Information Measures in Fuzzy Decision Trees

Xiaomeng Wang
Department of Computer Science
University of Magdeburg
39106 Magdeburg, Germany
E-mail: xwang@iws.cs.uni-magdeburg.de

Christian Borgelt
Department of Computer Science
University of Magdeburg
39106 Magdeburg, Germany
E-mail: borgelt@iws.cs.uni-magdeburg.de

***Abstract*— Decision trees are a popular form of classification models. It is well known that classical trees lack the ability of modelling vagueness. By connecting fuzzy systems and classical decision trees, we try to achieve classifiers that can model vagueness and are comprehensible. We discuss the core problem of how to compute the information measure used in the induction of fuzzy trees and propose some improvements. In addition, we consider fuzzy rule bases derived from fuzzy decision trees and present some heuristic strategies to prune them. We report the results of experiments in which we compare our approach to other well-known classification methods.**

I. Introduction

Data analysis is the process of computing various summaries and derived values from a given collection of data [1]. Here we discuss data analysis in the classification context.

Different classification models have individual strengths and weaknesses: sometimes a classifier is good for prediction but difficult to understand, like neural networks. Or a model works well in crisp domains, but cannot model vagueness, like classical decision trees.

In this paper we combine fuzzy theory with classical decision trees in order to learn a classification model, which is able to handle vagueness and also comprehensible.

In the past several variants of fuzzy decision trees were introduced by different authors. Boyen and Wenkel [5] presented the automatic induction of binary fuzzy trees based on a new class of discrimination quality measures. Janikow [6] adapted the well-known ID3 algorithm so that it works with fuzzy sets.

In this paper, we also adapted the ID3 algorithm to construct fuzzy decision trees and borrowed some basic ideas from [6]. In Section II we present our algorithm and examine in detail the core problem of how to compute the information measure in the attribute selection step. Going beyond Janikow's work we consider how to extract a fuzzy rule base from the induced fuzzy tree, and study heuristics to simplify it. In addition we discuss how to treat missing values. In Section III we report experimental results obtained with an implementation of our algorithm and compare them to those of some popular classifiers.

II. Fuzzy Decision Tree

A. Tree induction

In this paper we focus on the induction of a fuzzy decision tree (FDCT) on continuous attributes. Before the tree induction a fuzzy partition has to be created for each attribute. The fuzzy sets of these partitions will be used as fuzzy tests in the nodes of the fuzzy tree. To initialize these fuzzy partitions we adopted an existing algorithm, which creates them either completely automatically based on a given data set (called "automatic partitioning"), or based on a user specification of the shape and number of the membership functions (called "individual partitioning"). In the latter case the fuzzy sets are distributed evenly over the entire domain of each attribute. Here we assume the fuzzy partitions of the input variables are given.

Like classical decision trees with the ID3 algorithm, fuzzy decision trees are constructed in a top-down manner by recursive partitioning of the training set into subsets. We assume the basic algorithm to be known, and only point out some particular features of the fuzzy tree learning:

1) **The membership degree of examples**
   The membership degree of an example to an example set is not a binary element from $\{0, 1\}$ (as in classical decision trees), but from the interval $[0, 1]$. In each node, an example has a different membership degree to the current example set, and this degree is calculated from the conjunctive combination of the membership degrees of the example to the fuzzy sets along the path to the node and its membership degrees to the classes, where different t-norms ($\top$) can be used for this combination.
2) **Selection of test attributes**
   This point will be discussed in detail in the following subsections.
3) **Fuzzy tests**
   As mentioned above, in the inner nodes fuzzy tests are used instead of crisp tests. A fuzzy test of an attribute means to determine the membership degree of the value of an attribute to a fuzzy set.
4) **Stop criteria**
   Usually classical tree learning is terminated if all attributes are already used on the current path; or if all examples in the current node belong to the same class. Here we add another condition, namely whether the information measure is below a specified threshold. In FDCT any example can occur in any node with any membership degree. Thus in general more examples are considered per node and fuzzy trees are usually larger

than classical trees. The threshold defined here enables a user to control the tree growth, so that unnecessary nodes are not added. The experiments prove that an adequate threshold helps not only to avoid overfitting, but also to decrease the complexity of the tree.

B. Notation

For the formal discussion of attribute selection we introduce the following notation:

- $C = \{C_1, ..., C_m\}$ is the set of possible classes.
- $A = \{A_1, A_2, ..., A_n\}$ is a set of input attributes with domains $\text{dom}(A_i)$, $1 \leq i \leq n$.
- For each variable $A_i \in A$, $1 \leq i \leq n$:
  - $u^i \in \text{dom}(A_i)$ is a crisp value of attribute A_i.
  - D_i is the fuzzy partition of A_i
  - a_p^i denotes the fuzzy set (the linguistic term) p for the attribute A_i. For example $a_{low}^{pressure}$ means the fuzzy set "*low*" of the attribute "*pressure*".
- We consider a reference set $E = \{e_1, \ldots, e_s\}$ of examples $e_k = (\vec{u}_k, \vec{y}_k)$, $1 \leq k \leq s$. $\vec{u}_k$ is the input vector, $\vec{y}_k \in [0,1]^m$ the output vector of e_k. u_k^i (i.e. the i-th element of $\vec{u}_k$) is the (crisp) value of attribute A_i in example e_k. y_k^j (i.e. the j-th element of $\vec{y}_k$) is the membership degree of example e_k to the class C_j.
  The crisp classification of e_k can be computed as $\text{class}(e_k) = \text{argmax}_{j:0\leq j\leq m}\{y_k^j\}$. (Note that a classification problem can easily be extended for a continuous target variable by using fuzzy sets instead of crisp classes C_j. However, we confine ourselves to targets with a finite number of classes.)
- The training set is a fuzzy set over E defined by initial confidence weights $\chi = \{\chi_1, \ldots, \chi_s\}$, $\forall k, 0 \leq k \leq s : 0 \leq \chi_k \leq 1$. If no such information is provided by a user, we set $\forall k, 1 \leq k \leq s : \chi_k = 1$.
- For each node N in the fuzzy tree, $\chi^N = \{\chi_1^N, \ldots, \chi_s^N\}$ is the fuzzy example set (a fuzzy set over E) in N. In the root, this fuzzy example set coincides with the training set, i.e., $\forall k, 1 \leq k \leq s : \chi_k^{\text{Root}} = \chi_k$.
- $Z_j^N = \sum_{k=1}^s \top(\chi_k^N, y_k^j)$ stands for the example counter for class C_j in node N. $Z^N = \sum_{j=1}^s Z_j^N$ is the total counter for the examples of all classes. (More details about the computation and the interpretation of membership degrees as case weights are given below.)
- $I(\chi^N)$ denotes the entropy of the class distribution w.r.t. the fuzzy example set χ^N in node N. $I(\chi^N|A_i)$ is the weighted sum of entropies from all child nodes, if A_i is used as the test attribute in node N.
- $\text{Gain}(\chi^N, A_i) = I(\chi^N) - I(\chi^N|A_i)$ is the information gain w.r.t. attribute A_i, which is the first of the two attribute selection measures we consider.
- $\text{SplitI}(\chi^N, A_i)$ denotes the split information—the entropy w.r.t. the value distribution of attribute A_i (instead of the class distribution).
- $\text{GainR}(\chi^N, A_i) = \text{Gain}(\chi^N, A_i)/\text{SplitI}(\chi^N, A_i)$ is the information gain ratio w.r.t. attribute A_i, which is the second attribute selection measure we consider.

TABLE I
EXAMPLE1

before split	C_1		C_2		
	e_1	e_2	e_3	e_4	e_5
χ_j^N	1.0	1.0	1.0	1.0	1.0
Z^N	2.0		3.0		

TABLE II
EXAMPLE1

after split	C_1		C_2		
	e_1	e_2	e_3	e_4	e_5
small	0.8	0.7	1.0	0	1.0
large	0.6	0.9	0	1.0	0
$\hat{\chi}_j^N$	1.4	1.6	1.0	1.0	1.0
$\hat{Z}^N$	3.0		3.0		

C. Problems with test attributes selection

A standard method to select a test attribute in classical decision tree induction is to choose the attribute that yields the highest information gain. In this subsection we discuss the problems that occur if we apply this measure in fuzzy decision tree induction.

The definition of information gain is based on probability theory. Its value, as we know, can never be negative (a prove can be found, for instance, in [3]). However, depending on the computation, in FDCT negative information gain is possible. This phenomenon occurs due to the following two reasons, both of which can lead to a situation in which the sum of the weights of the example cases before and after a split and thus the class frequency distributions differ.

1) In fuzzy logic, the sum of the membership degrees of a value to the fuzzy sets of its variable can differ from 1, depending on how the fuzzy sets overlap.
2) Probability theory prescribes to use the product to express a (conditional) conjunction (i.e., $P(X \wedge Y) = P(X \mid Y) \cdot P(Y)$), whereas fuzzy logic offers other possibilities besides the product, for example $\top_{\min}(a, b)$ and $\top_{\text{Łuka}}(a, b)$.

Example 1 The following example illustrates point 1)

- Let $E = \{e_1, e_2, e_3, e_4, e_5\}$ be a reference set with examples coming from two classes C_1 and C_2, where $y_1^1 = y_2^1 = 1$, $y_1^2 = y_2^2 = 0$ (i.e., e_1, e_2 belong exclusively to C_1), and $y_3^1 = y_4^1 = y_5^1 = 0$, $y_3^2 = y_4^2 = y_5^2 = 1$ (i.e., e_3, e_4, e_5 belong exclusively to C_2).
- Let the membership degree of each example to the current fuzzy example set in node N be $\chi_k^N = 1$, $1 \leq k \leq 5$ (see Table I). These membership degrees are interpreted as case weights and thus we have $p_{C_1} : p_{C_2} = 2 : 3$ as the frequency distribution of the classes.
- After splitting the training set according to the fuzzy sets of attribute A — *small* and *large* — we obtain Table II showing the membership degree of each example. Since

$\chi^N_k = 1$, $1 \leq k \leq 5$, the membership degree of each example is the same as its membership degree to the respective fuzzy set[1], e.g. $\chi^{N|\text{small}}_1 = 0.8$ and $\chi^{N|\text{large}}_1 = 0.6$. If we interpret the membership degrees to the fuzzy example set as case weights we can sum the weights for the subsets to obtain the weights for the whole set (as it is possible in classical decision tree induction). In this way we obtain case weights (**not** membership degrees, because they may be greater than 1) $\hat{\chi}^N$, for which we have $\hat{\chi}^N_1 = 0.8 + 0.6 = 1.4$, $\hat{\chi}^N_2 = 0.7 + 0.9 = 1.6$, and $\hat{\chi}^N_3 = \hat{\chi}^N_4 = \hat{\chi}^N_5 = 1$. The frequency distribution of the classes w.r.t. these case weights is $p_{C_1} : p_{C_2} = (1.4 + 1.6) : (1.0 + 1.0 + 1.0) = 3 : 3 = 1 : 1$.

Obviously the sum of the case weights has changed after splitting the training set according to the fuzzy partition of attribute A. Since the entropy of a uniform probability distribution is maximal, the entropy $I(\hat{\chi}^N)$ after the split is certainly larger than the entropy $I(\chi^N)$ before the split. If we have $\forall k, 1 \leq k \leq 5 : \hat{\chi}^N_k = \chi^{N|\text{small}}_k + \chi^{N|\text{large}}_k$ (as it is implicitly assumed by the definition of information gain), we have that $\widehat{\text{Gain}}(\chi^N, A) = I(\hat{\chi}^N) - I(\chi^N|A)$ is definitely non-negative. Now it is easy to conclude:

$$\begin{array}{ll} \text{since} & I(\hat{\chi}^N) > I(\chi^N) \\ \text{and} & \widehat{\text{Gain}}(\chi^N, A) = I(\hat{\chi}^N) - I(\chi^N|A) > 0 \\ \Longrightarrow & \text{Gain}(\chi^N, A) = I(\chi^N) - I(\chi^N|A) < 0 \quad \text{is possible.} \end{array}$$

Indeed we have for the example considered above $\text{Gain}(\chi^N, A) = 0.917 - 0.971 = -0.054$.

As mentioned above, negative information gain can also result from the t-norm (e.g. $\top_{\min}$) that is used in the FDCT to compute the membership degrees of the examples in a node. An example for such a situation can easily be constructed in analogy to example 1.

A negative information gain, although it has no real meaning, can still yield a correct ranking of the candidate test attributes. But if information gain ratio is used, a negative value for the information gain can produce an inappropriate answer. To see this, let us consider a simple example: suppose we have two candidate attributes A and B with information gain $\text{Gain}(\chi^N, A) > \text{Gain}(\chi^N, B)$ and split information $\text{SplitI}(\chi^N, A) \gg \text{SplitI}(\chi^N, B)$.

1) In classical decision trees it is always $\text{Gain}(\chi^N, A) > \text{Gain}(\chi^N, B) \geq 0$, and then it may be that
$$0 \leq \frac{\text{Gain}(\chi^N, A)}{\text{SplitI}(\chi^N, A)} < \frac{\text{Gain}(\chi^N, B)}{\text{SplitI}(\chi^N, B)}.$$
This is desired, because it reduces the well-known bias of information gain towards many-valued attributes.
2) In the fuzzy domain, however, we can also have the situation $\text{Gain}(\chi^N, A) > 0 > \text{Gain}(\chi^N, B)$. In such a case, attribute A is always favored by the information gain ratio, because
$$\frac{\text{Gain}(\chi^N, A)}{\text{SplitI}(\chi^N, A)} > 0 > \frac{\text{Gain}(\chi^N, B)}{\text{SplitI}(\chi^N, B)},$$
independent of the relationship between $\text{SplitI}(\chi^N, A)$ and $\text{SplitI}(\chi^N, B)$, and this contradicts our intuition.

[1]In general, one has to combine the membership degree to the fuzzy example set and the membership degree to the fuzzy set of the attribute with a t-norm.

TABLE III
(FUZZY) CONTINGENCY TABLE FOR NODE N WITH ATTRIBUTE A AS THE TEST CANDIDATE

N (A)	C_1	C_2	*ASum*
a_1	$Z^{N\|a_1}_{C_1}$	$Z^{N\|a_1}_{C_2}$	$Z^{N\|a_1}$
a_2	$Z^{N\|a_2}_{C_1}$	$Z^{N\|a_2}_{C_2}$	$Z^{N\|a_2}$
CSum	$Z^N_{C_1}$	$Z^N_{C_2}$	Z^N

In this paper we try to eliminate the problems mentioned above by using the appropriate entropy to ensure a positive information gain.

D. Extended information measure

As mentioned above, negative information gain can occur in FDCT induction (e.g. in the version developed in [6]) if we use the entropy $I(\chi^N)$ computed from the membership degrees of the examples (interpreted as case weights) in the current node, in which a test attribute is to be chosen.

In this section we suggest a different way of computing the information measure in the fuzzy domain to make information gain ratio applicable as a selection measure. Using the entropy $I(\hat{\chi}^N)$ of the examples (derived from the case weights $\hat{\chi}^N$ as computed above), which implicitly includes the information of the test attribute, we can guarantee that the information measure is non-negative. We proceed as the following simple example demonstrates:

- Let $C = \{C_1, C_2\}$ be the set of classes.
- Let $E = \{e_1, \ldots, e_s\}$ be the reference set and χ^N be the fuzzy example set in node N.
- Let A be a candidate test attribute that has a fuzzy partition with two fuzzy sets $\{a_1, a_2\}$, i.e., with A as the test attribute in N we would have two branches, each connected to one fuzzy set.
- Let u^A_k be the value of attribute A in example e_k, $1 \leq k \leq s$.
- Let $\mu_{a_i}(u^A_k)$ be the membership degree of attribute value u^A_k to the fuzzy set a_i.
- Let $\chi^{N|a_i}_k = \top_1(\chi^N_k, \mu_{a_i}(u^A_k))$, $1 \leq k \leq s$, be the membership degree of the example e_k to the fuzzy example subset for the fuzzy set a_i, $i = 1, 2$.

To determine the best test attribute, we create a (fuzzy) contingency table (see Table III) for each candidate A in node N, from which we can compute the information measure for attribute A.

1) If A were the test attribute in N, the fuzzy example subsets in the two child nodes are $\chi^{N|a_i}$, $i = 1, 2$. Then it is $\chi_k^{N|a_i} = \top_1(\mu_{a_i}(u_k^A), \chi_k^N)$, $i = 1, 2$, $1 \le k \le s$.
2) $Z_{C_j}^{N|a_i}$ (here: $j \in \{0, 1\}, i \in \{0, 1\}$) is the counter for the examples that belong to fuzzy set a_i and class C_j. It can be computed as $Z_{C_j}^{N|a_i} = \sum_{k=1}^{s} \top_2(\chi_k^{N|a_i}, y_k^j)$ by interpreting the membership degrees as case weights.
3) $Z_{C_j}^N$, $j = 1, 2$, in row "CSum" is the counter for the examples which belong to class C_j. It is computed as: $Z_{C_j}^N = Z_{C_j}^{N|a_1} + Z_{C_j}^{N|a_2}$.
4) From row "CSum" we obtain the class frequency distribution and its entropy in N: $I(\hat{\chi}^N) = -\sum_{j=1}^{2} \left(\frac{Z_{C_j}^N}{Z^N}\right) \log_2 \left(\frac{Z_{C_j}^N}{Z^N}\right)$, where Z^N is the counter for the entire examples: $Z^N = Z_{C_1}^N + Z_{C_2}^N$.
5) Each row "a_i" represents a child node $N|a_i$. Line-by-line we can get the entropy of each fuzzy example subset for the fuzzy set a_i as $I(\chi^{N|a_i}) = -\sum_{j=1}^{2} \left(\frac{Z_{C_j}^{N|a_i}}{Z^{N|a_i}}\right) \log_2 \left(\frac{Z_{C_j}^{N|a_i}}{Z^{N|a_i}}\right)$, where $Z^{N|a_i}$, $i = 1, 2$, is the counter for the entire examples in child node $N|a_i$: $Z^{N|a_i} = Z_{C_1}^{N|a_i} + Z_{C_2}^{N|a_i}$
6) The weighted sum of entropies $I(\chi^N|A)$ of the subsets is then $I(\chi^N|A) = \sum_{i=1}^{2} \frac{Z^{N|a_i}}{Z^N} I(\chi^{N|a_i})$. We notice that $Z^N = Z^{N|a_1} + Z^{N|a_2} = Z_{C_1}^N + Z_{C_2}^N$.
7) The information gain of attribute A is $\widehat{\mathrm{Gain}}(\chi^N, A) = I(\hat{\chi}^N) - I(\chi^N|A)$. Since $\hat{\chi}_k^N = \sum_{i=1}^{2} \chi_k^{N|a_i}$, $1 \le k \le s$, $I(\hat{\chi}^N)$ is calculated from the same case weights as $I(\chi^N|A)$. Therefore $\widehat{\mathrm{Gain}}(\chi^N, A)$ is guaranteed to be non-negative.
8) The split information $\mathrm{SplitI}(\chi^N, A)$ of attribute A is computed from $Z^{N|a_i}$, the sum of the membership degrees of the examples to the fuzzy example subset for the fuzzy sets a_i. That is $\mathrm{SplitI}(\chi^N, A) = -\sum_{i=1}^{2} \left(\frac{Z^{N|a_i}}{Z^N}\right) log_2 \left(\frac{Z^{N|a_i}}{Z^N}\right)$
9) The information gain ratio of attribute A is $\mathrm{GainR}(\chi^N, A) = \widehat{\mathrm{Gain}}(\chi^N, A) / \mathrm{SplitI}(\chi^N, A)$. Since $\widehat{\mathrm{Gain}}(\chi^N, A)$ is non-negative, $\mathrm{GainR}(\chi^N, A)$ is, of course, non-negative too.

With the steps described above, we can easily estimate the information measure for the current candidates in node N and chose the best one as the test attribute.

Remark: information gain ratio is used in C4.5 [9] to select the test attribute in order to reduce the natural bias of information gain, i.e., the fact that it favors attributes with many values (which may lead to a model of low predictive power). In FDCT induction, fuzzy partitions are created for all attributes before the tree induction. To keep the tree simple, usually each partition possesses as few fuzzy sets as possible. Since the outgoing branches are labelled with fuzzy sets instead of crisp values, the problem mentioned above is mitigated, because continuous values are mapped to few fuzzy sets and thus the problem of many values is less severe. Therefore the effect of using information gain ratio in FDCT may not be so obvious as in classical decision trees.

E. Missing value handling

Real data often contain missing values. To handle such data we extend the learning algorithm, so that deleting examples with missing values from the training data set is not necessary anymore.

The first question to be answered is how to assign the examples with missing values of the test attribute to the outgoing branches of a tree node. In this paper a popular method from classical decision trees is adopted: an example e_k is distributed evenly to all children if the value u_k^i for test attribute A_i is unknown. That is

$$\mu_{a_p^i}(u_k^i) = \frac{1}{|D_i|} \qquad \text{if } u_k^i \text{ is unknown,} \tag{1}$$

where $|D_i|$ is the number of the fuzzy sets of A_i.

The information gain can be interpreted as "the information gained about the classes by ascertaining the value of the test attribute". A test of an example with a missing value for the test attribute, can obviously provide no information about the class membership of this example. Therefore the assessment of candidate attributes has to be modified accordingly, so that attributes with missing values are penalized.

Suppose we are given a reference set E having missing values for attribute A_i. Then the calculation of the information gain for candidate attribute A_i from the (fuzzy) contingency table can, as suggested in [9], be modified as following:

$$\begin{aligned}
&\widehat{\mathrm{Gain}}(\chi^N, A_i) \\
&\quad = \text{ frequency of examples with known } A_i \\
&\qquad\qquad \cdot (I(\hat{\chi}^N) - I(\chi^N|A_i)) \\
&\quad + \text{ frequency of examples with unknown } A_i \\
&\qquad\qquad \cdot 0 \\
&\quad = \alpha \cdot (I(\hat{\chi}^N) - I(\chi^N|A_i)), \text{ where } \alpha = \frac{Z^{N|A_i \text{ known}}}{Z^N}
\end{aligned}$$

Due to the factor α the real information gain is only dependent on those examples with known values for the test attribute.

The information gain ratio can be amended in a similar way:

$$\mathrm{GainR}(\chi^N, A_i) = \alpha \cdot \frac{I(\hat{\chi}^N) - I(\chi^N|A_i)}{\mathrm{SplitI}(\chi^N, A_i)} \tag{2}$$

Since the split information $\mathrm{SplitI}(\chi^N, A_i)$ is the entropy of the frequency distribution over the values of attribute A_i, the split information is increased artificially by evenly splitting the examples with missing values, and the information gain ratio is decreased accordingly (since $\mathrm{SplitI}(\chi^N, A_i)$ appears in the denominator). This effect is desired, because an attribute having missing values should be penalized. Since the increased split information already penalizes the measure, one may consider making the use of the factor α (see above) optional. That is, it is added only when a user explicitly requests it.

F. Fuzzy rule base

An important goal of this paper is to generate a comprehensible classification model, here a fuzzy rule base, which

TABLE IV

TEST DATA

data	*size*	*attributes*	*classes*	*missing value*
iris	150	4	3	no
glass	214	10 (incl. Id)	7	no
thyroid	215	5	3	no
wbc	699	10 (incl. Id)	2	yes
pima	768	8	2	no

can be generated from the fuzzy decision tree, that has been learned from data as described above.

The fuzzy rules are generated by transforming each path to a leaf of the tree into a rule. A simpler model or a model with better predictive power cannot be produced by such rewriting of the tree. To achieve this an optimization of the rule base is necessary. We optimize the rule base by rule pruning, where three heuristic strategies are used, which are adapted from [7]:

1) Pruning by information measure: the attribute having the smallest influence on the output should be deleted.
2) Pruning by redundancy: the linguistic term, which yields the minimal membership degree in a rule in the least number of cases, should be deleted.
3) Pruning by classification frequency: The rule, which yields the maximal fulfillment degree in the least number of cases, should be deleted.

Since the comprehensibility of a fuzzy system can be defined by the number of the rules, the number of attributes used in a rule and the number of fuzzy sets per attribute, the heuristics used in the strategies above are plausible. The pruning process can work automatically without any user interactions.

We do not discuss how the fuzzy rule base is used to classify new data, since it works in basically the same manner as standard fuzzy systems.

III. EXPERIMENTS

In this section, we report some results obtained from experiments run with the program FDCT, which was written by the first author of this paper, the well-known decision tree learner C4.5 (Release 8)[2] [9], a neural network training program [4], and NEFCLASS [8], which can generate a fuzzy rule based classifier by coupling neural networks with fuzzy systems. We compare the models generated by these programs w.r.t. precision, complexity, and the ability of dealing with missing values. For the tests we used five data sets from the UC Irvine Machine Learning Reposity [2]. Table IV shows general information about these data sets.

All experiments were run with 10-fold cross validation. C4.5 was run with the standard configuration. In NEFCLASS for each attribute a fuzzy partition with three fuzzy sets was created, which were evenly distributed over the attribute's domain. Fuzzy sets were also optimized during the rule pruning. The neural network program trained a multilayer perceptron (MLP) with one hidden layer containing 3 neurons for 1000 epochs.

[2]The learning result of C4.5 can be both a tree or a rule base. Here we used the generated rule base for the experiments.

TABLE V

10-FOLD CROSS VALIDATION

model		*iris*	*glass*	*thyroid*	*wbc*	*pima*
FDCT (1)	$\bar{\varepsilon}$	4.67%	34.29%	**3.33%**	2.79%	31.32%
	n	3	11	5	12	2
FDCT (2)	$\bar{\varepsilon}$	5.33%	31.90%	7.62%	2.64%	**18.82%**
	n	3	33	10	17	40
C4.5	$\bar{\varepsilon}$	4.01%	33.54%	7.03%	4.83%	23.3%
	n	4	14	7	8	8
NEFCLASS	$\bar{\varepsilon}$	**3.33%**	32.19%	11.60%	**2.35%**	25.89%
	n	3	14	6	21	14
NN	$\bar{\varepsilon}$	6.25%	**31.27%**	3.54%	5.05%	24.67%

Since we tried to generate comprehensible classification models, a trade-off between precision and complexity should be found. With this concern in mind, in FDCT a threshold of 0.05 for the information measure was chosen. That is, a test is created only if the chosen test attribute yields an information value higher than 0.05.

A. Precision and complexity

Table V shows the average error rate $\bar{\varepsilon}$, as well as the number of rules n of the resulting classifiers after pruning. The best error rate of the models is printed in bold face in the table.

In these experiments, FDCT was run with two different initial partitioning of the attributes – the automatic (labelled as *FDCT (1)*) and the individual partitioning (labelled as *FDCT (2)*) mentioned above. With the individual partitioning each attribute was partitioned with three fuzzy sets, which were evenly distributed over the attribute's domain, while with automatic partitioning the number of fuzzy sets was determined by the program.

If we consider only the precision of the models, it is very difficult to say which method is the best one, since each method produces the best result at least once. C4.5 never yields the worst error rate. FDCT (1) gives the highest precision (3.33%) for the *thyroid* data and at the same time a very small rule base (5 rules). For the *wbc* data NEFCLASS achieves the best performance of 2.35% with as many as 21 rules, while FDCT (1) provides performance only slightly worse (2.79%), for which it needs only about half the rules (12). The neural network fares worst for the *wbc*, which is probably due to the fact that an MLP with three hidden neurons (as used here) is comparable in power to about 3 rules. With so few rules no good performance can be expected for the *wbc* data.

For the *pima* data the worst classification rate is provided by FDCT with the automatic partitioning (however, with only 2 rules). The reason is that the partitioning algorithm created for only 2 of the 8 attributes two fuzzy sets and only one fuzzy set for each of the rest. Therefore the potential number of rules is only four, with which no learning algorithm can do much. In a comparison with the best result of FDCT (with individual partitioning, which required as many as 40 rules),

TABLE VI

LEARNING FROM DATA WITH MISSING VALUES

data		*5%*			*10%*		
		FDCT	*C4.5*	*Nefclass*	*FDCT*	*C4.5*	*Nefclass*
iris	$\bar{\varepsilon}$	**4.67%**	8.01%	5.33%	10.67%	12.00%	**6.67%**
	n	4	5	3	3	3	3
glass	$\bar{\varepsilon}$	39.52%	**34.61%**	37.19%	**29.04%**	40.25%	39.18%
	n	13	11	18	21	10	23
thyroid	$\bar{\varepsilon}$	**3.81%**	8.34%	33.92%	**8.57%**	10.24%	18.53%
	n	6	7	3	4	6	5
wbc	$\bar{\varepsilon}$	5.51%	**4.86%**	4.87%	7.68%	**5.28%**	5.58%
	n	23	11	32	20	12	39
pima	$\bar{\varepsilon}$	31.45%	26.71%	**22.66%**	35.00%	**27.23%**	27.59%
	n	2	8	21	2	6	25

we noticed that the attributes of the *pima* data have a relatively strong interrelationship. Therefore the data can be predicted better only by combining several attributes. A finer granularity, which was achieved by FDCT with the individual partitioning, enhanced the probability of a combination of attributes, and thus led to a better performance.

The same partitioning like in FDCT (2) was also used in NEFCLASS. For the *pima* data NEFCLASS provided a slightly lower precision, but with less than half the rules. We assume that the reason is that the fuzzy sets of NEFCLASS were trained during the learning and pruning phase, so they probably fit the data better. In contrast to this the fuzzy sets used in FDCT (2) were created once at the beginning and did not change anymore.

If we compare the two groups of results yielded by FDCT — taking not only the precision but also the complexity of the classifiers into account — we conclude that the learning process creates better classifiers if it works with automatic instead of individual partitioning. In particular, the number of rules of the first variant is often clearly less than that of the latter. Presumably the reason is that in the first variant the class information is taken into account, whereas it is neglected in the latter.

B. Tests on imperfect data

The experiments on the data with missing values, which were generated by randomly deleting values from each data set, demonstrate how well different learning methods can cope with imperfect data. In these tests FDCT was only run with automatic partitioning. The two sections of Table VI[3] show the results for data sets with 5% and 10% missing values, respectively. The best results are printed in bold face.

As expected, the performance of all methods decreased with the increased portion of missing values. FDCT provided for the *thyroid* data (both 5% and 10% missing values) the best result, as well as for the *iris* data (5%) and the *glass* data (10%). However, it is impossible to single out a method that is consistently superior to the others. The properties of the data seem to be more important than the portion of missing values.

[3]The neural network program does not appear in this table, since it cannot work with missing values.

In C4.5 the threshold values for tests of continuous attributes are determined dynamically and locally in the nodes; in NEFCLASS, although all attributes are partitioned with fuzzy sets before learning, the fuzzy sets are still optimized afterwards. In contrast to this the fuzzy sets used in FDCT are not changed anymore after creation. The lack of such dynamic fitting may be a disadvantage of the resulting fuzzy decision tree.

IV. CONCLUSIONS

In this paper we tried to extend classical decision trees by means of fuzzy methods in order to achieve the ability to model vagueness and to build comprehensible classifiers.

Although the learning principle of FDCT is the same as that of classical decision trees, it was nevertheless strongly influenced by fuzzy theory. In particular, in FDCT the information measure used for the test attribute selection, because of the properties of fuzzy logic, can become negative. We introduced amendments for two measures—information gain and information gain ratio—to ensure a correct ranking of the candidates. To deal with missing values, we also presented further modifications for these measures. To be able to better control the complexity of the tree, we suggested to use a threshold for the information measure.

To optimize the fuzzy rules we extract from a FDCT, we transferred three heuristic pruning strategies from NEFCLASS. Since the rules are expressed linguistically, the classifier is easy to interpret. In our experiments we observed that the approach proposed here often generates smaller and at the same time comparably good rule bases. Hence we conclude that we reached our goal of obtaining comprehensible classifiers.

Future work consists in trying other fuzzy partitioning techniques to enhance the quality of the initial partitioning.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Detlef Nauck for his help. This work was supported in part by the British Telecom.

REFERENCES

[1] M. Berthold und D.J. Hand. “Intelligent Data Analysis — An Introduction”. Springer, Berlin, Germany 1999

[2] C.L. Blake and C.J. Merz. “UCI Repository of machine learning databases”. http://www.ics.uci.edu/˜mlearn/MLRepository.html

[3] C. Borgelt and R. Kruse. “Graphical Models — Methods for Data Analysis and Mining”. J.Wiley & Sons, Chichester, England 2002

[4] C. Borgelt. “`mlp` — multilayer perceptron training” (computer software). http://fuzzy.cs.uni-magdeburg.de/˜borgelt/software.html#mlp

[5] X.P. Boyen and L. Wenkel. “Fuzzy Decision Tree Induction for Power System Security Assessment”. IFAC Symposium on control of power plants and power systems. Mexico, 1995

[6] C.Z. Janikow. “Fuzzy Decision Trees: Issues and Methods”. IEEE Transactions on Systems, Man and Cybernetics, 28(1):1–14. IEEE Press, Piscataway, NJ, USA 1998

[7] D. Nauck, U. Nauck and R. Kruse. “NEFCLASS for JAVA — New Learning Algorithmus”. Proc. 18th Int. Conf. North American Fuzzy Information Processing Society, 472–476. IEEE Press, Piscataway, NJ, USA 1999

[8] D. Nauck and U. Nauck. “Nefclass — neuro-fuzzy classification” (computer software). http://fuzzy.cs.uni-magdeburg.de/˜nauck/software.html

[9] J.R. Quinlan. “C4.5: Programs for Machine Learning”. Morgan Kaufman, San Mateo, CA, USA 1993.