Fuzzy Decision Trees: Issues and Methods

Cezary Z. Janikow[1]

Department of Mathematics and Computer Science
University of Missouri – St. Louis
janikow@radom.umsl.edu

Decision trees are one of the most popular choices for learning and reasoning from feature-based examples. They have undergone a number of alterations to deal with language and measurement uncertainties. In this paper, we present another modification, aimed at combining symbolic decision trees with approximate reasoning offered by fuzzy representation. The intent is to exploit complementary advantages of both: popularity in applications to learning from examples and high knowledge comprehensibility of decision trees, ability to deal with inexact and uncertain information of fuzzy representation. The merger utilizes existing methodologies in both areas to full advantage, but is by no means trivial. In particular, knowledge inferences must be newly defined for the fuzzy tree. We propose a number of alternatives, based on rule-based systems and fuzzy control. We also explore capabilities that the new framework provides. The resulting learning method is most suitable for stationary problems, with both numerical and symbolic features, when the goal is both high knowledge comprehensibility and gradually changing output. In this paper, we describe the methodology and provide simple illustrations.

1 Introduction

Today, in the mass storage era, knowledge acquisition represents a major knowledge engineering bottleneck [23]. Computer programs extracting knowledge from data successfully attempt to alleviate this problem. Among such, systems inducing symbolic decision trees for decision making, or classification, are very popular. The resulting knowledge, in the form of decision trees and inference procedures, has been praised for comprehensibility. This appeals to a wide range of users who are interested in domain understanding, classification capabilities, or the symbolic rules that may be extracted from the tree [32] and subsequently used in a rule-based decision system. This interest, in turn, has generated extensive research efforts resulting in a number of methodological and empirical advancements [6][11][20][30][31][32].

Decision trees were popularized by Quinlan (*e.g.*, [29]) with the ID3 program. Systems based on this approach work well in symbolic domains. Decision trees assign symbolic decisions to new samples. This makes them inapplicable in cases where a numerical decision is needed, or when the numerical decision improves subsequent processing [3].

1. This research has been partially supported by grant NSF IRI-9504334.

In recent years, neural networks have become equally popular due to relative ease of application and abilities to provide gradual responses. However, they generally lack similar levels of comprehensibility [34]. This might be a problem when users want to understand or justify the decisions. For such cases, researchers have attempted to combine some elements of symbolic and subsymbolic approaches [22][30][34]. A fuzzy approach is one of such extensions. Fuzzy sets provide bases for fuzzy representation [37]. Fuzzy sets and fuzzy logic allow the modelling of language-related uncertainties, while providing a symbolic framework for knowledge comprehensibility [38][39][40]. In fuzzy rule-based systems, the symbolic rules provide for ease of understanding and/or transfer of high-level knowledge, while the fuzzy sets, along with fuzzy logic and approximate reasoning methods, provide the ability to model fine knowledge details [40]. Accordingly, fuzzy representation is becoming increasingly popular in dealing with problems of uncertainty, noise, and inexact data [20]. It has been successfully applied to problems in many industrial areas ([15] details a number of such applications). Most research in applying this new representative framework to existing methodologies has concentrated only on emerging areas such as neural networks and genetic algorithms (*e.g.*, [7][15][17][20][34][36]). Our objective is to merge fuzzy representation, with its approximate reasoning capabilities, and symbolic decision trees while preserving advantages of both: uncertainty handling and gradual processing of the former with the comprehensibility, popularity, and ease of application of the latter. This will increase the representative power, and therefore applicability, of decision trees by amending them with an additional knowledge component based on fuzzy representation. These modifications, in addition to demonstrating that the ideas of symbolic AI and fuzzy systems can be merged, are hoped to provide the resulting fuzzy decision trees with better noise immunity and increasing applicability in uncertain/ inexact contexts. At the same time, they should proliferate the much praised comprehensibility by retaining the same symbolic tree structure as the main component of the resulting knowledge.

Decision trees are made of two major components: a procedure to build the symbolic tree, and an inference procedure for decision making. This paper describes the tree-building procedure for fuzzy trees. It also proposes a number of inferences.

In decision trees, the resulting tree can be pruned/restructured - which often leads to improved generalization [24][32] by incorporating additional bias [33]. We are currently investigating such techniques. Results, along comparative studies with systems such as C4.5 [32], will also be reported separately. In fuzzy representation, knowledge can be optimized at various levels: fuzzy set definitions, norms used, *etc*. Similar optimization for fuzzy decision trees is being investigated, and some initial results have been presented [12][13].

This paper concentrates on presenting the two major components for fuzzy decision trees. This is done is section 5, after briefly introducing domains (section 2), elements of fuzzy representation (section 3), and decision trees (section 4). Even though a few illustrations are used to augment that presentation, additional illustrative experiments are presented section 6.

2 Domain Types

Domain types usually encountered and processed in traditional machine learning and in fuzzy systems tend to be different. Because our objective is to merge both approaches, we need to explicitly address this issue.

In general, there exist two different kinds of domains for attributes: *discrete* and *continuous* (here *real-valued*, not continuous in time). Many algorithms require that data be described with discrete values. Replacing a continuous domain with a discrete one is not generally sufficient because most learning algorithms are only capable of processing small-cardinality domains. This requires some *partitioning*, or *covering* in general. For *nominal* domains, this requires a form of *clustering*. While many algorithms assume this is done apriori (*e.g.*, ID3 [29] and AQ [22])), some can do that while processing the data (*e.g.*, CART [1], and to a limited degree Dynamic-ID3 [6] and Assistant [16], and more recently C4.5 [32]). In this case, usually only values appearing in data are used to define the boundaries (such as those used in equality tests of Figure 3b).

The covering should be *complete*, that is, each domain element should belong to at least one of the covers (subsets). This covering should be *consistent* (*i.e.*, partitioning, as in traditional symbolic AI systems), or it can be *inconsistent* when a domain value can be found in more than one subset (as in fuzzy sets). Each of these derived subsets is labeled, and such labels become abstract features for the original domain.

In general, a discrete domain can be unordered, ordered partially, or ordered completely (as will every discretized domain). Any ordering can be advantageous as it can be used to define the notion of *similarity*. For example, consider the attribute *SkinColor*, with discrete values: *White*, *Tan*, *Black*. In this case one may say that *White* is “more similar” to *Tan* than to *Black*. Such orderings, along with similarity measures, are being used to advantage in many algorithms (*e.g.*, AQ [22]). They will also be used to advantage in fuzzy trees.

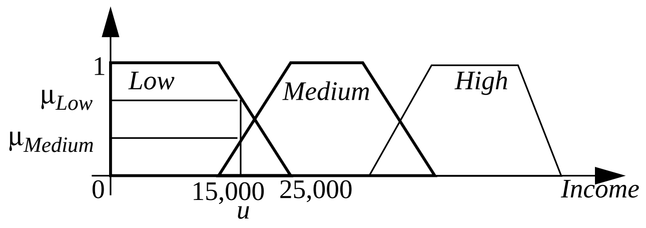
3 Fuzzy Sets, Fuzzy Logic, and Approximate Reasoning

Set and *element* are two primitive notions of set theory. In *classical* set theory, an element either belongs to a certain set or it does not. Therefore, a set can be defined by a two-valued characteristic function $U \to \{0, 1\}$, where U is the *universe of discourse*. However, in the real world, this is often unrealistic because of imprecise measurements, noise, vagueness, subjectivity, *etc*. Concepts such as *Tall* are inherently fuzzy. Any set of tall people would be subjective. Moreover, some people might be obviously tall, while others only slightly (in the view of the subjective observer). To deal with such ambiguities, symbolic systems have attempted to include additional numerical components [2][22]. For example, decision trees have been extended to accommodate probabilistic measures [30], and decision rules have been extended by flexible matching mechanisms [22]. Fuzzy sets provide another alternative.

In fuzzy set theory, a fuzzy subset of the universe of discourse U is described by a *membership* function $\mu_v(V): U \rightarrow [0, 1]$, which represents the degree to which $u \in U$ belongs to the set v. A *fuzzy linguistic variable* (such as V) is an attribute whose domain contains linguistic values (also called *fuzzy terms*), which are labels for the fuzzy subsets [39][40]. Stated differently, the meaning of a fuzzy term is defined by a fuzzy set, which itself is defined by its membership function. For example, consider the continuous (or high-cardinality discrete) attribute *Income*. It becomes a fuzzy variable when three linguistic terms, such as *Low*, *Medium*, and *High*, are used as domain values. The defining fuzzy sets usually overlap (to reflect "fuzziness" of the concepts) and should cover the whole universe of discourse (for completeness). This is illustrated in Figure 1, where some actual income u belongs to both the *Low* and the *Medium* subsets, with different degrees. Fuzzy sets are generally described with *convex* functions peaking at 1 (*normal* fuzzy sets). Trapezoidal (or triangular) sets are most widely used, in which case a normal set can be uniquely described by the four (three) corners - values of U. For example, *Low* in Figure 1 could be represented as <0,0,10,000,25,000>. Trapezoidal sets tend to be the most popular due to computational/storage efficiency [4][8][35].

Similar to the basic operations of *union*, *intersection*, and *complement* defined in classical set theory (which can all be defined with the characteristic function only), such operations are defined for fuzzy sets. However, due to the infinite number of membership values, infinitely many interpretations can be assigned to those operations. Functions used to interpret intersection are denoted as *T-norms*, and functions for union are denoted as *T-conorms*, or *S-norms*. Originally proposed by Zadeh *min* and *max* operators for intersection and union, respectively, define the most optimistic and the most pessimistic norms for the two cases. Among other norms, *product* for intersection and *bounded sum* (by 1) for union are the two most commonly used [4]. Even though norms are binary operations, they are commutative and associative.

Figure 1 Fuzzy subsets of the domain of *Income*, and memberships for an actual income *x*.



Similarly to *proposition* being the primitive notion of classical logic, *fuzzy proposition* is the basic primitive in fuzzy logic. Fuzzy representation refers to knowledge representation based on fuzzy sets and fuzzy logic. *Approximate reasoning* deals with reasoning in fuzzy logic.

An atomic fuzzy proposition (often called a *fuzzy restriction*) is of the form $[V \text{ is } v]$, where V is a fuzzy variable and v is a fuzzy term from its linguistic domain. The interpretation of the proposition is defined by the fuzzy set defining v, thus by μ_v. As classical propositions, fuzzy propositions can be combined by logical connectives into fuzzy statements, such as "$[V_1 \text{ is } v_1] \wedge [V_2 \text{ is } v_2]$". If both of the fuzzy variables share the same universe of discourse, the combination can be interpreted by intersecting the two fuzzy sets associated with the two linguistic values. Otherwise, the relationship expressed in the statement can be represented with a fuzzy relation (*i.e.*, a multi-dimensional fuzzy set). In this case, the same T-norms and S-norms can be used for $\wedge$ and $\vee$, respectively. For example, for the above conjunctive proposition, the relation would be defined by: $\mu_R(V_1, V_2) = T(\mu_{v_1}(V_1), \mu_{v_2}(V_2))$. The nice thing about fuzzy relations is that once the relation is defined, fuzzy sets satisfying the relation can be induced from fuzzy sets on other elements of the relation by mean of *composition* [4][8].

A piece of knowledge is often expressed as a rule, which is an implication from the (often conjunctive) antecedent to the (often atomic proposition) consequent. Knowledge can be expressed with a set of such rules. Similarly, a fuzzy rule of the form "if $[V_1 \text{ is } v_1] \wedge [V_2 \text{ is } v_2] \ldots$ then $[V_c \text{ is } v_c]$" expresses a piece of knowledge. The implication can be interpreted using the basic connectives and then represented by a fuzzy relation, or it may have its own direct interpretation. In fact, in approximate reasoning implication is usually interpreted with a T-norm. When the *min* norm is used, this corresponds to "clipping" the fuzzy set of the consequent. When *product* is used, this corresponds to "scaling" that fuzzy set, in both cases by the combined thruthness of the antecedent. These concepts are later illustrated in Figure 2.

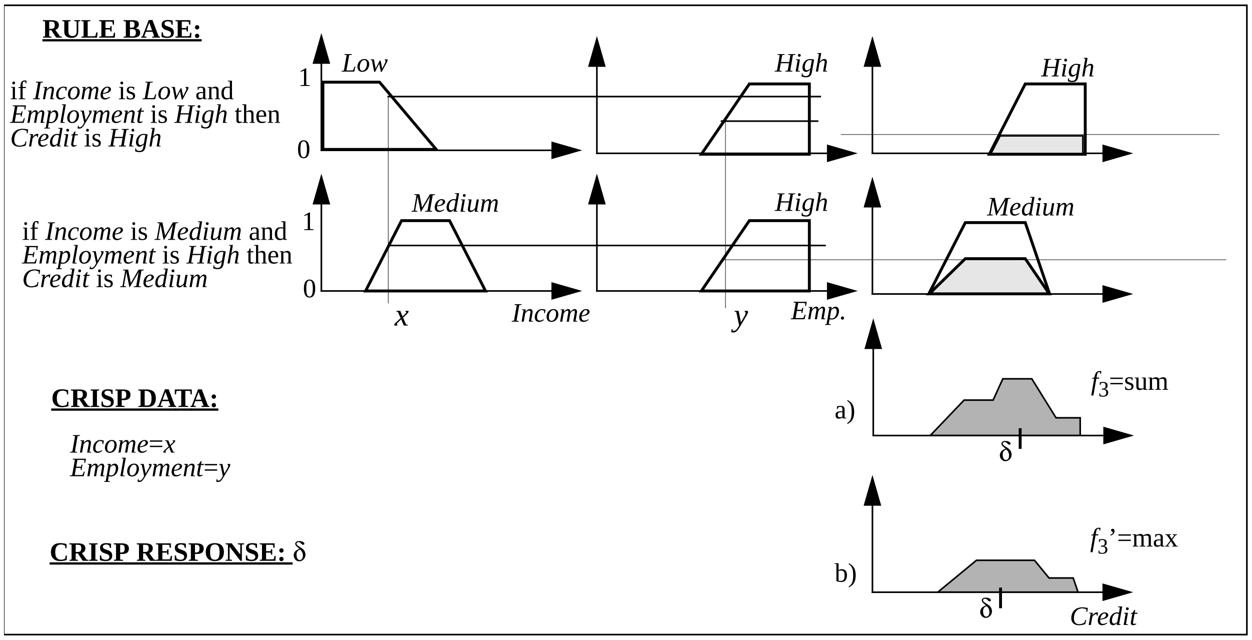
A fuzzy rule-based system is a set of such fuzzy rules along with special inference procedures. As was the case with a single rule, knowledge of the rules can be expressed by a fuzzy relation, with composition used for the inference. This *global-inference* approach is computationally expensive, and it is usually replaced with *local-inference*, where rules fire individually, and then the results are combined. In fact, under many common conditions, such as crisp inputs and T-norm implications, the two approaches yield equivalent results. Because we do not deal directly with fuzzy rules, we are not explicitly concerned with this equivalence in our subsequently defined local inferences. Interested readers are referred to [4] and [8] for an extensive discussion.

In the local-inference stage of a fuzzy rule-based system, current data (facts, measurements, or other observations) are used analogously to other rule-based systems. That is, the satisfaction level of each rule must be determined and a conflict resolution used. This is accomplished with the following mechanisms:

1. f_0 to determine how a data input for a fuzzy variable V_i, satisfies the fuzzy restriction $[V_i \text{ is } v^i_j]$.
2. f_1 to combine levels of satisfaction of fuzzy restrictions of the conjunctive antecedent (the resulting value is often called *degree of fulfillment* or *satisfaction* of the rule). Since f_1 will use some of the T-norms, which are associative and commutative, we may write f_1(“conjunction of atomic fuzzy propositions“, $data$) to imply multiple application of the appropriate T-norm operator.
3. f_2 to propagate satisfaction of the antecedent to the consequent (the result is often referred to as the degree of satisfaction of the consequent).
4. f_3 for the conflict resolution from multiple consequents.

$f_0(v, u)$ is determined from the membership $\mu_v(u)$ for crisp inputs u. For fuzzy inputs, the height of the intersection of the two corresponding fuzzy sets can be used [4]. f_1 and f_2 are defined with T-norms, f_3 is defined with an S-norm. These choices are illustrated in Figure 2, assuming a two-rule system using fuzzy linguistic variables *Income* (of Figure 1) and *Employment* (with similar fuzzy sets over its universe). As illustrated, the antecedent of the first rule is interpreted with *product*, and this level of satisfaction is aggregated with the consequent using *min* (the consequent’s fuzzy set is clipped). The second rule is evaluated with *min* and *product*, respectively (the consequent’s fuzzy set is scaled). Combinations of the two consequents with *sum* and *max aggregations* [26] are illustrated in a) and b), respectively. If some form of center of gravity is used for calculation of the crisp response, the results would depend on the choice of the operators used.

Figure 2 Illustration of the inference on fuzzy rules



In many practical applications, such as fuzzy control, the resulting fuzzy set (result of f_3) must be converted to a crisp response. This is important since we will expect crisp responses from our fuzzy tree. There is a number of possible methods in local inference. It may be as simple as taking the center of gravity of the fuzzy set associated with the consequent of the most satisfied rule, or it may combine information from multiple rules.

For computational reasons, this *defuzzification* process must also be efficient. One way to accomplish that is to replace the continuous universe for the decision variable (of the consequent) with a discrete one. Following this idea, the most popular defuzzification methods in local inference are *Center-of-Area/Gravity* and *Center-of-Sum*, which differ by the treatment of set overlaps, but the necessary computations are still overwhelming [4].

When either a crisp value, or one of the existing fuzzy sets (*i.e.*, to avoid representing new arbitrarily shaped functions) is desired for output, great computational savings can be accomplished by operating on crisp representations of the fuzzy sets of the consequent variable, which can all be precomputed. Then, all rules with the same consequent can be processed together to determine the combined degree of satisfaction β_p of the consequent's fuzzy set v_p. This information is subsequently combined, as in the *Fuzzy-Mean* inference [8]:

$$\delta = \frac{\sum_{p=1}^{n} (\beta_p \cdot \zeta_p)}{\sum_{p=1}^{n} \beta_p}$$

where ζ_p is the crisp representation of the fuzzy set v_p (*e.g.*, the center of gravity or the center of maxima) and n is the number of linguistic values of the consequent's fuzzy variable. Alternatively, all rules can be treated independently, in which case the summation takes place over all rules in the fuzzy database *(*for example, the *Height* method in [4]). From the interpretation point of view, this method differs from the previous one by accumulating information from rules with the same consequents. Thus, care must be taken about redundancy of the rule base, and in general theoretical fuzzy sets are violated (See Yager in [15]). Also, ζ_p will adequately represent v_p's fuzzy set only if the set is symmetric [4].

In methods such as the two above, one may also use additional weights α to emphasize some membership functions, as in the *Weighted-Fuzzy-Mean* method [8]:

$$\delta = \frac{\sum_{i=1}^{m} (\alpha_p \cdot \beta_i \cdot \zeta_p)}{\sum_{i=1}^{m} \alpha_p \cdot \beta_i}$$

where summation takes place over all m rules, p refers to the linguistic value found in the consequent of the i^{th} rule, and β_i denotes the degree of fulfillment of the rule's consequent.

When the weights reflect areas under the membership functions (relating to sizes of the fuzzy sets), the method resembles the standard Center-of-Gravity, except that it is computationally simpler.

For extensive discussion of both theoretical and practical defuzzification methods the reader is referred to [4][8][26]. Our own inferences (section 5.4) will be derived from the machine learning point of view, but many of them will in fact be equivalent to some of the existing computationally simple local inferences in approximate reasoning, with summation over the rule base.

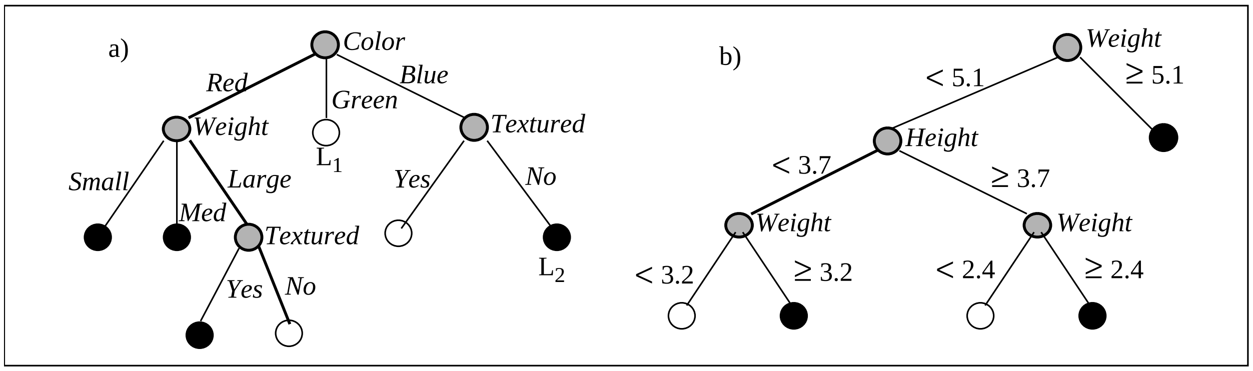
We must also refer to the interpolative nature of most of the inferences from a fuzzy rule base. For example, [*Income* is *Medium]*, or the crisp income value [*Income*=42,000], could be inferred from two satisfied rules having consequents [*Income* is *Low*] and [*Income* is *High*]. This is appropriate for this variable (with continuous universe, and thus with ordered fuzzy terms, see section 2). The interpretation of this inference is that if, given some context, one arrives at the two conflicting responses, it is safe to take the middle-ground approach. However, for nominal variables such as *WhoToMarry*, with domain {*Alex*, *Julia*, *Susan*}, this approach fails. Variables of this kind frequently appear in machine learning applications, and we need to accommodate them in fuzzy decision trees. Fortunately, some of the existing defuzzification methods in local inferencing are non-interpolative (for example, *Middle-of-Maxima* in [4]). Accordingly, some of our own inferences will also be non-interpolative.

AI methods must not only provide for knowledge representation and inference procedures, but for knowledge acquisition as well – knowledge engineering by working with human experts is a very tedious task [5][23]. Therefore, an important challenge for fuzzy systems is to generate the fuzzy rule base automatically. In machine learning, knowledge acquisition from examples is the most common practical approach. Similarly, fuzzy rules can be induced. To date, most research has concentrated on using neural networks or genetic algorithms, individually or in hybrid systems [7][9][11][17][36]. However, in machine learning, decision trees are the most popular method. That is why our objective is to use decision trees as the basis for our learning system.

4 Decision Trees

In *supervised learning*, a sample is represented by a set of features expressed with some descriptive language. We assume a conjunctive interpretations of the features representing a given sample. Samples with known classifications, which are used for training, are called *examples*. The objective is to induce decision procedures with *discriminative* (most cases), *descriptive*, or *taxonomic* bias [21] for classification of other samples. Following the *comprehensibility principle*, which calls upon the decision procedures to use language and mechanisms suitable for human interpretation and understanding [23], symbolic systems remain extremely important in many applications and environments. Among such, decision trees are one of the most popular.

Figure 3 Examples of a decision tree generated by ID3 (a) and CART (b).



ID3 (*e.g*, [29]) and CART [1] are the two most important discriminative learning algorithms working by recursive partitioning. Their basic ideas are the same: partition the sample space in a data-driven manner, and represent the partition as a tree. Examples of the resulting trees are presented in Figure 3, built from examples of two classes, illustrated as black and white. An important property of these algorithms is that they attempt to minimize the size of the tree at the same time as they optimize some quality measure [1][24][25]. Afterwards, they use the same inference mechanism. Features of a new sample are matched against the conditions of the tree. Hopefully, there will be exactly one leaf node whose conditions (on the path) will be satisfied. For example, a new sample with the following features: $[Weight = Small]$ $[Color = Green]$ $[Textured = Yes]$ matches the conditions leading to L_1 in Figure 3a. Therefore, this sample would be assigned the same classification as that of the training examples also satisfying the conditions (*i.e.*, those in the same node). For instance, it would be assigned the white classification here.

Nevertheless, the two trees are different. ID3 assumes discrete domains with small cardinalities. This is a great advantage as it increases comprehensibility of the induced knowledge (and results in the wide popularity of the algorithm), but may require an apriori partitioning. Fortunately, machine learning applications often employ hybrid techniques, which require a common symbolic language. Moreover, machine learning often works in naturally discrete domains, but even in other cases some background knowledge or other factors may suggest a particular covering. Some research has been done in the area of domain partitioning while constructing a symbolic decision tree. For example, Dynamic-ID3 [6] clusters multi-valued ordered domains, and Assistant [16] produces binary trees by clustering domain values (limited to domains of small cardinality). However, most research has concentrated on apriori partitioning techniques [18], and on modifying the inference procedures to deal with incomplete (missing nodes) and inconsistent (multiple matches, or a match to a leaf node containing training examples of non-unique classes) trees [2][20][24][25][30][31][32]. These problems might result from noise, missing features in descriptions of examples, insufficient set of training examples, improper partitioning, language with insufficient descriptive power or simply with an insufficient set of features. The latter problem has been addressed in some learning systems (*constructive induction*). However, decision tree algorithms can select the best features from a given set, they generally cannot build new features needed for a particular task [32] (even though some attempts have been made, such as [27]). Another important feature of ID3 trees is that each attribute can provide at most one condition on a given path. This also contributes to comprehensibility of the resulting knowledge.

The CART algorithm does not require prior partitioning. The conditions in the tree are based on thresholds (for continuous domains), which are dynamically computed (a feature also incorporated with C4.5 [32]). Because of that, the conditions on a path can use a given attribute a number of times (with different thresholds), and the thresholds used on different paths are very likely to differ. This is illustrated in Figure 3b. Moreover, the number of possible thresholds equals the number of training examples. These ideas often increase the quality of the tree (the induced partition of the sample space is not as restricted as it is in ID3), but they reduce its comprehensibility. Also, CART is capable of inducing new features, restricted to linear combinations of the existing features [1].

Among important recent developments one should mention Fuzzy-CART [9] and UR-ID3, for uncertain reasoning in ID3 [19]. The first is a method which uses the CART algorithm to build a tree. However, the tree is not the end result. Instead, it is only used to propose fuzzy sets (*i.e.*, a covering scheme) of the continuous domains (using the generated thresholds). Afterwards, another algorithm, a layered network, is used to learn fuzzy rules. This improves the initially CART-generated knowledge, and produces more comprehensible fuzzy rules. UR-ID3, on the other hand, starts by building a strict decision tree, and subsequently fuzzifies the conditions of the tree.

Our objective is high comprehensibility as well, accomplished with the help of the ID3 algorithm. We base our methodology on that algorithm, and modify it to be better suited to process various domains. Therefore, we make the same assumption of having a predefined partitioning/ covering. This partitioning can eventually be optimized. This work has been initiated [13]. To deal with cases where no such partitioning is available, we plan to ultimately follow Fuzzy-CART[9]. That is, we plan to have the CART algorithm provide the initial covering, which will be further optimized.

We now outline the ID3 partitioning algorithm, which will be modified in section 5.3. For more details, see [29][32]. The root of the decision tree contains all training examples. It represents the whole description space since no restrictions are imposed yet. Each node is recursively split by partitioning its examples. A node becomes a leaf when either its samples come from a unique class or when all attributes are used on the path. When it is decided to further split the node, one of the remaining attributes (*i.e.*, not appearing on the current path) is selected. Domain values of that attribute are used to generate conditions leading to child nodes. The examples present in the node being split are partitioned into child nodes according to their matches to those conditions. One of the most popular attribute selection mechanisms is one that maximizes information gain [25]. This mechanism, outlined below, is computationally simple as it assumes independence of attributes.

1. Compute the information content at node *N*, given by $I^N = -\sum_{i=1}^{|C|} (p_i \cdot \log p_i)$, where C is the set of decisions, and p_i is the probability that a training example in the node represents class i.

2. For each attribute a_i not appearing on the path to *N* and for each of its domain values a_{ij}, compute the information content $I^{N|a_{ij}}$ in a child node restricted by the additional condition $a_i = a_{ij}$.

3. Select the attribute a_i maximizing the information gain $I^N - \sum_j^{|D_i|} (w_j \cdot I^{N|a_{ij}})$, where w_j is the relative weight of examples at the child node following the a_{ij} condition to all examples in *N*, and D_i is the symbolic domain of the attribute.

4. Split the node *N* using the selected attribute.

Many modifications and upgrades of the basic algorithm have been proposed and studied. The most important are other attribute selection criteria [25], gain ratio accommodating domain sizes [32], other stopping criteria including statistical tests of attribute relevance [32], and tree-generalization postprocessing [24][32].

5 Fuzzy Decision Trees

5.1 Fuzzy Sample Representation

Our fuzzy decision tree differs from traditional decision trees in two respects: it uses splitting criteria based on fuzzy restrictions, and its inference procedures are different. Fuzzy sets defining the fuzzy terms used for building the tree are imposed on the algorithm. However, we are currently investigating extensions of the algorithm for generating such fuzzy terms, along with the defining fuzzy sets, either off-line (such as [9]) or on-line (such as [1][6]).

For the sake of presentation clarity, we assume crisp data form[2]. For instance, for attributes such as *Income,* features describing an example might be such as *Income*=$23,500. Examples are also augmented with “confidence” weights.

As an illustration, consider the same previously illustrated fuzzy variables *Income* and *Employment*, and the fuzzy decision *Credit*. That is, assume that applicants applying for credit indicated only last year’s income and current employment (a very simplistic scenario). Assume that each applicant indicated the exact income amount (more informative than just income brackets) and the number of hours worked (which is again more informative than the information whether they worked at all). Each application was either rejected or accepted (a binary decision). Or, alternatively, each application could have been given a score. For example, an applicant whose reported income was $52,000, who was working 30 hours a week, and who was given credit with some hesitation, could become the following training example [*Inc*=52,000][*Emp*=30] $\Rightarrow$ [*Credit*=0.7]:-weight=1. Our fuzzy decision tree can also handle fuzzy examples such as [*Inc*=52,000][*Employment*=*High*] $\Rightarrow$ [*Credit*=*High*], but for clarity we will not discuss such natural extensions here.

5.2 Other Assumptions and Notation

For illustration, we use only trapezoidal fuzzy sets. Currently, the tree is not pruned. Each internal node has a branch for each linguistic value of the split variable, except when no training examples satisfy the fuzzy restriction. To deal with the latter, we assume that during the inference, an unknown value in a node occurs either when the needed feature is missing from a sample’s description, or the node has no branch for it. Before we define the fuzzy-build and fuzzy-inference procedures, let us introduce subsequently used notation.

1. The set of fuzzy variables is denoted by $V = \{V_1, V_2 \dots V_n\}$.
2. For each variable $V_i \in V$

2. To be consistent with many machine learning problems, we allow nominal attributes as well. Extensions to linguistic data are straightforward.

- crisp example data is $u^i \in U_i$
- D_i denotes the set of fuzzy terms
- v_p^i denotes the fuzzy term p for the variable V_i (*e.g.*, v_{Low}^{Income}, as necessary to stress the variable or with anonymous values – otherwise p alone may be used)

3. The set of fuzzy terms for the decision variable is denoted by D_c.

4. The set of training examples is $E = \{e_j | e_j = (u_j^1, \ldots u_j^n, y_j)\}$, where y_j is the crisp classification (see the example at the end of section 5.1). Confidence weights of the training examples are denoted by $W = \{w_j\}$, where w_j is the weight for $e_j \in E$.

5. For each node N of the fuzzy decision tree

- F^N denotes the set of fuzzy restrictions on the path leading to N, *e.g.*, $F^{L_2} = \{[Color \text{ is } Blue], [Textured \text{ is } No]\}$ in Figure 3
- V^N is the set of attributes appearing on the path leading to N: $V^N = \{V_i | \exists p([V_i \text{ is } v_p^i] \in F^N)\}$
- $\chi^N = \{x_j^N\}$ is the set of memberships in N for all the training examples[3]
- $N|v_p^i$ denotes the particular child of node N created by using V_i to split N and following the edge $v_p^i \in D_i$. For example, the node L_1 in Figure 3 is denoted by $Root|v_{Green}^{Color}$
- $S_{V_i}^N$ denotes the set of N's children when $V_i \in (V - V^N)$ is used for the split. Note that $S_{V_i}^N = \{(N|v_p^i) \mid v_p^i \in D_i^N\}$, $D_i^N = \{v_p^i \in D_i | \exists(e_j \in E)\ (x_j^N > 0 \wedge \mu_{v_p^i}(u_j^i) > 0)\}$; in other words, there are no nodes containing no training examples, and thus some linguistic terms may not be used to create subtrees
- P_k^N denotes the example count for decision $v_k^c \in D_C$ in node N. It is important to note that unless the sets are such that the sum of all memberships for any u is 1, $P_k^N \neq \sum_{v_p^i \in D_i} P_k^{N|v_p^i}$; t hat is, the membership sum from all children of N can differ from that of N; this is due to fuzzy sets; the total membership can either increase or decrease while building the tree
- P^N and I^N denote the total example count and information measure for node N
- $G_i^N = I^N - I^{S_{V_i}^N}$ denotes the information gain when using $V_i$in N ($I^{S_{V_i}^N}$ is the weighted information content – see Section 4)

6. α denotes the area, ς denotes the centroid of a fuzzy set.

3. Retaining one-to-one correspondence to examples of E, this set implicitly defines training examples falling into N. This time-space trade-off is used in the implementation [10].

7. To deal with missing attribute values, we borrow the existing decision tree methodology. It has been proposed [31] that the best approach is to evenly split an example into all children if the needed feature value is not available, and then to reduce the attribute's utilization by the percentage of examples with unknown value:
   - denote $P^{N| (u^i \text{ unknown})}$ as the total count of examples in the node N with unknown values for V_i
   - define $f_0(e_j, [V_i \text{ is } v_p^i]) = \begin{cases} \frac{1}{|D_i|} & \text{if } u_j^i \text{ unknown} \\ \mu_{v_p^i}(u_j^i) & \text{otherwise} \end{cases}$

5.3 Procedure to Build A Fuzzy Decision Tree

In ID3, a training example's membership in a partitioned set (and thus in a node) is binary. However, in our case, the sets are fuzzy and, therefore, ideas of fuzzy sets and fuzzy logic must be employed. Except for that, we follow some of the best strategies available for symbolic decision trees.

To modify the way in which the number of examples falling to a node is calculated, we adapt norms used in fuzzy logic to deal with conjunctions of fuzzy propositions. Consider the case of visiting node N during tree expansion. Example e_j has its membership x_j^N in the node calculated on the basis of the restrictions F^N along the path from the root to N. Matches to individual restrictions in F^N are evaluated with f_0 above and then combined via f_1. For efficiency, this evaluation can be implemented incrementally due to properties of T-norms.

1. Start with all the examples E, having the original weights W, in the root node. Therefore, $\chi^{Root} = W$ (in other words, all training examples are used with their initial weights).
2. At any node N still to be expanded, compute the example counts (x_j^N is the membership of example e_j in N, that is, the membership in the multi-dimensional fuzzy set defined by the fuzzy sets of the restrictions found in F^N, computed incrementally with f_0 and f_1)

   $$P_k^N = \sum_{j=1}^{|E|} f_2(x_j^N, \mu_{v_k^c}(y_j)),\ P^N = \sum_{k=1}^{|D_c|} P_k^N$$

3. Compute the standard information content: $I^N = -\sum_{k=1}^{|D_c|} \left(\frac{P_k^N}{P^N} \cdot \log \frac{P_k^N}{P^N} \right)$.
4. At each node, we search the set of remaining attributes from $V - V^N$ to split the node (as in ID3, we allow only one restriction per attribute on any path):
   - calculate $I^{S_{V_i}^N}$, the weighted information content for V_i, adjusted for missing features. This is accomplished by weighting the information content by the percent-

age of examples with known values for V_i. The reciprocal of the total memberships in all children of N is a factored out term from the last sum

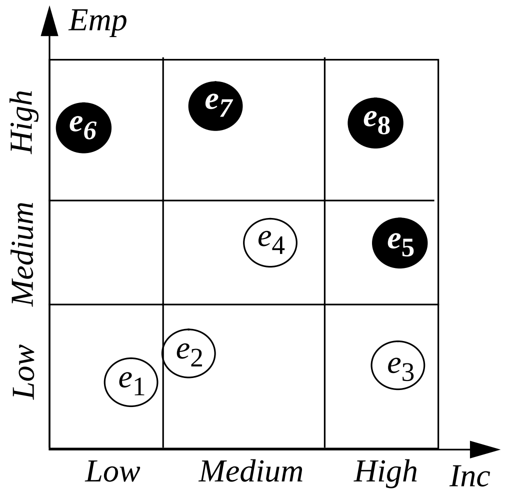
$$I^{S^N_{V_i}} = \frac{P^N - P^{N|(u^i \text{ unknown})}}{P^N} \cdot \frac{1}{\sum_{v^i_p \in D_i} P^{N|v^i_p}} \cdot \sum_{v^i_p \in D_i} (P^{N|v^i_p} \cdot I^{N|v^i_p})$$

- select attribute V_i such that the information gain G^N_i is maximal (might be adjusted for domain size [29]); the expansion stops if the remaining examples with $x^N_j > 0$ have a unique classification or when $V^N = V$; other criteria may be added

5. Split N into $|D_i|$ subnodes. Child $N|v^i_p$ gets examples defined by $\chi^{N|v^i_p}$. The new memberships are computed using the fuzzy restrictions leading to $N|v^i_p$ in the following way: $x^{N|v^i_p}_j = f_1(f_0(e_j, v^i_p), x^N_j)$.
However, if a child node contains no training examples, it cannot contribute to the inference (this is a property of the inferences). Therefore, all such children are removed (nodes are folded). Moreover, when some form of tree generalization is implemented in the future [32], additional nodes may have to be removed. This is why some internal nodes may end up with missing children.

Table 1 Numeric data for the illustration.

E	*Inc*	*Emp*	*Credit*	*W*
e_1	0.20	0.15	0.0	1.0
e_2	0.35	0.25	0.0	1.0
e_3	0.90	0.20	0.0	1.0
e_4	0.60	0.50	0.0	1.0
$\boldsymbol{e_5}$	0.90	0.50	1.0	1.0
$\boldsymbol{e_6}$	0.10	0.85	1.0	1.0
$\boldsymbol{e_7}$	0.40	0.90	1.0	1.0
$\boldsymbol{e_8}$	0.85	0.85	1.0	1.0



The above tree-building procedure is the same as that of ID3. The only difference is based on the fact that a training example can be found in a node to any degree. Given that the node memberships can be computed incrementally [10], computational complexities of the two algorithms are the same (up to constants).

Let us illustrate the tree building mechanism. Consider the two fuzzy variables *Income* (denoted *Inc* for abbreviation) and *Employment* (*Emp*), both with fuzzy sets such as those of Figure 1, the binary decision *Credit*, and the eight examples as in Table 1. As given, the first four examples belong to the *No* class (thus 0.0[4], illustrated in white). The other examples belong to the *Yes* class (1.0), illustrated in black. All confidence weights are equal. At the root (denoted *R*)

$$\chi^R = \{\, (x_1^R = 1.0)\,,\, \dots\, (x_8^R = 1.0)\,\}$$

$$P_{No}^R = 4.0,\, P_{Yes}^R = 4.0$$

$$I^R = 1.0$$

We continue expansion since $(V^R = \varnothing) \neq V$ and no complete class separation between black and white has been accomplished (we disregard other stopping criteria). We must select from $V - V^R = \{Emp, Inc\}$ the attribute which maximizes the information gain. Following section 5.3, we get

$$I^{S_{Inc}^R} = 0.9871,\, G_{Inc}^R = 0.0129$$

$$I^{S_{Emp}^R} = 0.3630,\, G_{Emp}^R = 0.6370$$

This means that *Emp* is a better attribute to split the root node. This can be easily confirmed by inspecting the data visualization on the right of Table 1, where horizontal splits provide better separations. Accordingly, the root R gets expanded with the following three children $R|\, v_{Low}^{Emp}$, $R|\, v_{Medium}^{Emp}$, and $R|\, v_{High}^{Emp}$. This expansion is illustrated in Figure 4, along with the intermediate computations. Some indices are omitted, f_1=*min* is used, and shade levels indicate the classification of the node's examples on the U_c scale.

The computations indicate that no further expansions of the left and the right nodes are needed: in both cases $I^N = 0$. This means that no further improvements can be accomplished since each of these leaves contains examples of a unique class (different ones here). An important aspect of fuzzy decision trees can be observed here. The example e_2, present with the initial memberships 1.0 in R, appears in two child nodes: in $R|\, v_{Low}^{Emp}$ with 0.5 and in $R|\, v_{Medium}^{Emp}$ with 0.33. This is obviously due to overlapping fuzzy sets for *Emp*.

The middle child must be split. Since $V - V^{R|\, v_{Medium}^{Emp}} = \{Inc\}$, there is no need to select the split attribute. Using *Inc*, we get the expansion illustrated in Figure 5.

At this moment, we have exhausted all the available attributes of V, so nothing more can be done (in practice, decision tree expansion usually stops before this condition triggers in order to

4. This can be any value from U_C, as illustrated in Section 5.1 and 6.2.

prevent over-specialization [32]). In the resulting tree only one of the five leaves does not contain examples of a unique class.

Figure 4 Partial fuzzy tree after expanding the root *R* (set elements with zero membership are omitted, and so are indexes which are obvious from the context).

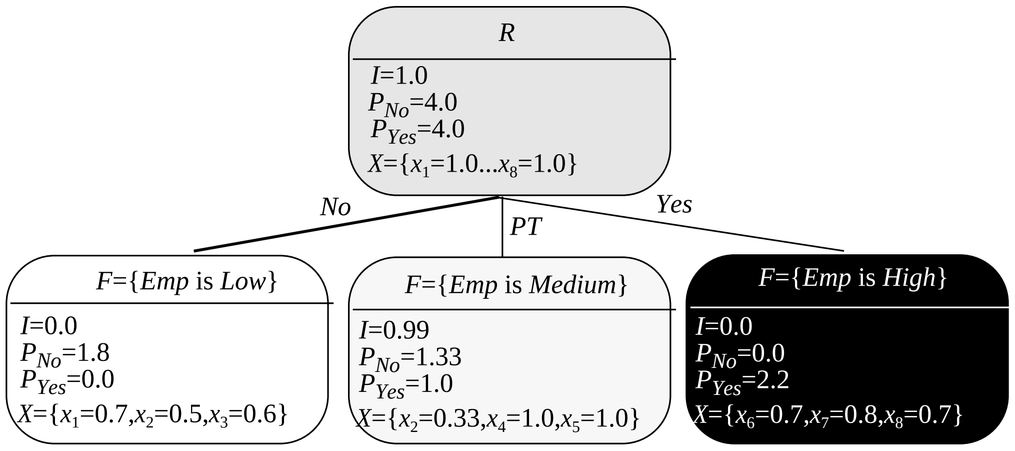
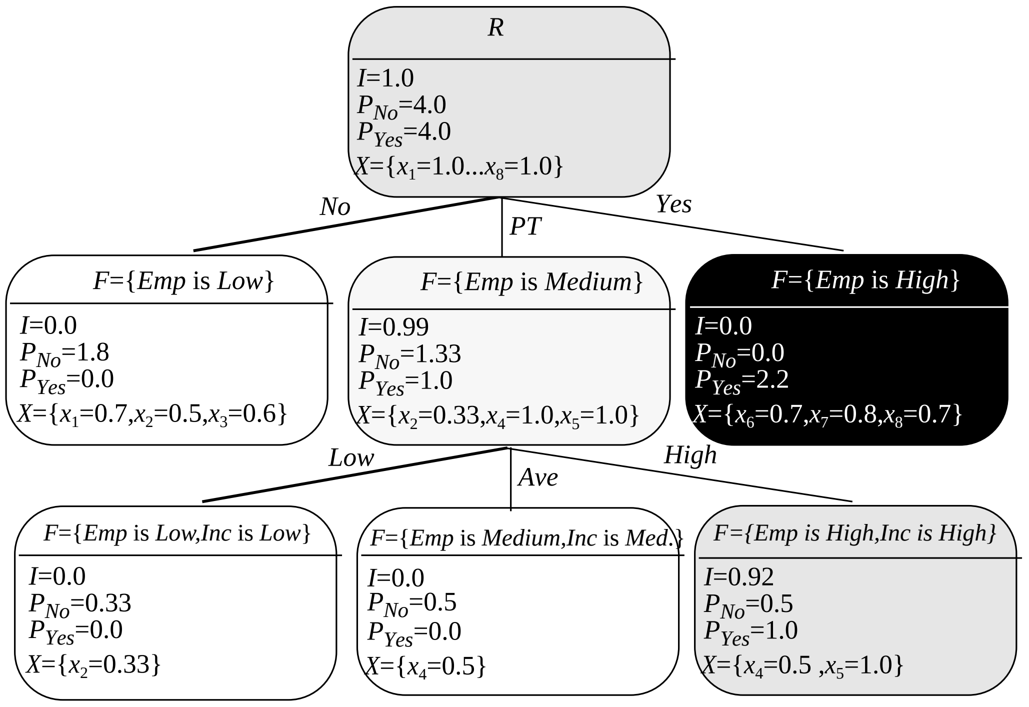


Figure 5 Complete fuzzy tree after expanding the $R|\, v^{Emp}_{Medium}$ child.



This tree (its structure, along with data in the leaves) and the fuzzy sets used for its creation combine to represent the acquired knowledge. Even though we need an inference procedure to interpret the knowledge, it is easy to see that unemployed people are denied credit, while fully employed people are given credit. For part-timers, only some of those with high income get credit. This natural interpretation illustrates knowledge comprehensibility of the tree. In the next section, we describe a number of inferences, that is methods for using this knowledge to assign such classifications to new samples.

5.4 Inference for Decision Assignment

Our fuzzy decision tree can be used with the same inference as that of ID3 (section 4). However, by doing so we would lose two pieces of information: the fuzzy sets are an important part of the knowledge the fuzzy tree represents, and the numerical features, which can be used to describe a sample, also carry more information than the abstract linguistic features. Therefore, we must define new inference procedures to utilize this information.

A strict decision tree can be converted to a set of rules [32]. One may think of each leaf as generating one rule – the conditions leading to the leaf generate the conjunctive antecedent, and the classification of the examples of the leaf generates the consequent. In this case, we get a consistent set of rules only when examples of every leaf have a unique classification, which may happen only when a sufficient set of features is used and the training set is consistent. Otherwise, we get a rule with a disjunctive consequent (which rule may be interpreted as a set of inconsistent rules with simple consequents). We get a complete set of rules only if no children are missing. Incompleteness can be treated in a number of ways – in general, an approximate match is sought. Inconsistencies can be treated similarly (see section 4). Because in fuzzy representation a value can have a nonzero membership in more than one fuzzy set, the incompleteness problem diminishes in fuzzy trees, but the inconsistency problem dramatically increases. To deal with this, we will define inferences following some of the approximate reasoning methods used in fuzzy control. We will also provide additional mechanisms to guarantee decisions in those cases where the classification fails otherwise.

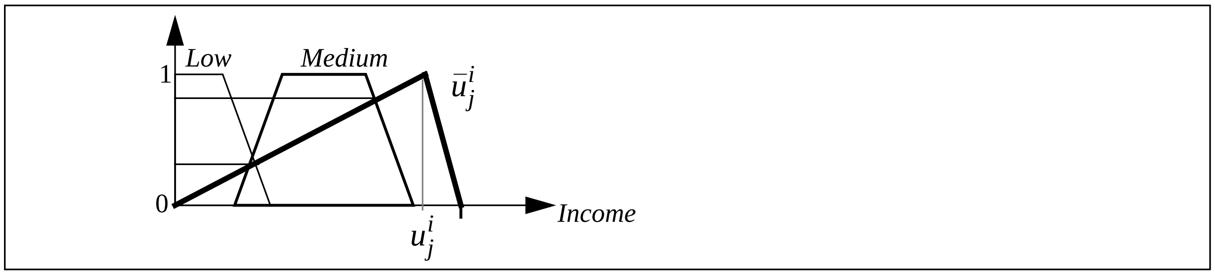
To define the decision procedure, we must define f_0, f_1, f_2, f_3 for dealing with samples presented for classification. These operators may differ from those used for tree-building – let us denote them g_0, g_1, g_2, g_3. First, we define g_0. It does not operate independently on the fuzzy restrictions associated with outgoing branches. Instead, it requires simultaneous information about the fuzzy restrictions. As previously stated, we also assume that all sample features are crisp values. There are three cases:

1. If the sample has a nonzero membership in at least one of the fuzzy sets associated with the restrictions out of a node, then the membership values are explicitly used: $g_0(e_j, [V_i \text{ is } v_p^i]) = \mu_{v_p^i}(u_j^i)$.
2. If the sample has an unknown value for the attribute used in the node, or when the value is known but memberships in all sets associated with the existing branches are zero (possibly related to $D_i^N \subset D_i$), and the variable in question has a nominal domain, then the sample is evenly split into all children: $g_0(e_j, [V_i \text{ is } v_p^i]) = 1/|D_i^N|$.

3. If the sample has the necessary value u_j^i, but all memberships are zero, and the variable tested in the node has a completely ordered set of terms, then following the idea of similarity [22] fuzzify the crisp value and compute matches between the resulting fuzzy set and the existing sets: $g_0(e_j, [V_i \text{ is } v_p^i]) = max_{z \in U_i}(\mu_{v_p^i}(z) | \mu_{v_p^i}(z) = \mu_{\bar{u}_j^i}(z))$ where $\bar{u}_j^i$ is defined with the triangular fuzzy set spanning over U_i and centering at u^i. This is illustrated in Figure 6.

Let us observe that since we need to present the methodology for the decision tree and not for the fuzzy rules that can be extracted from the tree, our inferences will be defined for the tree and not for a fuzzy rule base. To decide the classification assigned to a sample, we have to find leaves whose restrictions are satisfied by the sample, and combine their decisions into a single crisp response. Such decisions are very likely to have conflicts, found both in a single leaf (non-unique classifications of its examples) and across different leaves (different satisfied leaves have different examples, possibly with conflicting classifications).

Figure 6 Fuzzification of a crisp value in a node with all zero memberships, and the resulting matches.



Let us define L to be the set of leaves of the fuzzy tree. Let $l \in L$ be a leaf. Each leaf is restricted by F^l; all the restrictions must be satisfied to reach the leaf. A sample e_j must satisfy each restriction $[V_i \text{ is } v_p^i] \in F^l$ (using g_0), and all results must be combined with g_1 to determine to what degree the combined restrictions are satisfied. g_2 then propagates this satisfaction to determine the level of satisfaction of that leaf. We could use any T-norm, but in the implementation we will use the two most often used: *product* and *min*.

A given leaf by itself may contain inconsistent information if it contains examples of different classes. Different satisfied leaves must also have their decisions combined. Because of this two-level disjunctive representation, the choice of g_3 will be distributed into these two levels.

We first propose a general form for computationally efficient defuzzification, which is based on the Weighted-Fuzzy-Mean method mentioned in Section 3 (also known as simplified Max-Gravity [26]).

$$\delta_n = \frac{\sum_{i=1}^{|L|} (g_2(g_1(F^l, e_j)) \cdot S_l^n)}{\sum_{i=1}^{|L|} (g_2(g_1(F^l, e_j)) \cdot s_l^n)}, n \in \{1, \ldots 4\}$$

In other words, when e_j is presented for classification, first all satisfied paths are determined, and subsequently these levels of satisfaction are propagated to the corresponding leaves. *S* and *s* reflect the information contained in a leaf (*i.e.*, information expressed by χ^l) in such a way that if all examples with nonzero memberships in the leaf have single classifications (all have nonzero memberships in the same fuzzy set for the decision variable, and zero memberships in all other sets), then *S*/*s* yields the center of gravity of the fuzzy set associated with that classification. For efficient inferences, *S* and *s* can be precomputed for all leaves. However, in other cases different choices for *S* and *s* reflect various details taken into account. We present a number of inferences, to be evaluated empirically in future studies, some of which have better intuitive explanations than others – they all result from trying to combine elements of rule-based reasoning with approximate reasoning in fuzzy control.

We now propose four inferences based on the above. The first two inferences disregard all but the information related to the most common classification of individual leaves. The next two inferences take inconsistencies inside of individual leaves into account.

1. A leaf may assume the center of gravity of the fuzzy set associated with its most common classification (measured by the sum of memberships of the training examples in that leaf). The corresponding fuzzy set (and term) is denoted κ.

   $$S_l^1 = \alpha_\kappa \cdot \zeta_\kappa \qquad s_l^1 = \alpha_\kappa$$

   In this case $S/s = \zeta_\kappa$ even if the examples of the leaf have nonzero memberships in more than one fuzzy set of the decision variable. When the sample has a nonzero membership in only one of the leaves, or when only one leaf is taken into account, then $\delta_1 = \zeta_\kappa$. In the same case, but when all leaves are considered, the resulting inference is equivalent to Weighted-Fuzzy-Mean (section 3).

2. When the first case is used and the inference combines multiple leaves, no information reflecting the number of training examples in a leaf is taken into account. Alternatively, our inference may favor information from leaves having more representatives in the training set, hopefully helping with reducing the influence of noisy/atypical training examples (one may argue both for and against this inference).

   $$S_l^2 = P_\kappa^l \cdot \alpha_\kappa \cdot \zeta_\kappa = P_\kappa^l \cdot S_l^1 \qquad s_l^2 = P_\kappa^l \cdot \alpha_\kappa = P_\kappa^l \cdot s_l^1$$

If the sample satisfies a single leaf only, then $\delta_2 = \delta_1$.

3. An individual leaf may assume the sum-center of gravity of all of its fuzzy sets weighted by information provided in *X* (*i.e.*, by its example counts).

$$S_l^3 = \sum_{k=1}^{|D_c|} P_k^l \cdot \alpha_k \cdot \zeta_k \qquad s_l^3 = \sum_{k=1}^{|D_c|} P_k^l \cdot \alpha_k$$

Note that $\delta_3 = \delta_2$ when each leaf contains only training examples having non-zero memberships in unique fuzzy sets of the decision variable.

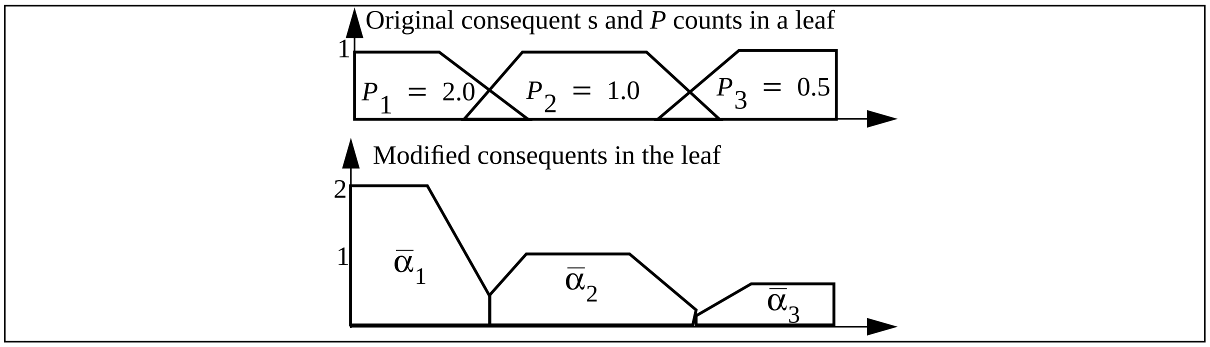
4. The same as case 3, except that max-center of gravity (see [26] and Figure 2) is used in an efficient computation. Here, $\bar{\alpha}_k$ represents the area of a fuzzy set restricted to the sub-universe in which the set dominates the others when scaled by P_k^l, and $\bar{\zeta}$ represents the center of gravity of the so modified set. This is illustrated in Figure 7, assuming three fuzzy sets.

$$S_l^4 = \int_{u \in U_C} (sup_k (P_k^l \cdot \mu_{v_k^c}(u) \cdot u)) du = \sum_{k=1}^{|D_c|} \bar{\alpha}_k \cdot \bar{\zeta}_k$$

$$s_l^4 = \int_{u \in U_C} (sup_k (P_k^l \cdot \mu_{v_k^c}(u))) du = \sum_{k=1}^{|D_c|} \bar{\alpha}_k$$

Note that δ_4 is generally different from the other inferences unless all fuzzy sets of the decision variable are disjoint.

Figure 7 Illustration of computation of max-center formulation.



The next three inferences take a more simplistic approach – they only consider leaves which are most satisfied by e_j (such a dominant leaf is denoted τ). They differ by the method of inferring the decision in that single leaf. δ_5 uses only the center of gravity of the fuzzy set with the highest membership among the training examples of that leaf, δ_6 uses the sum-gravity, and δ_7 uses the max-gravity center [4][26] in that leaf.

$$\delta_5 = \frac{\alpha_\kappa^\tau \cdot \zeta_\kappa^\tau}{\alpha_\kappa^\tau} = \frac{S_{l_\tau}^1}{s_{l_\tau}^1} = \frac{P_\kappa^{l_\tau} \cdot \alpha_\kappa^{\tau} \cdot \zeta_\kappa^\tau}{P_\kappa^{l_\tau} \cdot \alpha_\kappa^\tau} = \frac{S_{l_\tau}^2}{s_{l_\tau}^2} = \zeta_\kappa^\tau$$

$$\delta_6 = \frac{\sum_{k=1}^{|D_c|}(P_k^{l_\tau} \cdot \alpha_k \cdot \zeta_k)}{\sum_{k=1}^{|D_c|}(P_k^{l_\tau} \cdot \alpha_k)} = \frac{S_{l_\tau}^3}{s_{l_\tau}^3} \qquad \delta_7 = \frac{\sum_{k=1}^{|D_c|}(\bar{\alpha}_k^\tau \cdot \bar{\zeta}_k)}{\sum_{k=1}^{|D_c|}\bar{\alpha}_k} = \frac{S_{l_\tau}^4}{s_{l_\tau}^4}$$

Finally, δ_8 uses sum-gravity to combine information from all leaves, when each leaf disregards all but the its highest-membership fuzzy set.

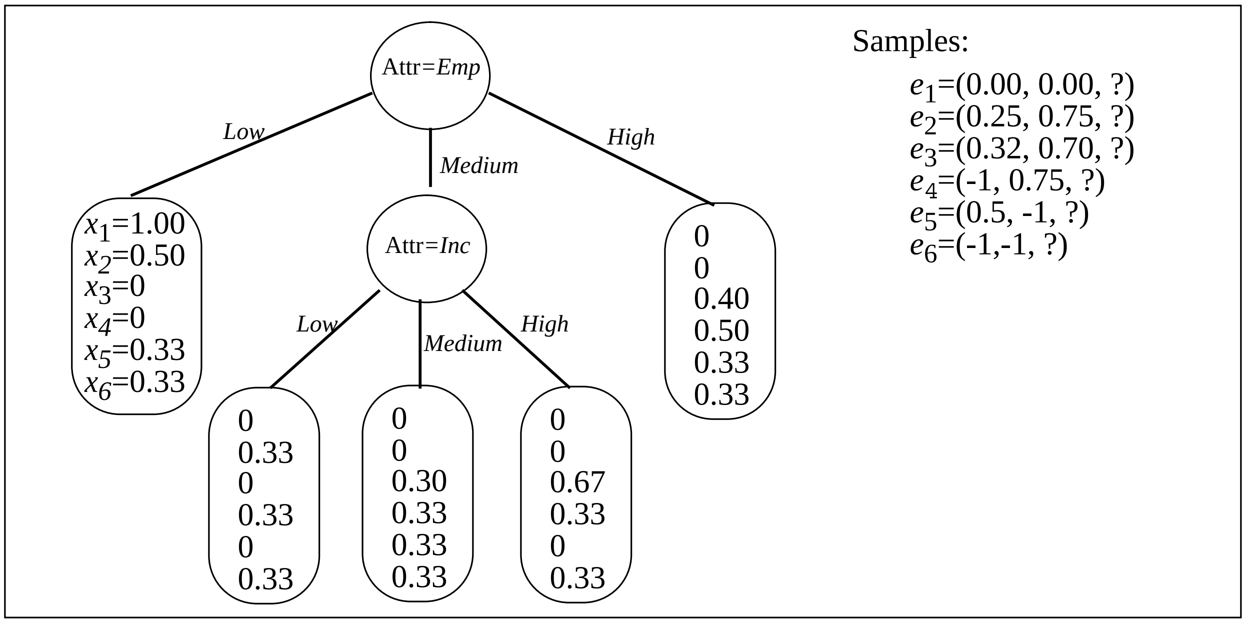
$$\delta_8 = \frac{\int_{u \in U_c} (sup_k(max_{l \in L}(P_k^l) \cdot \mu_{v_k^c}(u)) \cdot u)\,du}{\int_{u \in U_c} (sup_k(max_{l \in L}(P_k^l) \cdot \mu_{v_k^c}(u)))\,du} = \frac{\sum_{k=1}^{|D_c|}(max_{l \in L}(\bar{\alpha}_k^l) \cdot \bar{\zeta}_k^l)}{\sum_{k=1}^{|D_c|} max_{l \in L}(\bar{\alpha}_k^l)}$$

If the sample e_j to be classified does not satisfy any leaves, for example when missing training examples prevented some branches from being generated, no decision can be reached. This is clearly not acceptable if a decision must be taken. An alternative approach would be to report some statistical information from the tree. However, we may utilize domain type information to do better than that, using some similarity measure (see Section 2). Therefore, when a sample does not satisfy any leaves, we force all but nominal variables to be treated as if all memberships would always evaluate to zero (*i.e.*, we fuzzify the actual value of the attribute to $\bar{u}_p^i$ – see section 5.4). If this approach still does not produce a decision, we relax all restrictions imposed on such variables: membership is assumed to be 1 for any restriction with a non-zero membership (if such exists), and $1/|D_i^N|$ for all other sets labeling the restrictions out of the same node. In other words, we modify g_0 of section 5.4 to always use case 3, except for nominal attributes for which either case 1 or 2 is used, as needed. This will always guarantee a decision.

As an illustration, consider the previously built tree (section 5.3) and the six samples listed in Figure 8 (these are different from the training data). To assign a classification to a sample, we must first determine which leaves can contribute to making the decision. This is done by computing satisfactions of all leaves for each of the samples, using the fuzzy sets and the fuzzy restrictions from the tree. Results of this computation are illustrated in Figure 8, separately for each of the six samples, using g_0 as previously defined and g_1=*max*. It can be seen that some samples are so typical that only one leaf can deal with them (such as e_1), while others satisfy multiple leaves (e_2 and e_3). When some features are not available (as in e_4 and e_5), more of the leaves get satisfied. e_6 is a very special case when nothing is known about the sample. As expected, all leaves get satisfied equally.

The next step is to apply the selected inference. The choice of inference will determine which of the leaves with non-zero truth value will actually contribute to the decision, and how this will be done. For example, δ_3 will take centers of gravity of the sets (determined by classification of the training examples in those leaves) from all satisfied leaves, and then it will compute the center of gravity of these. Using this inference, the six samples will generate the following decisions: $\delta_3(e_1) = 0.00$ (*i.e.*, the *No* decision), $\delta_3(e_2) = 0.00$, $\delta_3(e_3) = 0.71$, $\delta_3(e_4) = 0.63$, $\delta_3(e_5) = 0.59$, and $\delta_3(e_6) = 0.51$. A very interesting observation can be made regarding the last sample, which was completely unknown. The decision is not exactly 0.50 despite that both classes were equally represented in the training set (see Figure 4). This is due to the fact that the actual counts of the training examples in the leaves differ from those in the root (as pointed out in sections 5.2 and 5.3). In fact, it can be verified in Figure 5 that the total in-leaf count for examples of the *No* class is 3.13, while the total count for the *Yes* class is 3.2. On the other hand, if we require that the sum of all memberships for any u value is 1, the counts for the examples in the tree would equal those counts for the original examples and $\delta_3(e_6)$ would be exactly 0.5 in this case. Which method should be used is an open question, to be investigated in the future.

Figure 8 Memberships in the sample tree for six selected samples (-1 indicates missing data, ? indicates the sought classification). Each node lists the values $x_1^{Node} \ldots x_6^{Node}$.



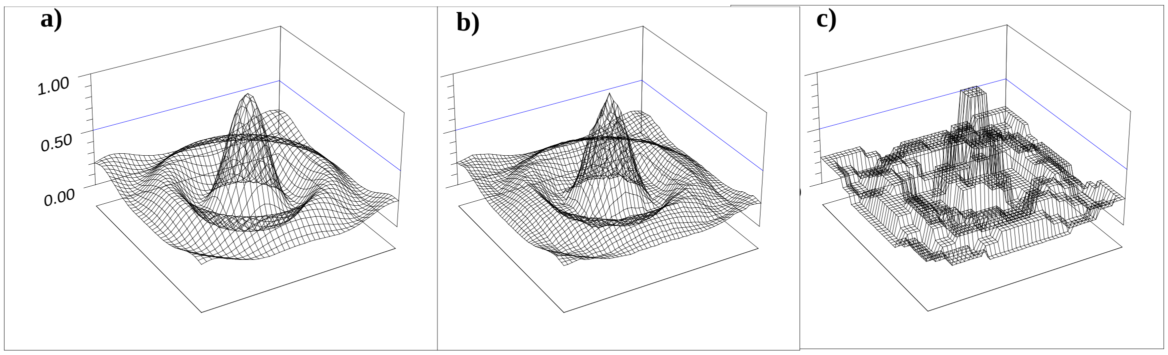
6 Illustrative Experiments

We have conducted a number of preliminary experiments with a pilot implementation FIDMV [10][12][13]. Here we report selected results. These are intended to illustrate system characteristics and capabilities. More systematic testing is being conducted and will be reported separately.

6.1 Continuous Output

Reference [35] presents a fuzzy-neural network designed to produce interpolated continuous output from fuzzy rules. The *Mexican Sombrero* function, illustrated in Figure 9a, was used to test the network's capabilities to reproduce the desired outputs. This was accomplished with thirteen nets, combined to produce a single output. In the illustrative experiment, data from a 13x13 grid was used. Each of the two domains was partitioned with thirteen triangular fuzzy sets. Function values were partitioned with seven fuzzy sets. The rules were generated by hand and assigned to the thirteen networks, whose combined outputs would produce the overall response. In our experiment, we attempted to reproduce the same nonlinear function, but in a much simpler setting. We used exactly the same data points as the training set, and the same fuzzy sets. We trained the fuzzy tree, and then we tested its acquired knowledge using a denser testing grid in order to test inferences of the tree both on samples previously seen in training as well as on new ones. Knowledge represented with δ_3 is shown in Figure 9b. It can be seen that the overall function has been recovered except for details, which were not captured due to suboptimal choice for the fuzzy sets (as pointed out in [13]). The same fuzzy tree, in conjunction with the least-information-carrying δ_5, produced the surface of Figure 9c. It is also important to point out that the fuzzy-neural system architecture was tailored to this particular problem, and that the rules were generated by hand, while the fuzzy decision tree did not require any particular data nor initial rules.

Figure 9 The original function (a), and responses of the same fuzzy tree with two different inferences.



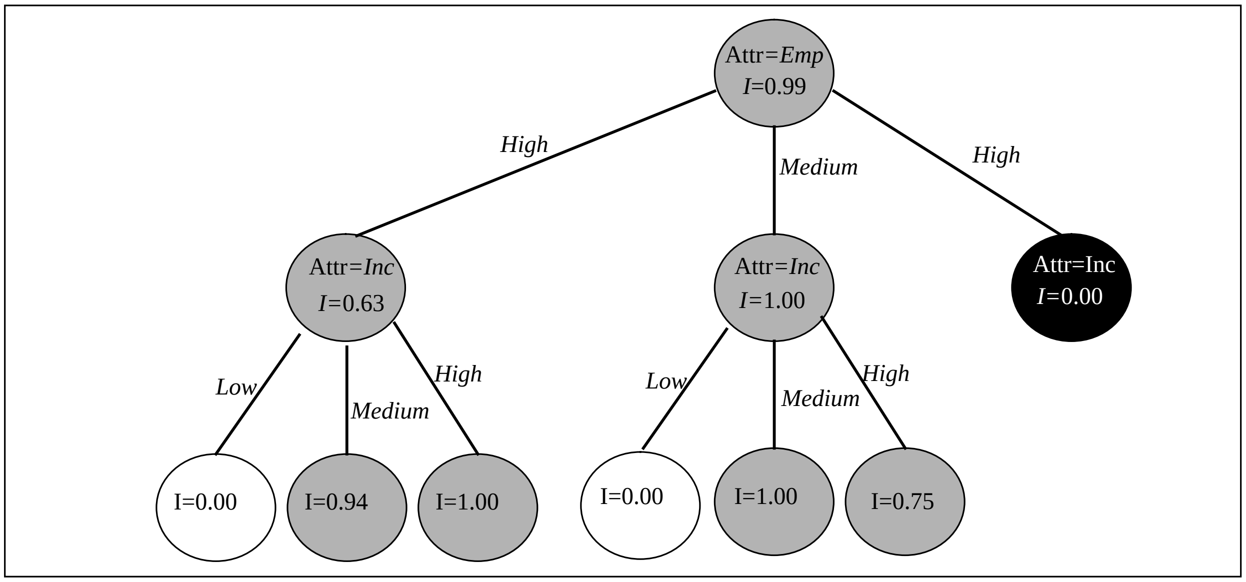
6.2 Noisy and Missing/Inconsistent Data

As another illustration, let us continue the example used to illustrate the tree-building and inference routines (sections 5.3 and 5.4). However, let us complicate it to say that the training example e_4 had its decision raised to 0.5 (undecided about credit), and that for some reason the employment status for example e_5 could not be determined.

By carefully analyzing the space as illustrated in Table 1, one could expect that the credit decision for fully employed people should not be affected (give them credit). On the other hand, the decision for the unemployed may be slightly affected toward giving credit (e_5, for which we do not know the employment status, was given credit). Changes in the decisions for the *PT* employment status are very difficult to predict.

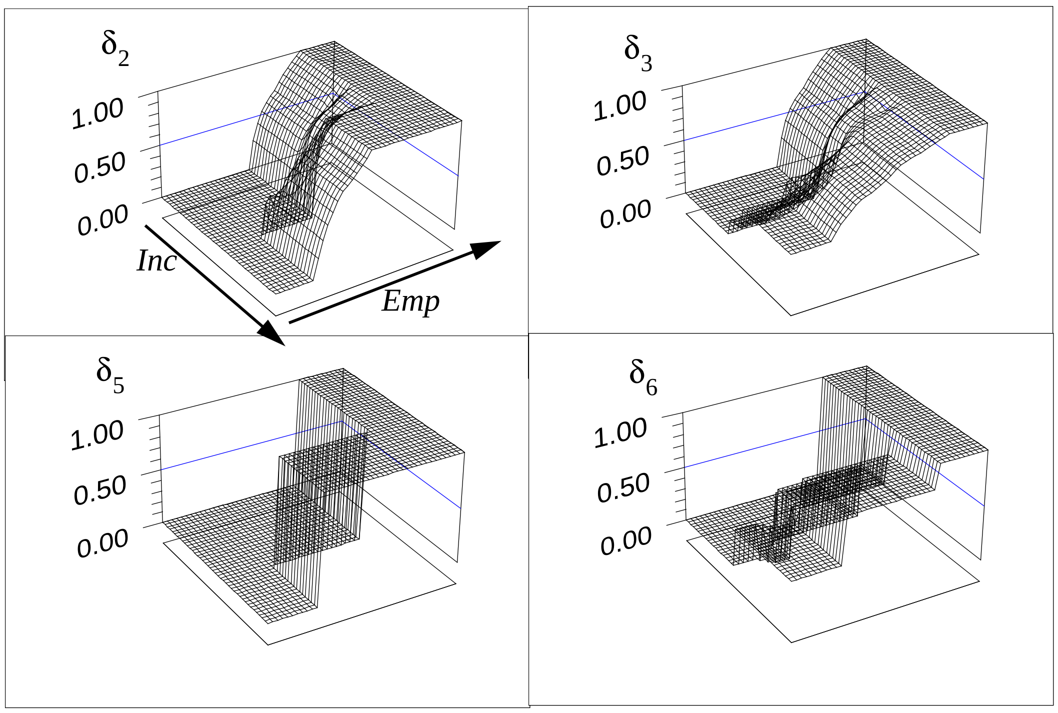
Using the same parameters as those in section 5.3, the new tree of Figure 10 is generated. From this tree, it can be verified that indeed the decision for full employment was not affected, while the other cases are much less clear – even income cannot unambiguously determine whether to give credit. What is the actual knowledge represented in this tree? As indicated before, it depends on the inference method applied to the tree. Response surfaces for four selected inferences are plotted in Figure 11. These plots illustrate the fact that full employment guarantees credit, and that otherwise the level of income helps but does not guarantee anything. Moreover, based on the inference used, different knowledge details are represented. For example, using only best-matched leaves and only best decisions from such leaves (δ_5) produces sudden discontinuities in the response surface. This behavior is very much similar to that of symbolic systems such as ID3.

Figure 10 Graphical illustration of the generated tree.:



While the structure of the tree depends on the fuzzy sets used, Figure 11 also indicates that the actual definitions for those sets affect the resulting response surface. Therefore, further research into optimization and off or on-line data-driven domain partitioning is very important. Another important observation is that some inferences behave as filters removing the effects of noisy/ incomplete data. This observation was more heavily explored in [10].

Figure 11 Illustration of the inferences.



6.3 Summary and Future Research

We proposed to amend the application-popular decision trees with additional flexibility offered by fuzzy representation. In doing so, we attempted to preserve the symbolic structure of the tree, maintaining knowledge comprehensibility. On the other hand, the fuzzy component allows us to capture concepts with graduated characteristics, or simply fuzzy. We presented a complete method for building the fuzzy tree, and a number of inference procedures based on conflict resolution in rule-based systems and efficient approximate reasoning methods. We also used the new framework to combine processing capabilities independently available in symbolic and fuzzy systems.

As illustrated, the fuzzy decision tree can produce real-valued outputs with gradual shifts. Moreover, fuzzy sets and approximate reasoning allow for processing of noisy and inconsistent/ incomplete data. The effect of such inferior data can be controlled by utilizing various inference methods. These inferences, as well as actual behavior in naturally noisy/incomplete domains, has to be empirically evaluated and will be subsequently reported.

In the future, more inference procedures will be proposed, and they all will be empirically evaluated. We are currently designing a preprocessing procedure based on CART, which will be used to propose the initial coverings. These sets will eventually be optimized in the tree-building procedure. We also plan to explore on-line domain partitioning. Finally, a number of empirical studies are currently under way.

Acknowledgments

7 Bibliography

[1] L. Breiman, J.H. Friedman, R.A. Olsen & C.J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

[2] P. Clark & T. Niblett. *"Induction in Noisy Domains"*. In *Progress in Machine Learning*, I. Bratko & N. Lavrac (eds). Sigma Press, 1987.

[3] T.G. Dietterich, H. Hild & G. Bakiri. *"A Comparative Study of ID3 and Backpropagation for English Text-to-Speach Mapping". Proceedings of the Interantional Conference on Machine Learning*, 1990.

[4] D. Drinkov, H. Hellendoorn & M. Reinfrank. *An Introduction to Fuzzy Control*. Springer-Verlag, 1996.

[5] B.R. Gaines. *"The Trade-off Between Knowledge and Data in Knowledge Acquisition"*. In G. Piatesky-Shapiro and W.J. Frawley *Knowledge Discovery in Databases*. AIII Press/MIT Press, 1991, pp. 491-505.

[6] R. Gallion, D.C. St.Clair, C. Sabharwal & W.E. Bond. *"Dynamic ID3: A Symbolic Learning Algorithm for Many-Valued Attribute Domains"*. In *Proceedings of the 1993 Symposium on Applied Computing*. ACM Press, 1993, pp. 14-20.

[7] I. Enbutsu, K. Baba &N. Hara. *"Fuzzy Rule Extraction from a Multilayered Neural Network"*. In *International Joint Conference on Neural Networks*, 1991, pp. 461-465.

[8] R. Jager. *Fuzzy Logic in Control*. Ph.D. dissertation, Technische Universiteit Delft, 1995.

[9] J. Jang. *"Structure Determination in Fuzzy Modeling: a fuzzy CART approach"*. In *Proceedings of the IEEE Conference on Fuzzy Systems*, 1994, pp. 480-485.

[10] C.Z. Janikow. *"Fuzzy Processing in Decision Trees"*. In *Proceedings of the Sixth International Symposium on Artificial Intelligence*, 1993, pp. 360-367.

[11] C.Z. Janikow. *"Learning Fuzzy Controllers by Genetic Algorithms"*. In *Proceedings of the 1994 ACM Symposium on Applied Computing*, 1994, pp. 232-236.

[12] C.Z. Janikow. "*A Genetic Algorithm Method for Optimizing the Fuzzy Component of a Fuzzy Decision Tree*". In *GA for Patter Recognition*, S. Pal & P. Wang (eds.), CRC Press, pp. 253-282.

[13] C.Z. Janikow. "*A Genetic Algorithm Method for Optimizing Fuzzy Decision Trees*". *Information Sciences*, 89(3-4), pp. 275-296, March 1996.

[14] G. Kalkanis & G.V. Conroy. *"Principles of Induction and Approaches to Attribute-Based Induction". Knowledge Engineering Review* 6(4), 1991, pp. 307-333.

[15] A. Kandel & G. Langholz (eds.). *Fuzzy Control Systems*. CRC, 1993.

[16] I. Konenko, I. Bratko & E. Roskar. *"Experiments in Automatic Learning of Medical Diagnostic Rules"*. Technical Report, J. Stefan Institute, Yugoslavia, 1994.

[17] K. Kobayashi, H. Ogata & R. Murai. *"Method of Inducing Fuzzy Rules and Membership Functions"*. In *Proceedings of the IFAC Symposium on Artificial Intelligence in Real-Time Control*, 1992, pp. 283-290.

[18] M. Lebowitz. *"Categorizing Numeric Information for Generalization". Cognitive Science*, Vol. 9, 1985, pp. 285-308.

[19] P.E. Maher & D.C. St.Clair. *“Uncertain Reasoning in an ID3 Machine Learning Framework”*. In *Proceedings of the Second IEEE International Conference on Fuzzy Systems*, 1993, pp. 7-12.

[20] D. McNeill & P. Freiberger. *Fuzzy Logic*. Simon & Schuster, 1993.

[21] R.S. Michalski. *“A Theory and Methodology of Inductive Learning”*. In *Machine Learning I*, Michalski, R.S., Carbonell, J.G. & Mitchell, T.M. (eds.). Morgan Kaufmann, 1986, pp. 83-134.

[22] R.S. Michalski. *“Learning Flexible Concepts”*. In *Machine Learning III*, R. Michalski & Y. Kondratoff (eds). Morgan Kaufmann, 1991.

[23] R.S. Michalski. *“Understanding the Nature of Learning”*. In *Machine Learning: An Artificial Intelligence Approach*, R. Michalski, J. Carbonell & T. Mitchell (eds.), Vol, II, pp. 3-26. Morgan Kaufmann, 1986

[24] J. Mingers. *“An Empirical Comparison of Prunning Methods for Decision Tree Induction”*. *Machine Learning* 4, 1989, pp. 227-243.

[25] J. Mingers. *“An Empirical Comparison of Selection Measures for Decision-Tree Induction”*. *Machine Learning* 3, 1989, pp. 319-342.

[26] M. Mizumoto. *“Fuzzy Controls Under Various Fuzzy Reasoning Methods”*. *Information Science*, 45, 1988, pp. 129-151.

[27] G. Pagallo & D. Haussler. *“Boolean feature Discovery in Empirical Learning”*. *Machine Learning*, V5, No1, 1990, pp. 71-100.

[28] J.R. Quinlan. *“The Effect of Noise on Concept Learning”*. In *Machine Learning II*, R. Michalski, J. Carbonell & T. Mitchell (eds). Morgan Kaufmann, 1986.

[29] J.R. Quinlan. *“Induction on Decision Trees”*. *Machine Learning*, Vol. 1, 1986, pp. 81-106.

[30] J.R. Quinlan. *“Decision Trees as Probabilistic Classifiers”*. In *Proceedings of the Fourth International Workshop on Machine Learning*, 1987, pp. 31-37.

[31] J.R. Quinlan. *“Anknown Attribute-Values in Induction”*. In *Proceedings of the Sixth International Workshop on Machine Learning*, 1989, pp. 164-168.

[32] J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA. 1993.

[33] C. Schaffer. *“Overfitting Avoidance as Bias”*. *Machine Learning 10* (1993), pp. 153-178.

[34] S. Sestino & T. Dillon. *“Using Single-Layered Neural Networks for the Extraction of Conjunctive Rules and Hierarchical Classifications”*. *Journal of Applied Intelligence 1*, pp. 157-173, 1991.

[35] I. Suh, Hong & T.W. Kim. *“Fuzzy Membership Function Based Neural Networks with Applications to the Visual Servoing of Robot Manipulators”*. *IEEE Transactions on Fuzzy Systems*, Vol 2, No. 3, 8/1994, pp. 203-220.

[36] L. Wang & J. Mendel. *“Generating Fuzzy Rules by Learning From Examples”*. *IEEE Transactions on Systems, Man and Cybernetics*, v. 22 No. 6, 1992, pp. 1414-1427.

[37] L.A. Zadeh. *“Fuzzy Sets”*. *Information and Control* 8 (1965), pp. 338-353.

[38] L.A. Zadeh. *“Fuzzy Logic and Approximate Reasoning”*. *Synthese* 30 (1975), pp. 407-428.

[39] L.A. Zadeh. *“A Theory of Approximate Reasoning”*. In Hayes, Michie & Mikulich (eds) *Machine Intelligence* 9 (1979), pp. 149-194.

[40] L.A. Zadeh. *“The Role of Fuzzy Logic in the Management of Uncertainity in Expert Systems”*. *Fuzzy Sets and Systems*, 11, 1983, pp. 199-227.