data elements with the focal element masses of these data elements. For a variable x, it defines a unique mapping from data element $x(i)$ to a vector of associated masses $\langle m_{x(i)}(F^1), \ldots, m_{x(i)}(F^h)\rangle$.

Example 2. Fig. 2 shows a full fuzzy covering of the universe $\Omega = [0, 1]$ with three fuzzy labels: *small*, *medium* and *large* with 50% overlap. The following focal elements occur: {*small*}, {*small*, *medium*}, {*medium*}, {*medium*, *large*} and {*large*}, but the sets {*small*, *medium*, *large*} and {*small*, *large*} cannot occur since at no point do small, medium and large all overlap and small and large do

not overlap. For the data point $x_1 = 0.2$, the appropriate labels are small and medium, and the appropriateness degrees of these labels are read from the membership values as follows:

$$\mu_{small}(0.2) = 1, \quad \mu_{medium}(0.2) = 0.5$$

The mass assignment on the appropriate labels can be calculated based on Eq. (2) to give

$$m_{0.2} = \{small\} : 0.5, \{small, medium\} : 0.5$$

Similarly, for $x_2 = 0.44$, $x_3 = 0.78$, we obtain

$$m_{0.44} = \{small, medium\} : 0.3, \{medium\} : 0.7$$

$$m_{0.78} = \{medium, large\} : 0.6, \{large\} : 0.4$$

The linguistic translation can be illustrated as follows:

$$\begin{pmatrix} x \\ \hline 0.2 \\ 0.44 \\ 0.78 \end{pmatrix} \overset{LT}{\rightarrow} \left(\begin{array}{ccccc} m_x(\{s\}) & m_x(\{s,m\}) & m_x(\{m\}) & m_x(\{m,l\}) & m_x(\{l\}) \\ \hline 0.5 & 0.5 & 0 & 0 & 0 \\ 0 & 0.3 & 0.7 & 0 & 0 \\ 0 & 0 & 0 & 0.6 & 0.4 \end{array}\right)$$

3. Linguistic decision tree

Tree induction learning models have received a great deal of attention over recent years in the fields of machine learning and data mining because of their simplicity and effectiveness. Among them, the ID3 [16] algorithm for decision trees induction has proved to be an effective and popular algorithm for building decision trees from discrete valued data sets. The C4.5 [17] algorithm was proposed as a successor to ID3 in which an entropy based approach to crisp partitioning of continuous universes was adopted. One inherent disadvantage of crisp partitioning is that these methods make the induced decision trees sensitive to noise. This noise is not only due to the lack of precision or errors in measured features but is often present in the model itself since the available features may not be sufficient to provide a complete model of the system. For each attribute, disjoint classes are separated with clearly defined boundaries. These boundaries are 'critical' since a small change close to these points will probably cause a complete change in classification. Due to the existence of uncertainty and imprecise information in real-world problems, the class boundaries may not be defined clearly. In this case, decision trees may produce high misclassification rates in testing even if they perform well in training [12,14].

3.1. Fuzzy discretization and linguistic decision tree

Before we introduce a new decision tree model, the disadvantages of crisp discretization and the advantages of fuzzy discretization are discussed. Fig. 3 shows a decision tree in a two-class problem, in which there are two continuous attributes x and y. Using crisp discretization, the decision space is partitioned into a set of non-overlapping subregions A_1, A_2 and A_3, which have clear boundaries with each other. The object for classification will definitely fall into one of these areas. For example, the given object ($x = 13.5$, $y = 46.0$) will be classified as A_3, However, if this object is distorted due to noise so that ($x = 12.9$, $y = 46.2$), then the object will be misclassified as A_1 (see Fig. 3a).

In contrast, consider the use of fuzzy discretization (Fig. 3b), where the continuous universe is partitioned by overlapped trapezoidal fuzzy sets $\{x_1, x_2\}$ and $\{y_1, y_2\}$. As shown in right-hand figure, A_1, A_2 and A_3 generated from fuzzy discretization appear as overlapping subregions with blurred boundaries. The possibility degree of an object belonging to the each class will be given

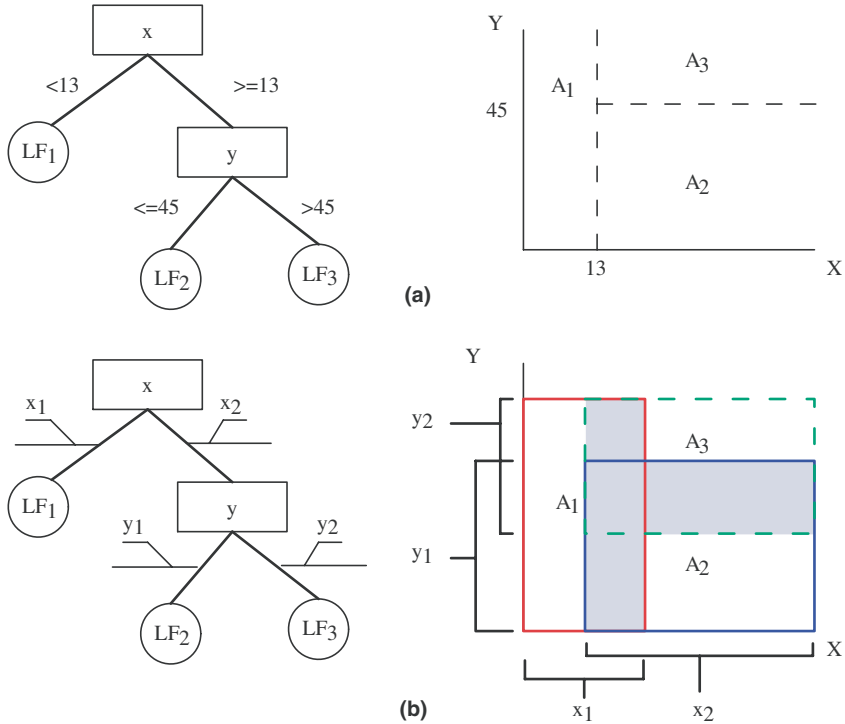

Fig. 3. Illustrations of crisp discretization and fuzzy discretization for decision tree models. (This figure was inspired by the similar figure appearing in [14].)

by the membership of pre-defined fuzzy sets. The object will fall in the overlapping area. These results can then aid the human user to make their final decisions or suggest further investigation.

Many fuzzy approaches for decision tree learning have been proposed to overcome the weaknesses described above [2,6,13,14]. In particular, Ref. [13] gives a comprehensive overview of the fuzzy decision tree literature. Different from other approaches, the algorithm we present here is based on label semantics and emphasises algorithm transparency.

Consider a database $\boldsymbol{D} = \{\langle x_1(i), \ldots, x_n(i)\rangle \mid i = 1, \ldots, N\}$ where each instance has n attributes and is labeled by one of the classes: $\{C_1, \ldots, C_m\}$. Unless otherwise stated, we use uniformly distributed fuzzy sets with 50% overlap to discretized each continuous attribute universe and obtain a corresponding linguistic data set $\boldsymbol{LD}$ by applying linguistic translation (Definition 6). A linguistic decision tree is a decision tree where the nodes are the random set label descriptions and the branches correspond to particular focal elements based on $\boldsymbol{LD}$. More formally:

Definition 7 (*Linguistic decision tree*). A linguistic decision tree is a set of branches with associated class probabilities of the following form:

$$LDT = \{\langle B_1, Pr(C_1|B_1), \ldots, Pr(C_m|B_1)\rangle, \ldots \langle B_s, Pr(C_1|B_s), \ldots, Pr(C_m)|B_s)\rangle\}$$

and a branch B with k nodes is defined as

$$B = \langle F_{j_1}, \ldots, F_{j_k}\rangle$$

where, $k \leqslant n$ and $F_j \in \mathscr{F}_j$.

F_{j_i} for $i = 1, \ldots, k$ are the branch nodes represented by focal elements of attribute j_i and i is the node position in the branch B. Within a LDT (see Fig. 4) each node splits into branches according to the focal elements of this

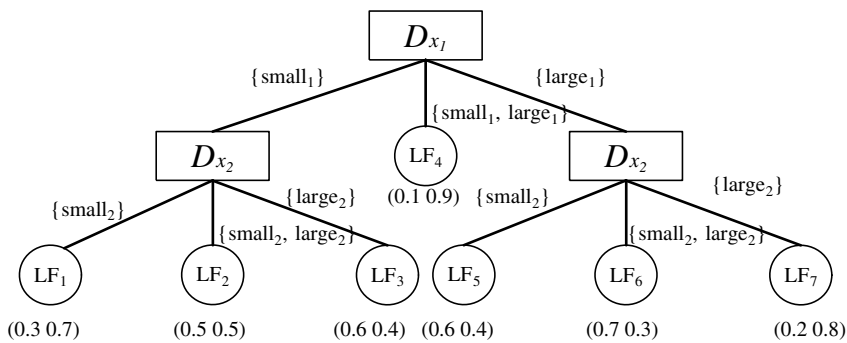

Fig. 4. An example of a linguistic decision tree on a binary classification problem.

node (attribute). One attribute is not allowed to appear more than once in a branch, and an attribute which is not currently part of a branch is referred to as a *free attribute*.

Definition 8 (*Free attributes*). The set of attributes free to use for expanding a given branch B is defined by

$$ATT_B = \{x_j \,|\, \forall F \in \mathscr{F}_j; F \notin B\}$$

The *length of a branch*, corresponding to the number of component nodes (attributes), is less than or equal to n, the number of attributes. In a LDT, the length of the longest branch is called the *depth of the LDT*, which is also less than or equal to n. Each branch has an associated probability distribution on the classes. For example, a LDT shown in Fig. 4 might be obtained from training, the branch $\langle\langle\{large_1\}, \{small_2, large_2\}\rangle, 0.7, 0.3\rangle$ means the probability of class C_1 is 0.3 and C_2 is 0.7 given attribute 1 that can be only described as *large* and attribute 2 can be described as *small* and *large*. We need to be aware that the linguistic expressions such as *small*, *medium* or *large* for each attribute are not necessarily the same, since they are defined independently on each attribute.

3.2. Evaluating class probabilities for a given branch

According to the definition of LDT (Definition 7), for a branch of a LDT of the form of $B = \langle F_{j_1}, \ldots, F_{j_k}\rangle$, the probability of class C_t ($t = 1, \ldots, m$) given B can then be evaluated from the training linguistic dataset $\boldsymbol{LD}$ as follows:

$$Pr(C_t|B) = \frac{S(B|\boldsymbol{LD}_t)}{S(B|\boldsymbol{LD})} = \frac{\sum_{i\in\boldsymbol{LD}_t}\prod_{r=1}^{k} m_{x_{jr}(i)}(F_{jr})}{\sum_{i\in\boldsymbol{LD}}\prod_{r=1}^{k} m_{x_{jr}(i)}(F_{jr})} \tag{3}$$

where $S(\cdot)$ is a function for calculating the sum of products of masses associated with focal elements consisting the branch B (see Example 3). $\boldsymbol{LD}_t$ is the subset consisting of instances belong to class t and $S(B|\boldsymbol{LD}) \neq 0$. In fact $S(B|\boldsymbol{LD}_t)$ is proportional to the probability that the branch is satisfied and that the class is C_t. Similarly, $S(B|\boldsymbol{LD})$ is proportional to the probability that the branch is satisfied. All probabilities are calculated on the basis of the linguistic database $\boldsymbol{LD}$. To understand the form of the equation note that the probability branch $\langle F_{j_1}, \ldots, F_{j_k}\rangle$ is satisfied by attribute values $\langle x_1, \ldots, x_n\rangle$ is simply taken as being proportional to the product of the probabilities of focal element F_{j_r} given x_{j_r} (i.e. $m_{x_{j_r}}(F_{j_r})$). This corresponds to the conditional independence assumption that the labels appropriate to describe x_{jr} depends only on the

value of that variable, once known, and is independent of the values of all other variables. These products are then summed across the relevant parts of the database to obtain the required probabilities.

In the case of $S(B|\boldsymbol{LD}) = 0$, which can occur when the training database for the LDT is small then there is no non-zero linguistic data covered by the branch. In this case, we obtain no information from the database so equal probabilities are assigned to each class:

$$Pr(C_t|B) = \frac{1}{m} \text{ for } t = 1, \ldots, m \quad \text{if } S(B|\boldsymbol{LD}) = 0 \tag{4}$$

At a particular depth, if one of the class probability reaches a certain threshold, for example 0.9, then we might take the view that this branch is sufficiently discriminating and that further expansion may lead to overfitting. In this case terminating the tree expansion at the current depth will probably help maintain accuracy on the test set. To this end, we employ a *threshold probability* to determine whether or not a particular branch should terminate.

Definition 9 (*Threshold probability*). In the process of building a linguistic decision tree, if the maximum class probability given a particular branch is greater than or equal to a given threshold probability T, then the branch will be terminated at the current depth.

Obviously, when using this probability-based thresholding, the branches of a tree may have different lengths. For example, see Fig. 4, where the threshold probability is $T = 0.9$, so that the 4th branch $\langle\{small_1, large_1\}\rangle$ is terminated at the depth 1 while the other branches expand to the next depth.

In the above discussion we have been concerned with continuous (or numerical) attribute, but can we learn with discrete (or nominal) attributes? One problem is that the values of discrete attributes do not have a natural ordering like continuous ones. For example, values for a person's age can be sorted in an increasing manner so that the labels *young*, *middle-aged* and *old*, can be meaningfully defined by fuzzy sets. However, if we consider the gender of a person, there are only two possible values: *male* or *female*, which are unordered. Hence, partitioning discrete attribute domains using fuzzy labels is problematic. Instead we do not attempt to group discrete values but treat discrete values as distinct labels which do not overlap with each other. Hence, the following focal elements for the attribute "gender" are: {male} and {female}. In this representation the associated masses for each focal element will be binary, i.e. either zero or one. For instance,

$$m_{\text{gender}}(\{\text{male}\}) = \begin{cases} 1 & \text{if gender} = \text{male} \\ 0 & \text{otherwise} \end{cases}$$

Table 2
A small-scale linguistic dataset with only two attributes

Instance	Attribute 1 (x_1)			Attribute 2 (x_2)			Class
	$\{s_1\}$	$\{s_1, l_1\}$	$\{l_1\}$	$\{s_2\}$	$\{s_2, l_2\}$	$\{l_2\}$	
1	0	0.4	0.6	0	0.7	0.3	+
2	0.2	0.8	0	0.5	0.5	0	−
3	0	0.9	0.1	1	0	0	−
4	0	0	0	0	1	0	+
5	0	1	0	0.3	0.7	0	+

Missing values can be handled simply by assigning the masses of corresponding focal elements to zeros. For example, in Table 2, the 4th instance has a missing value in Attribute 1. Instead of pre-processing [1] the data, we simply assign the value zero to each mass for focal elements of this missing value.

Example 3. Consider a two-class problem with 2 attributes, where $LA_1 = \{small_1(s_1), large_1(l_1)\}$ and $LA_2 = \{small_2(s_2), large_2(l_2)\}$. We assume the focal set $\mathscr{F}_1 = \{\{s_1\}, \{s_1, l_1\}, \{l_1\}\}$ and $\mathscr{F}_2 = \{\{s_2\}, \{s_2, l_2\}, \{l_2\}\}$. Suppose that the linguistic database generated from the training database is given in Table 2, and it has two target classes: positive (+) and negative (−). Now suppose we are given two branches of the form:

$$B_1 = \langle\langle\{small_1\}, \{small_2\}\rangle, Pr(+|B_1), Pr(-|B_1)\rangle$$
$$B_2 = \langle\langle\{small_1, large_1\}, \{small_2, large_2\}\rangle, Pr(+|B_2), Pr(-|B_2)\rangle$$

These two branches are evaluated according to Eqs. (3) and (4) so that

$$Pr(+, B_1) = \frac{\sum_{i=1,4,5} \prod_{r=1,2} m_{x_{j_r}(i)}(F_{j_r})}{\sum_{i=1}^{5} \prod_{r=1,2} m_{x_{j_r}(i)}(F_{j_r})} = \frac{\sum_{i=1,4,5} m_{x_1(i)}(\{s_1\}) \times m_{x_2(i)}(\{s_2\})}{\sum_{i=1}^{5} m_{x_1(i)}(\{s_1\}) \times m_{x_2(i)}(\{s_2\})}$$
$$= \frac{0 \times 0 + 0 \times 0 + 0 \times 0.3}{0 \times 0 + 0.2 \times 0.5 + 0 \times 1 + 0 \times 0 + 0 \times 0.3} = 0$$

$$Pr(-, B_1) = \frac{\sum_{i=2,3} \prod_{r=1,2} m_{x_{j_r}(i)}(F_{j_r})}{\sum_{i=1}^{5} \prod_{r=1,2} m_{x_{j_r}(i)}(F_{j_r})} = \frac{\sum_{i=2,3} m_{x_1(i)}(\{s_1\}) \times m_{x_2(i)}(\{s_2\})}{\sum_{i=1}^{5} m_{x_1(i)}(\{s_1\}) \times m_{x_2(i)}(\{s_2\})}$$
$$= \frac{0.2 \times 0.5 + 0 \times 0}{0 \times 0 + 0.2 \times 0.5 + 0 \times 1 + 0 \times 0 + 0 \times 0.3} = \frac{0.1}{0.1} = 1$$

[1] Some data pre-processing techniques replace the missing values with the mean of this attribute within a numerical attribute or categorize the missing values as a new value within a nominal attribute.

$$
\begin{aligned}
Pr(+,B_2) &= \frac{\sum_{i=1,4,5}\prod_{r=1,2} m_{x_{j_r}(i)}(F_{j_r})}{\sum_{i=1}^{5}\prod_{r=1,2} m_{x_{jr}(i)}(F_{j_r})} \\
&= \frac{\sum_{i=1,4,5} m_{x_1(i)}(\{s_1,l_1\}) \times m_{x_2(i)}(\{s_2,l_2\})}{\sum_{i=1}^{5} m_{x_1(i)}(\{s_1,l_1\}) \times m_{x_2(i)}(\{s_2,l_2\})} \\
&= \frac{0.4\times 0.7 + 0\times 1 + 1\times 0.7}{0.4\times 0.7 + 0.8\times 0.5 + 0.9\times 0 + 0\times 1 + 1\times 0.7} \\
&= \frac{0.28+0.7}{0.28+0.4+0.7} = 0.710
\end{aligned}
$$

$$
\begin{aligned}
Pr(-,B_2) &= \frac{\sum_{i=2,3}\prod_{r=1,2} m_{x_{j_r}(i)}(F_{j_r})}{\sum_{i=1}^{5}\prod_{r=1,2} m_{x_{j_r}(i)}(F_{j_r})} \\
&= \frac{\sum_{i=2,3} m_{x_1(i)}(\{s_1,l_1\}) \times m_{x_2(i)}(\{s_2,l_2\})}{\sum_{i=1}^{5} m_{x_1(i)}(\{s_1,l_1\}) \times m_{x_2(i)}(\{s_2,l_2\})} \\
&= \frac{0.8\times 0.5 + 0.9\times 0}{0.4\times 0.7 + 0.8\times 0.5 + 0.9\times 0 + 0\times 1 + 1\times 0.7} \\
&= \frac{0.4}{0.28+0.4+0.7} = 0.290
\end{aligned}
$$

3.3. Classification

Consider classifying a given data vector $\vec{y} = \langle y_1, \ldots, y_n\rangle$ which may not be contained in the training data set $\boldsymbol{D}$. Firstly, we need to translate $\vec{y}$ into a linguistic form based on the fuzzy covering of the training data. In the case that, for some attribute the data element appears beyond the range of training data $[R_{\min}, R_{\max}]$, we assign the appropriateness degrees of $R_{\min}$ or $R_{\max}$ to the element depending on which side of the range it appears.

Jeffrey's rule:

$$Pr(a) = Pr(a|b)Pr(b) + Pr(a|\neg b)Pr(\neg b)$$

is used for classifying a new data element, where $Pr(b)$ and $Pr(\neg b)$ are considered as the beliefs in b and *not* b, respectively [7]. This can be generalised when given a new condition c:

$$Pr(a|c) = Pr(a|b)Pr(b|c) + Pr(a|\neg b)Pr(\neg b|c)$$

Hence, we can evaluate the probabilities of class C_t based on a given LDT consisting of s branches by using Jeffrey's rule as follows:

$$Pr(C_t|\vec{y}) = \sum_{v=1}^{s} Pr(C_t|B_v)Pr(B_v|\vec{y}) \tag{5}$$

where $Pr(B|\vec{y})$ is the probability of a particular branch given a data element. This can be evaluated by taking the product of associated masses of focal elements of the data element along the branch. More formally,

$$Pr(B|\vec{y}) = \sum_{r=1}^{k} m_{y_{j_r}}(F_{j_r}) \tag{6}$$

In classical decision trees, classification is made according to the class label of the branch in which the data falls. In our approach, the data for classification partially satisfies the constraints represented by a number of branches and the probability estimates across the whole decision tree are then used to obtain an overall classification.

Example 4. Suppose we are given the linguistic decision tree shown in Fig. 4 for a two-class problem with $\mathscr{F}_1 = \{\{small_1\}, \{small_1, large_1\}, \{large_1\}\}$, $\mathscr{F}_2 = \{\{small_2, \{small_2, large_2\}, \{large_2\}\}$. A data element $\vec{y} = \langle y_1, y_2\rangle$ for classification is given such that $\mu_{small_1}(y_1) = 1$, $\mu_{large_1}(y_1) = 0.4$ and $\mu_{small_2}(y_2) = 0.2$, $\mu_{large_2}(y_2) = 1$.

The LDT given in Fig. 4 can be written as

$$\begin{aligned} LDT &= \{B_1, B_2, B_3, B_4, B_5, B_6, B_7\} \\ &= \{\langle\langle\{small_1\}, \{small_2\}\rangle, 0.3, 0.7\rangle, \\ &\quad \langle\langle\{small_1\}, \{small_2, large_2\}\rangle, 0.5, 0.5\rangle, \\ &\quad \langle\langle\{small_1\}, \{large_2\}\rangle, 0.6, 0.4\rangle, \\ &\quad \langle\langle\{small_1, large_1\}\rangle, 0.1, 0.9\rangle, \\ &\quad \langle\langle\{large_1\}, \{small_2\}\rangle, 0.6, 0.4\rangle \\ &\quad \langle\langle\{large_1\}, \{small_2, large_2\}\rangle, 0.7, 0.3\rangle \\ &\quad \langle\langle\{large_1\}, \{large_2\}\rangle, 0.2, 0.8\rangle\} \end{aligned}$$

The mass assignments on $\langle y_1, y_2\rangle$ are

$$m_{y_1} = \{small_1, large_1\} : 0.4, \{small_1\} : 0.6$$

$$m_{y_2} = \{small_2, large_2\} : 0.2, \{large_2\} : 0.8$$

According to Eq. (6), we obtain that

$$Pr(B_2|\vec{y}) = m_{y_1}(\{small_1\}) \times m_{y_2}(\{small_2, large_2\}) = 0.6 \times 0.2 = 0.12$$

$$Pr(B_3|\vec{y}) = m_{y_1}(\{small_1\}) \times m_{y_2}(\{large_2\}) = 0.6 \times 0.8 = 0.48$$

$$Pr(B_4|\vec{y}) = m_{y_1}(\{small_1, large_1\}) = 0.4$$

$$Pr(B_1|\vec{y}) = Pr(B_5|\vec{y}) = Pr(B_6|\vec{y}) = Pr(B_7|\vec{y}) = 0$$

Hence, from Jeffrey's rule (Eq. (5)) we obtain that

$$Pr(C_1|\vec{y}) = \sum_{v=1}^{7} Pr(B_v|\vec{y})Pr(C_1|B_v) = \sum_{v=2,3,4} Pr(B_v|\vec{y})Pr(C_1|B_v)$$
$$= 0.12 \times 0.5 + 0.48 \times 0.6 + 0.4 \times 0.1 = 0.388$$

and

$$Pr(C_2|\vec{y}) = \sum_{v=1}^{7} Pr(B_v|\vec{y})Pr(C_2|B_v) = \sum_{v=2,3,4} Pr(B_v|\vec{y})Pr(C_2|B_v)$$
$$= 0.12 \times 0.5 + 0.48 \times 0.6 + 0.4 \times 0.9 = 0.612$$

3.4. LID3 algorithm

Linguistic ID3 (LID3) is the learning algorithm we propose for building the linguistic decision tree based on a given linguistic database. Similar to the ID3 algorithm [16], search is guided by an information based heuristic, but the information measurements of a LDT are modified in accordance with label semantics. The measure of information defined for a branch B and can be viewed as an extension of the entropy measure used in ID3.

Definition 10 (*Branch entropy*). The entropy of branch B is given by

$$E(B) = -\sum_{t=1}^{m} Pr(C_t|B)\log_2(Pr(C_t|B)) \tag{7}$$

Now, given a particular branch B suppose we want to expand it with the attribute x_j. The evaluation of this attribute will be given based on the expected entropy defined as follows.

Definition 11 (*Expected entropy*).

$$EE(B, x_j) = \sum_{F_j \in \mathscr{F}_j} E(B \cup F_j) \cdot Pr(F_j|B) \tag{8}$$

where $B \cup F_j$ represents the new branch obtained by appending the focal element F_j to the end of branch B. The probability of F_j given B can be calculated as follows:

$$Pr(F_j|B) = \frac{S(B \cup F_j|\boldsymbol{LD})}{S(B|\boldsymbol{LD})} \tag{9}$$

We can now define the *Information Gain (IG)* obtained by expanding branch B with attribute x_j as

$$IG(B, x_j) = E(B) - EE(B, x_j) \tag{10}$$

The goal of tree-structured learning models is to make subregions partitioned by branches be less "impure", in terms of the mixture of class labels, than the unpartitioned dataset. For a particular branch, the most suitable free attribute for further expanding (or partitioning), is the one by which the "pureness" is maximumly increased with expanding. That corresponds to selecting the attribute with maximum information gain. As with ID3 learning, the most informative attribute will form the root of a linguistic decision tree, and the tree will expand into branches associated with all possible focal elements of this attribute. For each branch, the free attribute with maximum information gain will be the next node, from level to level, until the tree reaches the maximum specified depth or the maximum class probability reaches the given threshold probability.

3.5. Non-uniform discretization with fuzzy sets

In this section, we discuss two alternative discretization techniques that can be used instead of the uniform method described above: percentile-based discretization and entropy-based discretization. Basically, fuzzy discretization provides an interpretation between numerical data and linguistic data based on label semantics. The effectiveness of fuzzy discretization depends much on the algorithm's performance based on the linguistic data. The simplest approach is to use uniformly distributed fuzzy sets for discretization. However, in some real-world applications, background knowledge about attributes may be available and can be used directly for discretization rather than an automatic discretization technique. For example, a feature ranging from 1 to 99 to describe human age can be uniformly discretized into 3 intervals: [1, 33], [34, 66] and [67, 99]. But our background knowledge suggesting that the partition [1, 25], [26, 50] and [50, 99] may be more reasonable. However, if no relevant background knowledge is available, the question remains as to whether we can improve on uniform discretization.

In the LDT model, focal elements are directly used as branches in building the linguistic decision tree but not the fuzzy sets. In fact, there is a unique mapping from trapezoidal fuzzy sets to focal elements which can be represented by triangular functions. [2] Formally, these functions correspond to the $m_x(F)$ as x varies, for each focal element F. In order to improve the performance of our algorithm, we need to generate focal elements that are as discriminative as possible and the associated fuzzy sets can then be obtained according to Definitions 2 and 5. See Fig. 5, for example, where we obtain the discriminative focal elements (asymmetric triangular fuzzy sets) using the percentile-based

[2] The focal elements at the two extreme sides are still trapezoidal and not triangular, for example see Fig. 5.

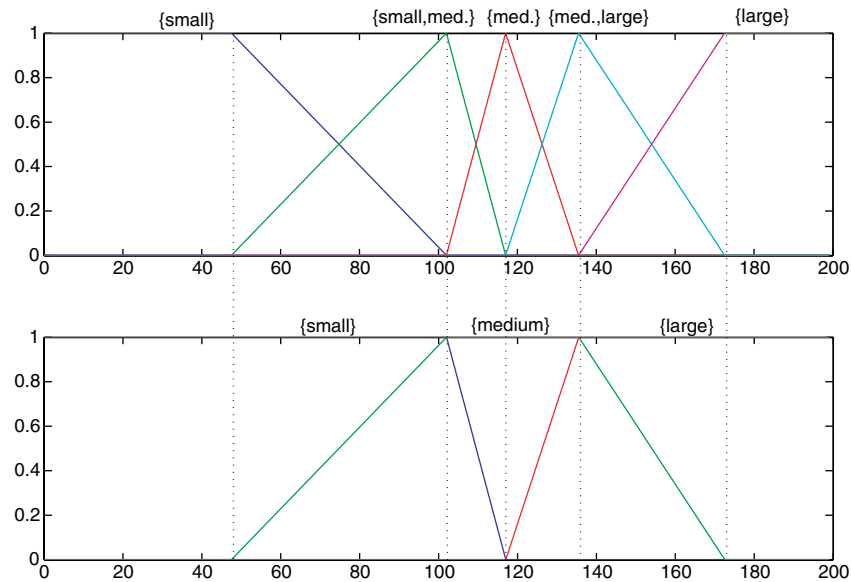

Fig. 5. An example illustrating the relation between fuzzy sets and focal elements based on given fuzzy sets.

discretization outlined below and then generate associated trapezoidal fuzzy sets using Definitions 2 and 5. The following two discretization methods are introduced and will be used to generate fuzzy labels in further experimental studies.

3.5.1. Percentile-based discretization

In this approach discretization is based on data distribution, so that the attribute universe is partitioned into intervals which each contains approximately the same number of data elements. It is a very intuitive way for generating fuzzy sets. For example, see Fig. 5 where the continuous attribute universe for variable x ranging from 0 to 200 is labeled by 3 fuzzy sets: *small*, *medium* and *large* ($N_F = 3$). According to the assumptions we made in Section 2, there are 5 focal elements. We then need 4 cut points that partition the universe into 5 intervals each containing approximately the same number of examples. The functions of focal elements $m_x(F)$ as x varies are drawn in the upper sub-figure and the fuzzy sets obtained are shown in the lower sub-figure.

3.5.2. Entropy-based discretization

In this approach, the discretization is based on the expected entropy of the resulting partition. In fact we aim to obtain the partition maximizing the *information gain* [12]. For a particular attribute, suppose we have a set of data

values $S = \{x_1, \ldots, x_N\}$ according to which we want to define q focal elements, then initially we need to find $q - 1$ cut points forming a partition of the universe. These boundary points are identified so as to maximize information gain in the following way. Every pair of adjacent data points suggests a potential boundary point: $\gamma_i = (x_i + x_{i+1})/2$ where $i = 1, \ldots, N - 1$. Now Fayyad [4] has proved that only the *class boundary points* can be the boundary points if we are to obtain the maximum information in classification, which implies that γ_i cannot lead to a partition that has maximum information gain if x_i and x_{i+1} belong to the same class. Therefore, we should generate a candidate set, which contains all of class boundary points, from which we then need to find $q - 1$ points with which we can maximize the information gain defined by Eq. (11) [12]:

$$Gain(S, \Theta) = Entropy(S) - \sum_{v=1,\ldots,q} \frac{|S_v|}{|S|} Entropy(S_v) \tag{11}$$

where Θ is a subset of the candidate set containing the $q - 1$ cut points. The cut points partition the original universe S into q intervals: $S_1 \ldots, S_q$. And the entropy is defined in Eq. (12), where m is the number of classes and p_i is the percentage of instances belong to a particular class within S:

$$Entropy(S) = \sum_{i=1}^{m} -p_i \log_2 p_i \tag{12}$$

3.6. Experimental studies

We evaluated the LID3 algorithm by using 14 datasets taken from the UCI Machine Learning repository [3]. These datasets have representative properties of real-world data, such as missing values, multi-classes, mixed-type data (numerical, nominal) and unbalanced class distributions, etc. Table 3 shows the dataset, the number of classes, the number of instances, the number of numerical (Num.) and nominal (Nom.) attributes and whether or not the database contains missing values.

3.6.1. Experimental strategies

In the following experiments, unless otherwise stated, attributes are discretized by 2 trapezoidal fuzzy sets with 50% overlap, and classes are evenly splited into two sub-datasets, one half for training and the other half for testing, this is referred to as 50–50 split experiment. The maximal depth is set manually and the results presented in this paper show the best performance of LID3 across a range of depth settings.We also test the LID3 algorithm with different threshold probabilities T ranging from 0.6 to 1.0 in steps of 0.1 and for the different fuzzy discretization methods: uniform (Uni), entropy-based (En) and percen-

Table 3
Descriptions of datasets from UCI repository

No.	Dataset	Classes	Size	Missing values	Num. of attributes	
					Numerical	Nominal
1	Balance	3	625	No	4	0
2	Breast-cancer[a]	2	286	Yes	3	6
3	Breast-w	2	699	No	9	0
4	Ecoli	8	336	No	7	1
5	Glass	6	214	No	9	0
6	Heart-c	2	303	Yes	6	7
7	Heart-Statlog	2	270	No	7	6
8	Heptitis	2	155	Yes	6	13
9	Ionosphere	2	351	No	34	0
10	Iris	3	150	No	4	0
11	Liver	2	345	No	6	0
12	Pima	2	768	No	8	0
13	Sonar[b]	2	208	No	60	0
14	Wine	3	178	No	14	0

[a] There are nine nominal attributes in the dataset, but three of them are ordered and therefore can be treated as numerical attributes.
[b] A particular single split is used.

tile-based (Per). For each dataset, we ran 50–50 random split experiment for 10 times. The average test accuracy with standard deviation are shown in the right-hand side of Table 5, the probability and the depth at which we obtain this accuracy are listed in Table 4.

3.6.2. The influence of the threshold probability

As can be seen from the results, the best accuracy is usually obtained with high threshold probabilities $T = 0.9$ or $T = 1.0$, especially for datasets with only numerical attributes (such as breast-w, iris, balance, wine) or where numerical attributes play important roles in learning (ecoli, heptitis). Recent work on PETs (Probability Estimation Trees) [15] also suggests that the full expanded estimation trees give better performance than pruned trees. [3] The reason for this is that the heuristics used to generate small and compact tree by pruning tend to reduce the quality of the probability estimates [15]. In this context linguistic decision trees can be thought of as a type of probability estimation trees but where the branches correspond to linguistic descriptions of objects.

The difference in accuracy resulting from varying the threshold probability T is quite data dependent. For instance, Fig. 6 shows the results of four typical

[3] In classical decision tree learning, pruning can reduce overfitting so that the pruned trees have better generalization and perform better then unpruned trees. However, this is not the case for PETs [15].

Table 4
Summary of the threshold probabilities and depths for obtaining the best accuracy with different discretization methods on the given datasets

No.	LID3-Uniform		LID3-Entropy		LID3-Percentile	
	Threshold	Depth	Threshold	Depth	Threshold	Depth
1	1.0	4	1.0	4	1.0	4
2	0.7	2	0.7	2	0.7	2
3	1.0	4	1.0	3	1.0	3
4	1.0	7	1.0	7	1.0	7
5	0.9	9	0.8	9	0.8	8
6	0.9	3	0.9	4	0.9	3
7	0.9	3	0.9	3	0.9	4
8	0.9	4	0.9	4	0.9	3
9	0.9	6	0.9	6	0.9	6
10	1.0	3	1.0	3	1.0	3
11	0.9	5	1.0	5	1.0	5
12	0.9	5	0.9	4	0.9	3
13	1.0	8	1.0	8	1.0	8
14	1.0	4	1.0	5	1.0	5

datasets: breast-w, heart-statlog, glass and breast-cancer. In breast-w, the accuracy curves are nested relative to increasing values of T. The models with high T values outperform those with lower T values in all depths. Dataset iris, balance, sonar, wine, ecoli also behave in this way. On the other hand, for datasets heart-statlog, pima, liver, heart-c and heptitis, the accuracy curve of $T = 0.9$ is better than all other T values at certain depths. In addition, datasets glass and ecoli have accuracy curves which are very close to each other and are even identical on some trials. For the breast-cancer dataset the accuracy actually decreases with increasing T.

3.6.3. Comparing LID3 to other machine learning algorithms

From Table 4, we also see that the optimal values of T and depth are relatively invariant across the discretization techniques. Overall the entropy-based and percentile-based discretization methods performed better than the uniform discretization although no statistically significant difference was found between the three methods. We now compare LID3 with different discretization with C4.5, Naive Bayes (N.B.) Learning and Neural Networks (N.N.) [4] using 10 50–50 splits on each dataset and the average accuracy and standard deviation for each test are shown in Table 5.

The reason of choosing these three particular learning algorithms is as follows; C4.5 is the most well-known tree induction algorithm, Naive Bayes is a

[4] WEKA [19] is used to generate the results of J48 (C4.5 in WEKA) unpruned tree, Naive Bayes and Neural Networks with default parameter settings.

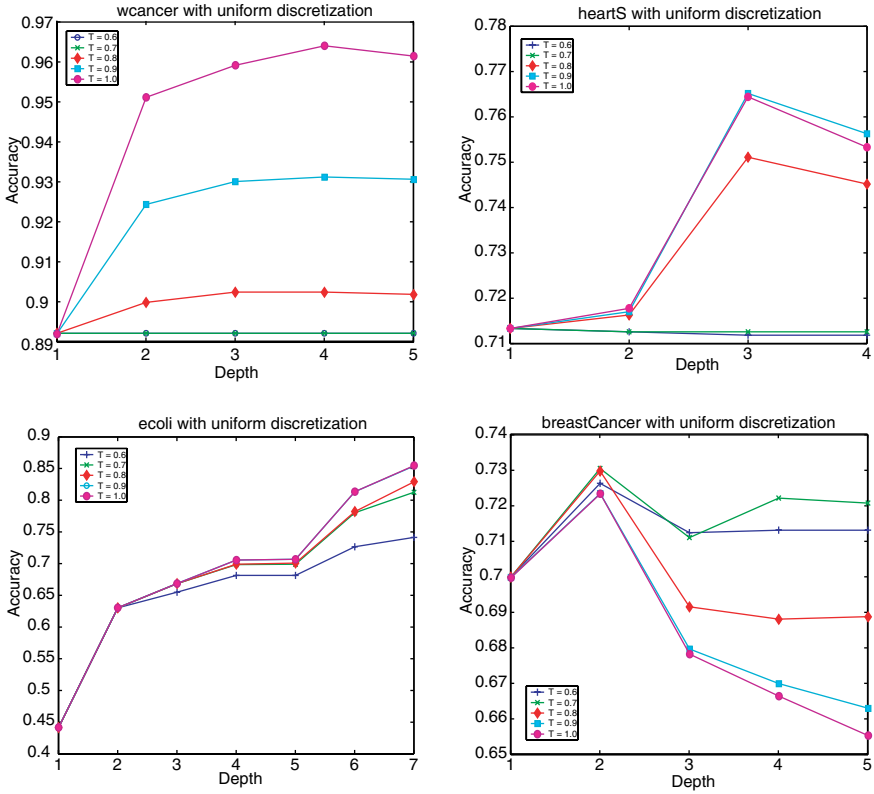

Fig. 6. Comparisons of accuracy at different depth with threshold probability ranges from 0.6 to 1.0 on four typical datasets.

simple but effective probability estimation method and neural networks are a blackbox model well known for its high predictive accuracy. We then carried out paired *t*-tests [12] with confidence level 90% to compare LID3-Uniform, LID3-Entropy and LID3-Percentile with each of the three algorithms. A summary of the results is shown in Table 6.

Across the data sets, all LID3 algorithms (with different discretization techniques) outperform C4.5, with LID3-Percentile achieving the best results with 10 wins, 4 ties and no losses. The performance of the Naive Bayes algorithm and LID3-Uniform is roughly equivalent although LID3-Entropy and LID3-Percentile outperform Naive Bayes. From Table 5, we can see that the datasets on which Naive Bayes outperforms LID3 are those with a mixture of continuous and discrete attributes, namely heart-c, heart-statlog and heptitis. Most of the comparisons with Neural Network result in ties rather than wins or losses, especially for LID3-Entropy and LID3-Percentile. Due to the limited

Table 5
Accuracy of LID3 based on different discretization methods and three other well-known machine learning algorithms

No.	C4.5	N.B.	N.N.	LID3-U	LID3-E	LID3-P
1	79.20 ± 1.53	89.46 ± 2.09	90.38 ± 1.18	83.80 ± 1.19	83.07 ± 3.22	86.23 ± 0.97
2	69.16 ± 4.14	71.26 ± 2.96	66.50 ± 3.48	73.06 ± 3.05	73.47 ± 2.66	73.06 ± 3.05
3	94.38 ± 1.42	96.28 ± 0.73	94.96 ± 0.80	96.43 ± 0.70	96.11 ± 0.78	96.11 ± 0.89
4	78.99 ± 2.23	85.36 ± 2.42	82.62 ± 3.18	85.41 ± 1.94	86.53 ± 1.28	85.59 ± 2.19
5	64.77 ± 5.10	45.99 ± 7.00	64.30 ± 3.38	65.96 ± 2.31	65.60 ± 2.57	65.87 ± 2.32
6	75.50 ± 3.79	84.24 ± 2.09	79.93 ± 3.99	76.71 ± 3.81	78.09 ± 3.58	77.96 ± 2.88
7	75.78 ± 3.16	84.00 ± 1.68	78.89 ± 3.05	76.52 ± 3.63	78.07 ± 3.63	79.04 ± 2.94
8	76.75 ± 4.68	83.25 ± 3.99	81.69 ± 2.48	82.95 ± 2.42	83.08 ± 2.82	83.08 ± 1.32
9	89.60 ± 2.13	82.97 ± 2.51	87.77 ± 2.88	88.98 ± 2.23	89.11 ± 2.30	88.01 ± 1.83
10	93.47 ± 3.23	94.53 ± 2.63	95.87 ± 2.70	96.00 ± 1.26	96.13 ± 1.60	96.40 ± 1.89
11	65.23 ± 3.86	55.41 ± 5.39	66.74 ± 4.89	58.73 ± 1.99	64.62 ± 2.80	69.25 ± 2.84
12	72.16 ± 2.80	75.05 ± 2.37	74.64 ± 1.41	76.22 ± 1.81	76.22 ± 1.85	76.54 ± 1.34
13	70.48 ± 0.00	70.19 ± 0.00	81.05 ± 0.00	86.54 ± 0.00	87.50 ± 0.00	89.42 ± 0.00
14	88.09 ± 4.14	96.29 ± 2.12	96.85 ± 1.57	95.33 ± 1.80	95.78 ± 1.80	95.89 ± 1.96

Table 6
Summary of comparisons of LID3 based on different discretization methods with three other well-known machine learning algorithms

	LID3-Uniform vs.	LID3-Entropy vs.	LID3-Percentile vs.
C4.5	9 wins–4 ties–1 losses	9 wins–5 ties–0 losses	10 wins–4 ties–0 losses
N.B.	3 wins–8 ties–3 losses	7 wins–4 ties–3 losses	7 wins–4 ties–3 losses
N.N.	5 wins–6 ties–3 losses	5 wins–8 ties–1 losses	5 wins–8 ties–1 losses

number and type of datasets we used for evaluation, we may not draw the strong conclusion that LID3 outperforms all the other three algorithms. However we can at least conclude that based on our experiments, the LID3 outperforms C4.5 and has equivalent performance to Naive Bayes and the Neural Networks. For the datasets with only numerical values, LID3 outperforms both C4.5 and Naive Bayes. Between different discretization methods, percentile-based and entropy-based approaches achieve better results than uniform discretization.

4. Forward merging of branches

In the last section, we showed that LID3 performs at least as well as and often better than three well-known classification algorithms across a range of datasets. However, even with only two fuzzy sets for discretization, the number of branches increases exponentially with the depth of the tree. Unfortunately,

the transparency of the LDT decreases with the increasing number of branches. To help to maintain transparency by generating more compact trees, a forward merging algorithm based on the LDT model is proposed in this section and experimental results are given to support the validity of our approach.

4.1. Forward merging algorithm

As we have seen in Section 3.5, given a continuous universe with a full fuzzy covering of N_F trapezoidal fuzzy sets, the focal elements can be represented by q ($q = 2N_F - 1$) triangular functions. If we vary x in an increasing manner, the focal elements then occur in the following natural order $F^1, \ldots, F^q$. We referred to F^i and F^{i+1} as being *adjacent focal elements* for $F^i \in \mathscr{F}$ and $i = 1, \ldots, q-1$. As described in Section 3, a node (if it is not terminated according to threshold probability T) is fully expanded with its focal elements. See Fig. 7a for instance, where the node is expanded into five leaves $LF_1, \ldots, LF_5$ with the following focal elements: $\{small\}$, $\{small, medium\}$, $\{medium\}$, $\{medium, large\}$ and $\{large\}$, where leaves next to each other (e.g., $\{small\}$ and $\{small, medium\}$) are adjacent focal elements. The branches whose leaves are adjacent focal elements are referred as *adjacent branches*. If any of two adjacent branches have sufficiently similar class probabilities according to some criteria, these two branches give similar classification results and therefore can then be merged into one branch in order to obtain a more compact tree (see Fig. 7b). We employ a *merging threshold* to determine whether or not two adjacent branches can be merged.

Definition 12 (*Merging threshold*). In a linguistic decision tree, if the maximum difference between class probabilities of two adjacent branches B_1 and B_2 is less than or equal to a given merging threshold T_m, then the two branches can be merged into one branch. Formally, if

$$T_m \geqslant \max_{c \in C}(|Pr(c|B_1) - Pr(c|B_2)|) \tag{13}$$

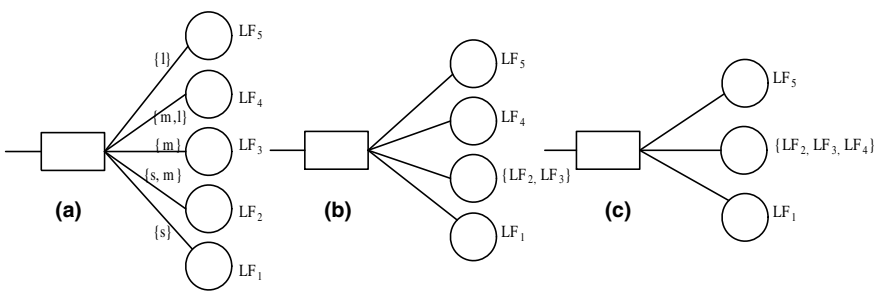

Fig. 7. Illustration of branch merging.

Table 7
Comparisons of accuracy Acc and the number of leaves (rules) N_R with different merging thresholds T_m across a set of UCI datasets

No.	N_F	$T_m = 0$		$T_m = 0.1$		$T_m = 0.2$		$T_m = 0.3$		$T_m = 0.4$	
		Acc	N_R	Acc	N_R	Acc	N_R	Acc	N_R	Acc	N_R
1	2	83.80	77	84.19	51	81.09	25	75.08	10	47.03	1
2	2	73.06	17	71.67	12	71.11	9	59.65	4	61.25	2
3	2	96.43	57	95.80	29	95.74	16	95.63	9	95.49	4
4	3	85.41	345	85.29	445	84.24	203	83.88	104	82.65	57
5	3	65.69	329	62.84	322	64.04	190	44.31	86	35.41	49
6	2	76.71	37	78.68	31	78.55	22	78.42	18	68.49	11
7	3	76.52	31	78.37	35	78.44	23	77.85	12	72.22	7
8	3	82.95	11	81.28	24	80.77	18	80.64	15	80.77	13
9	3	88.98	45	87.90	78	88.47	41	89.20	30	89.20	26
10	3	96.00	21	95.47	23	95.20	18	95.20	14	94.27	10
11	2	58.73	83	56.30	43	55.90	11	57.34	4	57.92	3
12	2	76.12	27	75.31	20	74.45	5	73.85	3	65.10	1
13	2	86.54	615	88.46	516	85.58	337	81.73	93	49.04	6
14	3	95.33	67	93.78	80	94.11	50	93.56	36	89.67	24

The results for $T_m = 0$ are obtained with $N_F = 2$ and results for other T_m values are obtained with N_F values listed in the second column of the table.

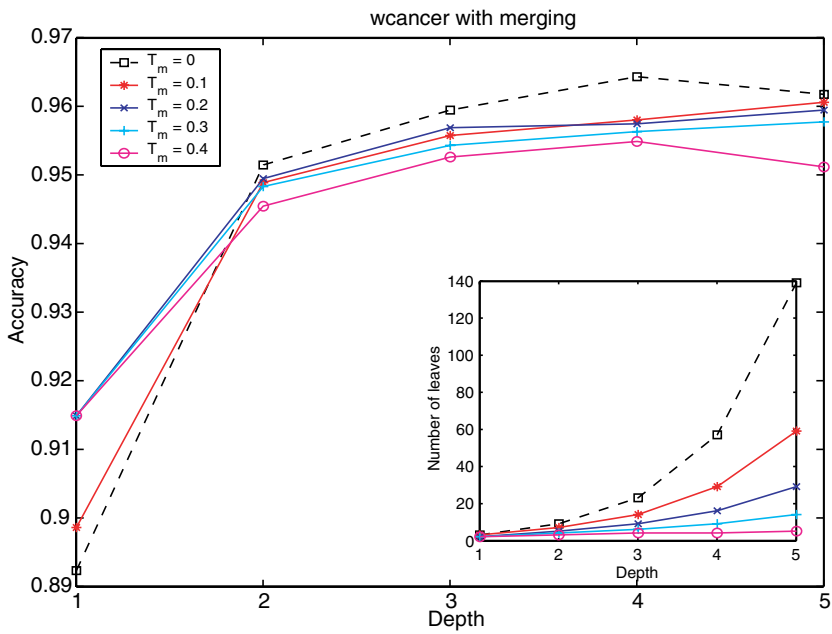

Fig. 8. The change in accuracy and number of leaves as T_m varies on the breast-w dataset with $N_F = 2$.

where $C = \{c_1, \ldots, c_m\}$ is the set of classes, then B_1 and B_2 can be merged into one branch MB.

Definition 13 (*Merged branch*). A merged branch MB with k nodes is defined as

$$MB = \langle \mathscr{M}_{j_1}, \ldots, \mathscr{M}_{j_k} \rangle$$

where $\mathscr{M}_j = \{F_j^1, \ldots, F_j^w\}$ is a set of focal elements such that F_j^i is adjacent to F_j^{i+1} for $i = 1, \ldots, w-1$, The associate mass for $\mathscr{M}_j$ is given by

$$m_x(\mathscr{M}_j) = \sum_{i=1}^{w} m_x(F_j^i) \tag{14}$$

where w is the number of merged adjacent focal elements for attribute j.

Based on Eqs. (3), (4) and (14) we use the following formula to calculate the class probabilities given a merged branch:

$$Pr(C_t|MB) = \frac{\sum_{i \in LD_t} \prod_{r=1}^{k} m_{x_{j_r}(i)}(M_{j_r})}{\sum_{i \in LD} \prod_{r=1}^{k} m_{x_{j_r}(i)}(M_{j_r})} \tag{15}$$

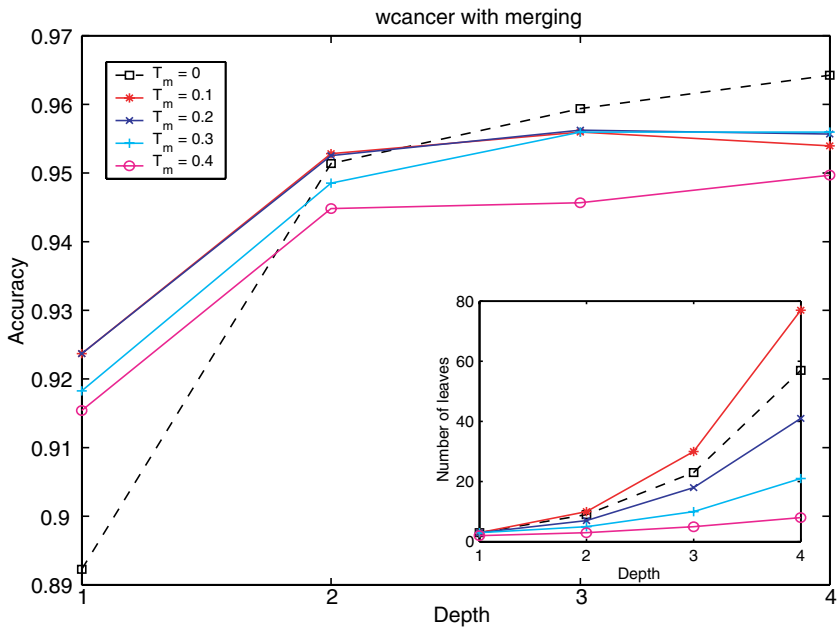

Fig. 9. The change in accuracy and number of leaves as T_m varies on the breast-w dataset with $N_F = 3$. While the dot trial $T_m = 0$ is with $N_F = 2$.

When the merging algorithm is applied in learning a linguistic decision tree, the adjacent branches meeting the merging criteria will be merged and reevaluated according to Eq. (15). Then the adjacent branches after the first round of merging will be examined in a further round of merging, until all adjacent branches cannot be merged further. We then proceed to the next depth. See Fig. 7b and c where leaves LF_2 and LF_3 are merged in the first round of merging, and LF_4 and $\{LF_2, LF_3\}$ are then merged further $\{LF_2, LF_3, LF_4\}$ if they meet the merging criteria in the second round of merging. The merged branches can be by compound expressions as described in a following section. The merging is applied as the tree develops from the root to the maximum depth and hence is referred to as *forward merging*.

4.2. Experimental studies

We tested the forward merging algorithm on the UCI datasets listed in Table 3 with 10 50–50 split experiments and the results are shown in Table 7. Obviously, there is a tradeoff between the algorithm accuracy and the algorithm transparency in terms of the number of leaves. The merging threshold T_m plays an important role in the accuracy–transparency tradeoff problem.

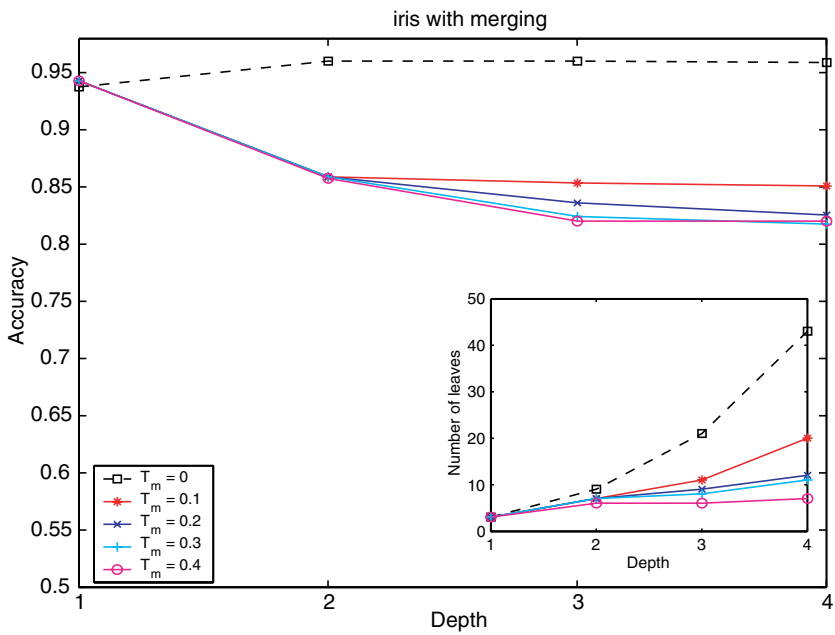

Fig. 10. The change in accuracy and number of leaves as T_m varies on the iris dataset with $N_F = 2$.

Algorithm accuracy tends to increase while algorithm transparency decreases with decreasing T_m and vice versa.

The number of fuzzy sets N_F in the merging algorithm is also a key parameter. Compared to $N_F = 3$, setting $N_F = 2$ can achieve better transparency, but for some datasets, with $N_F = 2$, the accuracy is greatly reduced although the resulting trees have significantly fewer of branches. For example, Figs. 8 and 9 show the change in test accuracy and the number of leaves (or the number of rules interpreted from a LDT) for different T_m on the breast-w dataset. Fig. 8 is with $N_F = 2$ and Fig. 9 with $N_F = 3$. Fig. 8 shows that the accuracy is not greatly influenced by merging, but the number of branches are greatly reduced. This is especially true for the curve marked by '+' corresponding to $T_m = 0.3$ where applying forward merging, the best accuracy (at the depth 4) is only reduced by approximately 1%, whereas, the number of branches is reduced by roughly 84%. However, in Fig. 9, at the depth 4 with $T_m = 0.3$, the accuracy also reduces about 1% but the number of branches only reduces by 55%. So, for this dataset, we should choose $N_F = 2$ rather than $N_F = 3$.

However, this is not always the case. For the dataset iris, the change in accuracy and the number of branches against depth with $N_F = 2$ and $N_F = 3$ are shown in Figs. 10 and 11, respectively. As we can see from Fig. 10, by applying

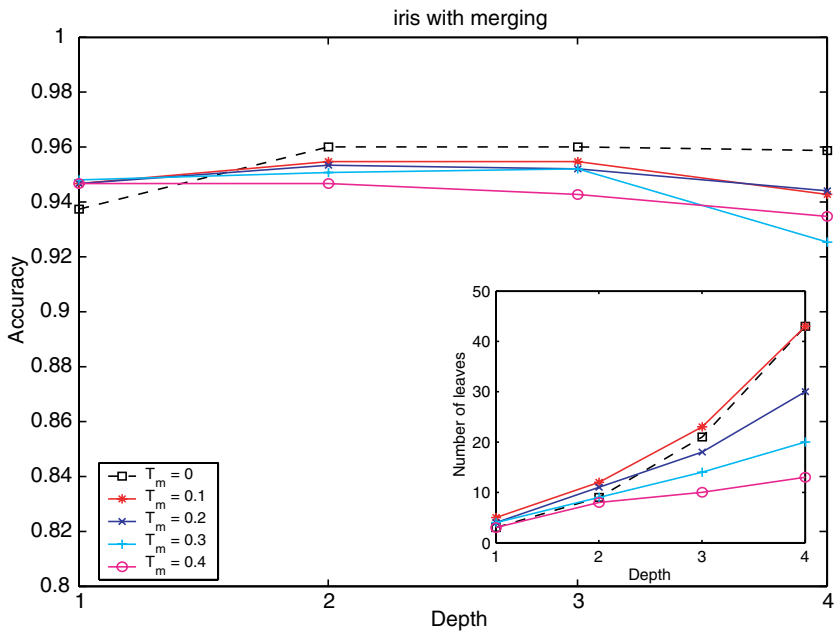

Fig. 11. The change in accuracy and number of leaves as T_m varies on the iris dataset with $N_F = 3$. While the dot trial $T_m = 0$ is with $N_F = 2$.

the forward merging algorithm, the accuracy is greatly changed. The best accuracy with merging is roughly 10% worse than non-merging algorithm. But for $N_F = 3$, as we can see from Fig. 11, the accuracy is not that greatly reduced compared to $N_F = 2$, and we still obtain a reduced number of branches, compared to the accuracy for $T_m = 0$ obtained from $N_F = 2$. In this case we should prefer $N_F = 3$.

Table 7 shows the results with optimal N_F and different T_m ranging from 0 to 0.4, where $T_m = 0$ represents no merging. *Acc* represents the average accuracy from 10 runs of experiments and N_R is the average number of rules (leaves). Unless otherwise stated, the results obtained in this section are with the threshold probability set to $T = 1$. The results for T_m from 0.1 to 0.4 are obtained at the depth where the optimal accuracy is found when $T_m = 0$. As we can see from the table, for most cases, the accuracy before and after merging are not significantly different but the number of branches are dramatically reduced. In some cases, the merging algorithm even outperforms the LID3 without merging. The possible reason for this is because the merging algorithm generates self-adapting granularities based on class probabilities. Compared to other methods that discretize attributes independently, merging may generate a more reasonable tree with more appropriate information granules.

5. Linguistic reasoning

In this section we use label semantics to provide a linguistic interpretation for LDTs. We also use this framework to show how LDTs can be used to classify data with *linguistic constraints* on attributes. In addition, a method for classification on *fuzzy data* is proposed and supported with empirical studies on a toy problem. Basically, we interpret the main logical connectives as follows: $\neg L$ means that L is not an appropriate label, $L_1 \wedge L_2$ means that both L_1 and L_2 are appropriate labels, $L_1 \vee L_2$ means that either L_1 or L_2 are appropriate labels, and $L_1 \rightarrow L_2$ means that L_2 is an appropriate label whenever L_1 is. If we consider label expressions formed from LA by recursive application of the connectives then an expression θ identifies a set of *possible label sets* according to the following λ-function.

Definition 14 (λ-*function*). Let θ and φ be expressions generated by recursive application of the connectives $\neg$, $\vee$, $\wedge$ and $\rightarrow$ to the elements of LA. Then the set of possible label sets defined by a linguistic expression can be determined recursively as follows:

(i) $\lambda(L_i(x)) = \{S \subseteq \mathscr{F} | \{L_i\} \subseteq S\}$
(ii) $\lambda(\neg\theta) = \overline{\lambda(\theta)}$

(iii) $\lambda(\theta \wedge \varphi) = \lambda(\theta) \cap \lambda(\varphi)$
(iv) $\lambda(\theta \vee \varphi) = \lambda(\theta) \cup \lambda(\varphi)$
(v) $\lambda(\theta \rightarrow \varphi) = \overline{\lambda(\theta)} \cup \lambda(\varphi)$

It should also be noted that the λ-function provides us with notion of logical equivalence for label expressions

$$\theta \equiv_{\mathscr{F}} \varphi \iff \lambda(\theta) = \lambda(\varphi)$$

Intuitively, $\lambda(\theta)$ corresponds to those subsets of $\mathscr{F}$ identified as being possible values of D_x by expression θ. In this sense the imprecise linguistic restriction 'x is θ' on x corresponds to the strict constraint $D_x \in \lambda(\theta)$ on D_x. Hence, we can view label descriptions as an alternative to linguistic variables as a means of encoding linguistic constraints [9]. Here we consider the linguistic constraints take the form of $\theta = \langle x_1 \text{ is } \theta_1, \ldots, x_n \text{ is } \theta_n \rangle$, where θ_j represents a label expression based on $LA_j : j = 1, \ldots, n$.

Example 5. Given a continuous variable x and $LA = \{small, medium, large\}$, suppose we are told that "x is **not large** but it is **small to medium**". This constraint can be interpreted as the logical expression

$$\theta_x = \neg large \wedge (small \vee medium)$$

According to Definition 14, the possible label sets of the given linguistic constraint θ_x are

$$\begin{aligned}\lambda(\theta_x) &= \lambda(\neg large \wedge (small \vee medium)) \\ &= \{\{small\}, \{small, medium\}, \{medium\}\}\end{aligned}$$

5.1. Linguistic interpretation

Based on the inverse of the λ-function (Definition 14), a set of linguistic rules (or label expressions) can be obtained from a set of possible label sets. For example, suppose we are given the possible label sets $\{\{small\}, \{small, medium\}, \{medium\}\}$, which does not have an immediately obvious interpretation. However using the α-function (see below), we can convert this set into a corresponding linguistic expression $\neg large \wedge (small \vee medium)$ or its logical equivalence. More details about linguistic reasoning can be found in [8].

Definition 15 (α-*Function*).

$$\forall F \in \mathscr{F} \quad \text{let } \mathscr{N}(F) = \left(\bigcup_{F' \in \mathscr{F} : F' \supseteq F} F' \right) - F \tag{16}$$

$$\text{then } \alpha_F = \left(\bigwedge_{L\in F} L\right) \wedge \left(\bigwedge_{L\in \mathscr{N}(F)} \neg L\right) \tag{17}$$

We can then map a set of focal sets to label expressions based on the α-function as follows:

$$\forall R \in \mathscr{F} \quad \theta_R = \bigvee_{F\in R} \alpha_F \quad \text{where } \lambda(\theta_R) = R \tag{18}$$

The motivation of this mapping is a follows. Given a focal set $\{s, m\}$ this states that the labels appropriate to describe the attribute are exactly *small* and *medium*. Hence, they include s and m and exclude all other labels that occur in focal sets that are supersets of $\{s, m\}$. Given a set of focal sets $\{\{s, m\}, \{m\}\}$ this provides the information that the set of labels is either $\{s, m\}$ or $\{m\}$ and hence the sentence providing the same information should be the disjunction of the α sentences for both focal sets. The following example gives the calculation of the α-function.

Example 6. Let $LA = \{very\ small\ (vs), small\ (s), medium\ (m), large\ (l), very\ large\ (vl)\}$ and $\mathscr{F} = \{\{vs, s\}, \{s\}, \{s, m\}, \{m\}, \{m, l\}, \{l\}, \{l, vl\}\}$. For calculating $\alpha_{\{l\}}$, we obtain Also we can also obtain

$$F' \in \mathscr{F} : F' \supseteq \{l\} = \{\{m, l\}, \{l\}, \{l, vl\}\} = \{m, l, vl\}$$

$$\mathscr{N}(\{l\}) = \left(\bigcup_{F'\in\mathscr{F}:F'\supseteq F} F'\right) - \{l\} = \{l, vl, m\} - \{l\} = \{vl, m\}$$

$$\alpha_{\{l\}} = \left(\bigwedge_{L\in F} L\right) \wedge \left(\bigwedge_{L\in\mathscr{N}(F)} \neg L\right) = \{l\} \wedge (\neg m \wedge \neg vl) = \neg m \wedge l \wedge \neg vl$$

Also we can also obtain

$$\alpha_{\{m,l\}} = m \wedge l, \quad \alpha_{\{l,vl\}} = l \wedge vl$$

Hence, a set of label sets $\{\{m, l\}, \{l\}, \{l, vl\}\}$ can be represented by a linguistic expression as follows:

$$\begin{aligned}\theta_{\{\{m,l\},\{l\},\{l,vl\}\}} &= \alpha_{\{m,l\}} \vee \alpha_{\{l\}} \vee \alpha_{\{l,vl\}} \\ &= (m \wedge l) \vee (\neg m \wedge l \neg vl) \vee (l \wedge vl) \equiv_{\mathscr{F}} large\end{aligned}$$

where '$\equiv_{\mathscr{F}}$' represents logical equivalence (see Definition 14).

As discussed in the last section, a merged LDT was obtained from a real-world dataset 'iris' at the depth 2 when $T_m = 0.3$ and where $LA_j = \{small_j(s_j), medium_j(m_j), large_j(l_j) \,|\, j = 1, \ldots, 4\}$.

$$
\begin{aligned}
LDT_{M\text{-}iris} &= \{MB_1, MB_2, MB_3, MB_4, MB_5, MB_6, MB_7, MB_8\} \\
&= \{\langle\{\{s_3\}\rangle, 1.0000, 0.0000, 0.0000\rangle \\
&\quad \langle\{\{\{s_3, m_3\}, \{m_3\}\}, \{s_4\}\rangle, 1.0000, 0.0000, 0.0000\rangle \\
&\quad \langle\{\{\{s_3, m_3\}, \{m_3\}\}, \{\{s_4, m_4\}, \{m_4\}\}\rangle, 0.0008, 0.9992, 0.0000\rangle \\
&\quad \langle\{\{\{s_3, m_3\}, \{m_3\}\}, \{m_4, l_4\}\rangle, 0.0000, 0.5106, 0.494\rangle \\
&\quad \langle\{\{\{s_3, m_3\}, \{m_3\}\}, \{l_4\}\rangle, 0.0000, 0.0556, 0.9444\rangle \\
&\quad \langle\{\{\{m_3, l_3\}, \{l_3\}\}, \{s_4\}\rangle, 0.3333, 0.3333, 0.3333\rangle \\
&\quad \langle\{\{\{m_3, l_3\}, \{l_3\}\}, \{\{s_4, m_4\}, \{m_4\}\}\rangle, 0.000, 0.8423, 0.1577\rangle \\
&\quad \langle\{\{\{m_3, l_3\}, \{l_3\}\}, \{\{m_4, l_4\}, \{l_4\}\rangle, 0.000, 0.0913, 0.9087\rangle\}
\end{aligned}
$$

We can then translate this tree into a set of linguistic expressions as follows:

$$
\begin{aligned}
LDT_{M\text{-}iris} &= \{\langle\langle s_3 \wedge \neg(m_3 \vee l_3)\rangle, 1.0000, 0.0000, 0.0000\rangle \\
&\quad \langle\langle m_3 \wedge \neg l_3, s_4 \wedge \neg(m_4 \vee l_4)\rangle, 1.0000, 0.0000, 0.0000\rangle \\
&\quad \langle\langle m_3 \wedge \neg l_3, m_4 \wedge \neg l_4\rangle, 0.0008, 0.9992, 0.0000\rangle \\
&\quad \langle\langle m_3 \wedge \neg l_3, \neg s_4 \wedge m_4 \wedge l_4\rangle, 0.0000, 0.5106, 0.4894\rangle \\
&\quad \langle\langle m_3 \wedge \neg l_3, (s_4 \vee m_4) \wedge l_4\rangle, 0.0000, 0.0556, 0.9444\rangle \\
&\quad \langle\langle l_3, s_4 \wedge \neg(m_4 \vee l_4)\rangle, 0.3333, 0.3333, 0.3333\rangle \\
&\quad \langle\langle l_3, m_4 \wedge \neg l_4\rangle, 0.000, 0.8423, 0.1577\rangle \\
&\quad \langle\langle l_3, l_4\rangle, 0.000, 0.0913, 0.9087\rangle\}
\end{aligned}
$$

Furthermore, the tree itself can be rewritten as a set of fuzzy rules. For example branch 2 corresponds to the rule: IF attribute 3 is **medium but not large** and attribute 4 is **only small**, THEN the class probabilities given this branches are (1.0000, 0.0000, 0.0000).

5.2. Classification under linguistic constraint

Consider the vector of linguistic constraint $\vec{\theta} = \langle\theta_1, \ldots, \theta_n\rangle$, where θ_j is the linguistic constraints on attribute j. We can evaluate a probability value for class C_t conditional on this information using a given linguistic decision tree as follows. The mass assignment given a linguistic constraint θ is evaluated by

$$
\forall F_j \in \mathscr{F}_j, \quad m_{\theta_j}(F_j) = \begin{cases} \frac{pm(F_j)}{\sum_{F_j \in \lambda(\theta_j)} pm(F_j)} & \text{if } F_j \in \lambda(\theta_j) \\ 0 & \text{otherwise} \end{cases} \tag{19}
$$

where $pm(F_j)$ is the prior mass for focal elements $F_j \in \mathscr{F}_j$ derived from the prior distribution $p(x_j)$ on Ω_j as follows:

$$
pm(F_j) = \int_{\Omega_j} m_x(F_j) p(x_j) \mathrm{d}x_j \tag{20}
$$

Usually, we assume that $p(x_j)$ is the uniform distribution over Ω_j so that

$$pm(F_j) \propto \int_{\Omega_j} m_x(F_j)\mathrm{d}x_j \tag{21}$$

More details on calculation of mass assignment given a linguistic constraint are given in Example 7. For branch B with s nodes, the probability of B given $\vec{\theta}$ is evaluated by

$$Pr(B|\vec{\theta}) = \prod_{r=1}^{k} m_{\theta_{jr}}(F_{jr}) \tag{22}$$

and therefore, by Jeffrey's rule [7]

$$Pr(C_t|\vec{\theta}) = \sum_{v=1}^{s} Pr(C_t|B_v)Pr(B_v|\vec{\theta}) \tag{23}$$

Example 7. Given the LDT in Example 4 Section 3.3 suppose we know that for a particular data element "x_1 **is not large** and x_2 **is small**". We then can translate this knowledge into the following linguistic constraint vector:

$$\vec{\theta} = \langle \theta_1, \theta_2 \rangle = \langle \neg large_1, small_2 \rangle$$

By applying the λ function (Definition 14), we can generate the associated label sets, so that

$$\lambda(\neg large_1) = \{\{small_1\}\}, \quad \lambda(small_2) = \{\{small_2\}, \{small_2, large_2\}\}$$

suppose the prior mass assignments are

$$pm_1 = \{small_1\} : 0.4, \{small_1, large_1\} : 0.3, \{large_1\} : 0.3$$
$$pm_2 = \{small_2\} : 0.3, \{small_2, large_2\} : 0.2, \{large_2\} : 0.5$$

From this, according to Eq. (19) we obtain that

$$m_{\theta_1} = \{small_1\} : 0.4/0.4 = \{small_1\} : 1$$
$$m_{\theta_2} = \{small_2\} : 0.3/(0.3 + 0.2), \{small_2, large_2\} : 0.2/0.2(0.2 + 0.3)$$
$$= \{small_2\} : 0.6, \{small_2, large_2\} : 0.4$$

This gives

$$Pr(B_1|\vec{\theta}) = m_{\theta_1}(\{small_1\}) \times m_{\theta_2}(\{small_2\}) = 1 \times 0.6 = 0.6$$
$$Pr(B_2|\vec{\theta}) = m_{\theta_1}(\{small_1\}) \times m_{\theta_2}(\{small_2, large_2\}) = 1 \times 0.4 = 0.4$$
$$Pr(B_3|\vec{\theta}) = Pr(B_4|\vec{\theta}) = Pr(B_5|\vec{\theta}) = Pr(B_6|\vec{\theta}) = Pr(B_7|\vec{\theta}) = 0$$

Hence, according to Jeffrey's rule

$$Pr(C_1|\vec{\theta}) = \sum_{v=1}^{7} Pr(B_v|\vec{\theta})Pr(C_1|B_v)$$
$$= \sum_{v=1,2} Pr(B_v|\vec{\theta})Pr(C_1|B_v) = 0.6 \times 0.3 + 0.4 \times 0.5 = 0.38$$
$$Pr(C_2|\vec{\theta}) = \sum_{v=1}^{7} Pr(B_v|\vec{\theta})Pr(C_2|B_v)$$
$$= \sum_{v=1,2} Pr(B_v|\vec{\theta})Pr(C_2|B_v) = 0.6 \times 0.7 + 0.4 \times 0.5 = 0.62$$

The methodology for classification under linguistic constraints allows us to fuse the background knowledge in linguistic form into classification. This is one of the advantages of using high-level knowledge representation language models such as label semantics.

5.3. Classification given fuzzy data

In previous sections LDTs have only been used to classify crisp data where objects are described in terms of precise attribute values. However, in many real-world applications limitations of measurement accuracy means that only imprecise values can be realistically obtained. In this section we introduce the idea of fuzzy data and show how LDTs can be used for classification in this context. Formally, a fuzzy database is defined to be a set of elements or objects each described by linguistic expressions rather than crisp values. In other words

$$\boldsymbol{FD} = \{\langle \theta_1(i), \ldots, \theta_n(i)\rangle : i = 1, \ldots, N\}$$

Currently there are very few benchmark problems of this kind with fuzzy attribute values. This is because, traditionally only crisp data values are recorded even in cases where this is inappropriate. Hence, we have generated a fuzzy database from a toy problem where the aim is to identify the interior of a figure of eight shape. Specifically, a figure of eight shape was generated according to the equation $x = 2^{(-0.5)}(\sin(2t) - \sin(t)$ and $y = 2^{(-0.5)}(\sin(2t) + \sin(t))$ where $t \in [0, 2\pi]$ (see Fig. 13). Points in $[-1.6, 1.6]^2$ are classified as legal if they lie within the 'eight' shape (marked with ×) and illegal if they lie outside (marked with points).

To form the fuzzy database we first generated a crisp database by uniformly sampling 961 points across $[-1.6, 1.6]^2$. Then each data vector $\langle x_1, x_2\rangle$was converted to a vector of linguistic expressions $\langle \theta_1, \theta_2\rangle$ as follows: $\theta_j = \theta_{R_j}$ where $R_j = \{F \in \mathscr{F}_j : m_{x_j}(F) > 0\}$ A LDT was then learnt by applying the LID3 algorithm to the crisp database. This tree was then used to classify both the crisp and fuzzy data. The results are shown in Table 8 and the results with $N_F = 7$ are shown in Fig. 12.

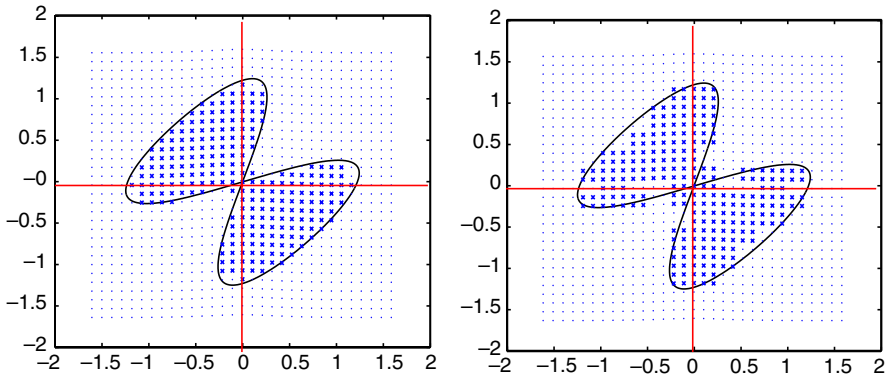

Fig. 12. Classification on crisp dataset (left) and fuzzy data without masses (right), where each attribute is discretized uniformly by 7 fuzzy sets.

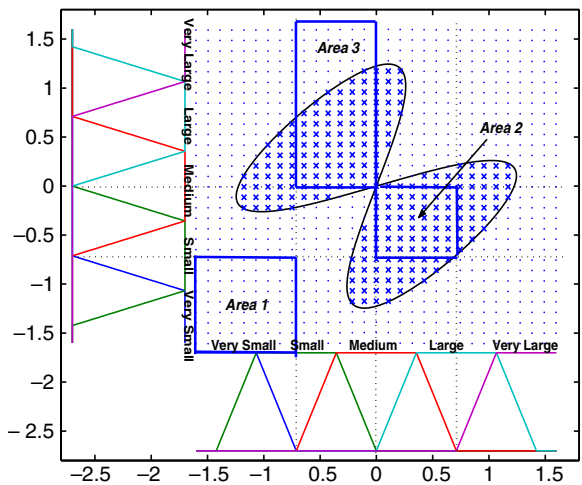

Fig. 13. Testing on the 'eight' problem with linguistic constraints $\vec{\theta}$, where each attribute is discretized by five trapezoidal fuzzy sets: *very small*, *small*, *medium*, *large* and *very large*.

Table 8
Classification accuracy based on crisp data and fuzzy data on the 'eight' problem

	$N_F = 3$	$N_F = 4$	$N_F = 5$	$N_F = 6$	$N_F = 7$
Crisp data	87.72%	94.17%	95.94%	97.29%	98.54%
Fuzzy data	79.29%	85.85%	89.39%	94.17%	95.01%

As we can see from Table 8, our model gives a reasonable approximation of the legal data area, though it is not as accurate as testing on crisp data. The

accuracy increases with N_F the number of fuzzy sets used for discretization. These results show that LDT model can perform well in dealing with fuzzy and ambiguous data. Here the 'eight' problem is also used for testing classification with linguistic constraints in the following example.

Example 8. Suppose a *LDT* is trained on the 'eight' database where each attribute is discretized by five fuzzy sets uniformly: *very small* (*vs*), *small* (*s*), *medium* (*m*), *large* (*l*) and *very large* (*vl*). Further, suppose we are given the following description of data points:

$$\vec{\theta}_1 = \langle x \text{ is } vs \vee s \wedge \neg m, y \text{ is } vs \vee s \wedge \neg m \rangle$$

$$\vec{\theta}_2 = \langle x \text{ is } m \wedge l, y \text{ is } s \wedge m \rangle$$

$$\vec{\theta}_3 = \langle x \text{ is } s \wedge m, y \text{ is } l \vee vl \rangle$$

Experimental results obtained based on the approach introduced in Section 5.2 are as follows:

$$Pr(C_1|\vec{\theta}_1) = 1.000, \quad Pr(C_2|\vec{\theta}_1) = 0.000$$

$$Pr(C_1|\vec{\theta}_2) = 0.000, \quad Pr(C_2|\vec{\theta}_2) = 1.000$$

$$Pr(C_1|\vec{\theta}_3) = 0.428, \quad Pr(C_2|\vec{\theta}_3) = 0.572$$

As we can see from Fig. 13, the above 3 linguistic constraints roughly correspond to the area 1, 2 and 3, respectively. By considering the occurrence of legal and illegal examples within these areas, we can verify the correctness of our approach.

6. Conclusions

In this paper, a decision tree learning algorithm is proposed based on a random set framework for Computing with Words referred to as label semantics. LID3 is proposed as a modified ID3 algorithm based on label semantics. Unlike classical decision trees, the new algorithm uses probability estimation based on linguistic labels. The linguistic labels are based on fuzzy discretization using a number of different methods including uniform partitioning, a percentile-based partitioning and an entropy-based partitioning. We found that the percentile-based discretization and entropy-based discretization outperform uniform discretization, but no statistically significance was found. By testing our new model on real-world datasets and comparing with three well-know machine learning algorithms, we found that LID3 outperformed C4.5 on all given datasets and outperforms Naive Bayes on datasets with numerical

attributes only. Also it has equivalent classification accuracy and better transparency when compared to back propagation Neural Networks.

In order to obtain a compact tree, a forward merging algorithm was proposed and the experimental results show that the number of branches can be greatly reduced without a significant loss in accuracy. In the last section, a formal method for interpreting a linguistic decision tree as a set of logical rules of labels is proposed. It is also show how LDT can be used to classify fuzzy data where objects are only described in terms of linguistic expressions. Future work will focus on extending the LDT model from classification problems to prediction problems and to allow information fusion so that background knowledge can be incorporated into the tree induction process.

References

[1] J.F. Baldwin, T.P. Martin, B.W. Pilsworth, Fril-fuzzy and evidential reasoning in artificial intelligence, John Wiley, 1995.
[2] J.F. Baldwin, J. Lawry, T.P. Martin. Mass assignment fuzzy ID3 with applications, in: Proceedings of the Unicom Workshop on Fuzzy Logic: Applications and Future Directions, pp. 278–294, London, 1997.
[3] C. Blake, C.J. Merz, UCI machine learning repository, <http://www.ics.uci.edu/~mlearn/MLRepository.html>.
[4] U.M. Fayyad, K.B. Irani, Multi-interval discretization of continuous-valued attributes for classification learning, in: Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence, vol. 2, Chambery France, August–September 1993.
[5] I.R. Goodman. Fuzzy sets as equivalence classes of random sets, in: R. Yager (Ed.), Fuzzy Sets and Possibility Theory, 1982, pp. 327–342.
[6] C.Z. Janikow, Fuzzy decision trees: issues and methods, IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics 28 (1) (1998).
[7] R.C. Jeffrey, The logic of decision, Gordon & Breach, New York, 1965.
[8] J. Lawry, A framework for linguistic modelling, Artificial Intelligence 155 (2004) 1–39.
[9] J. Lawry, Query evaluation from linguistic prototypes, in: Proceedings of 10'th IEEE International Conference on Fuzzy Systems, 2001.
[10] J. Lawry, A voting mechanism for fuzzy logic, International Journal of Approximate Reasoning 19 (1998) 315–333.
[11] J. Lawry, Label Semantics: a formal framework for modelling with words, in: Symbolic and Quantitative Approaches to Reasoning with UncertaintyLecture Notes in Artificial Intelligence, Springer-Verlag, 2001, pp. 374–384.
[12] T. Mitchell, Machine Learning, McGraw-Hill, New York, 1997.
[13] C. Olaru, L. Wehenkel, A complete fuzzy decision tree technique, Fuzzy Sets and Systems 138 (2003) 221–254.
[14] Y. Peng, P.A. Flach, Soft discretization to enhance the continuous decision trees, ECML/PKDD Workshop: IDDM, 2001.
[15] F. Provost, P. Domingos, Tree induction for probability-based ranking, Machine Learning 52 (2003) 199–215.
[16] J.R. Quinlan, Induction of decision trees, Machine Learning 1 (1986) 81–106.
[17] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, San Mateo, 1993.

[18] N.J. Randon, J. Lawry. Linguistic modelling using a semi-Naive Bayes framework, IPMU-2002, Annecy, France, 2002.
[19] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques with JavaMorgan Kaufmann, Morgan Kaufmann, 1999, Available from: <http://www.cs.waikato.ac.nz/~ml/weka/>.
[20] L.A. Zadeh, Fuzzy logic = computing with words, IEEE Transaction on Fuzzy Systems 4 (2) (1996) 103–111.