Learning Decision Trees in Continuous Space

J. Dombi[*] and Á. Zsiros[†]

Abstract

Two problems of the ID3 and C4.5 decision tree building methods will be mentioned and solutions will be suggested on them. First, in both methods a *Gain*-type criteria is used to compare the applicability of possible tests, which derives from the entropy function. We are going to propose a new measure instead of the entropy function, which comes from the measure of fuzziness using a monotone fuzzy operator. It is more natural and much simpler to compute in case of concept learning (when elements belong to only two classes: positive and negative).

Second, the well-known extension of the ID3 method for handling continuous attributes (C4.5) is based on discretization of attribute values and in it the decision space is separated with axis-parallel hyperplanes. In our proposed new method (CDT) continuous attributes are handled without discretization, and arbitrary geometric figures are used for separation of decision space, like hyperplanes in general position, spheres and ellipsoids. The power of our new method is going to be demonstrated on a few examples.

1 Introduction

A lot of applications on the area of artificial intelligence and data mining lead to a similar task: constructing a classification model based on our knowledge. Classification models help us to understand the underlying structure of a given problem and later they can be used to predict classes of unseen elements.

Classification models could be grouped by the way of construction: they are made by human experts or obtained from a set of examples, inductively. The built up model may be a non-hierarchical one (like a instance-based classifier, or a model obtained from a neural network, a genetic algorithm and a statistical method) or a hierarchical one like a decision tree (for a short overview and further references see [1]). We will deal with hierarchical classifiers built up inductively.

Throughout this article T denotes the given set of elements with their class information (training set) from which we would like to build up a decision tree:

$$T = \{(\mathbf{x}, c(\mathbf{x}))|\mathbf{x} \in X\},$$

[*]Department of Applied Informatics, University of Szeged, Árpád tér 2, H-6720 Szeged, Hungary, e-mail: dombi@inf.u-szeged.hu

[†]Department of Applied Informatics, University of Szeged, Árpád tér 2, H-6720 Szeged, Hungary, e-mail: zsiros@inf.u-szeged.hu

where $\mathbf{x}$ is an element of X, described by a sequence of attribute values: $\mathbf{x} = (a_1, \ldots, a_n)$, a_i is the value of the ith attribute, n is the number of attributes, and $c(\mathbf{x})$ gives the class of element $\mathbf{x}$. Finally denote $C_1, \ldots, C_k$ the possible classes of elements of T.

Usually two different types of attributes are distinguished: an attribute is discrete (categorical), if its value comes from a predefined finite set; and continuous (numerical), if it may be any element of a (real) interval.

Now, we are ready to give the definition of decision tree:

Definition 1 *Decision tree is a special rooted tree, in which a class identifier is associated to each leaf node (that determinates the class of elements reached that node), and each internal (or decision) node specifies some test, with one branch and subtree for each outcome of the test.*

Decision tree building methods might be grouped by the variety of tests used in their inner nodes. In some methods only single attribute selection is allowed as a test, for example in Quinlan's ID3 and in its extension to handle continuous attributes C4.5[1]. In other methods some mixture of attributes are also possible as tests, for instance in Cios's CID3[5] and in our new CDT method.

Thus, the final task is to make a decision tree from a given training set, where elements are described by some (discrete or continuous) attributes. Of course it is trivial to build up a tree consistent with the training set[4] (imagine a decision tree in which all leaf nodes contain only one element of the training set). But this tree does not tell us anything about the structure of the original problem. As Occam's razor tells us, we should look for one of the smallest decision trees. This philosopher idea says that a simple decision tree might perform better on unseen elements than a more complex one (see [10, 11]). Unfortunately, *the problem of finding the smallest decision tree consistent with a training set is NP-Complete[3]* as Hyafil and Rivest have already shown that.

To cope with this computational problem, a greedy, divide-and-conquer algorithm is used to build up a decision tree consistent with the training set (which, of course, usually leads not the smallest one). First, the one-node decision tree is taken (containing all elements of the training set). Later, in each step a test is selected and applied on the examples reached the current node. Then the test is assigned to the node and branches are made according to the outcomes of the selected test. Finally, the same method is applied on these newly created branches. It may be seen, that the critical point of this algorithm is the test selection criteria. The following methods (ID3, C4.5, CDT) differ only at this point.

In our CDT method a new function is used to measure the homogeneity of examples instead of the entropy function (which is used in ID3 and C4.5). It is similar to the entropy function in case of concept learning (which means that examples belong to only two classes). Unfortunately CDT can handle only such problems. On the other hand the decision space, described by a set of continuous attributes, is partitioned by arbitrary geometric figures in the CDT method, while in C4.5 only axis-parallel hyperplanes are used. These tests finally lead to a binary decision tree that correctly classifies all elements of the training set.

In the next section the test selection criteria of the ID3 and the C4.5 methods are introduced. Section 3 mentions some results from fuzzy theory that will be used later to introduce the new test selection criteria in section 4. Its work is demonstrated on an example. Finally the new decision tree building method is described in section 5 and its power is demonstrated on a few examples in section 6 followed by conclusions in section 7.

2 The ID3 and the C4.5 methods

In this section we are going to give a short description of the ID3 method, of its extension to handle continuous attributes (C4.5) and we are going to demonstrate their work through an example.

As it has been mentioned earlier, the crucial points of the tree-building algorithm are the variety of possible tests and the test selection criteria. In the ID3 and C4.5 methods tests are based on single attribute selection, so the possible tests are about the possible attributes.

The idea of test selection is to examine training examples and to find the attribute that separates the examples most perfectly considering their class membership (as the greedy approach suggests it). The ID3 algorithm uses a function coming from the field of information theory to measure the entropy in the original training set and in the subsets after partitioning. Formally the criteria is the following (where $|C_j|_T$ denotes the number of elements of T belong to class C_j):

$$\max_A\{Gain(A)\}$$

where

$$Gain(A) = I(T) - E(A,T)$$

$$I(T) = -\sum_{j=1}^{k} \frac{|C_j|_T}{|T|} \cdot \log_2\left(\frac{|C_j|_T}{|T|}\right)$$

is the entropy function and

$$E(A,T) = \sum_{i=1}^{m} \frac{|T_i|}{|T|} \cdot I(T_i)$$

is the weighted sum of the entropies of subsets. It is important to remark, that the application of entropy function and the *Gain* criteria, which is just the simple difference of $I(T)$ and $E(A,T)$, has no theoretical background, it is just a heuristic!

Unfortunately, the *Gain* criteria is biased towards discrete attributes with more outcomes, thus it has a modification, the *Gain ratio* test selection criteria (see [1]) which reduces the value of *Gain* in case of many valued discrete attributes.

The ID3 method can be used only on problems described by discrete attributes. Although it has an extension to handle continuous attributes (C4.5), in this method

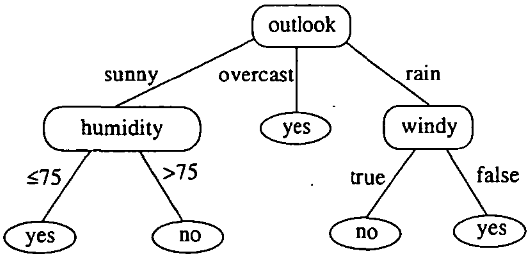

Figure 1: Decision tree built up from the training set on table 1

outlook	Temp(F)	Humidity(%)	Windy?	Class
sunny	75	70	true	Play
sunny	80	90	true	Don't Play
sunny	85	85	false	Don't Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
overcast	72	90	true	Play
overcast	83	78	false	Play
overcast	64	65	true	Play
overcast	81	75	false	Play
rain	71	80	true	Don't Play
rain	65	70	true	Don't Play
rain	75	80	false	Play
rain	68	80	false	Play
rain	70	96	false	Play

Table 1: The training set

the attributes are discretizated, handled like discrete ones with two outcomes. Different values of the candidate continuous attribute are ordered: $v_1 \leq v_2 \leq \ldots \leq v_n$ and all $\frac{v_i+v_{i+1}}{2}$, $i = 1, \ldots, n-1$ midpoints are checked as possible thresholds to divide the training set into two partition.

The test selection criteria in C4.5 is biased towards continuous attributes with numerous distinct values. Its examination and a modified criteria are found in [2].

There is a decision tree, built up from a 12 element training set on figure 1. This example is well-known from the literature [1, 11] and shows when to play a specific game. The test selection criteria is maximization of *Gain*, so first we choose attribute 'outlook'. Then the algorithm is called recursively to the first and third branches of the tree. Finally, we get a decision tree which classifies all training elements correctly.

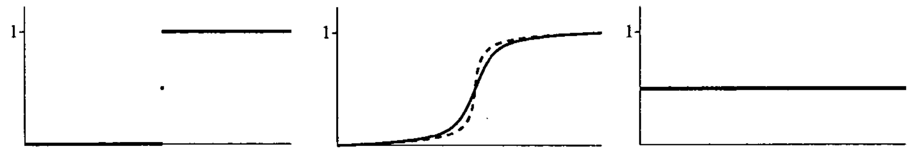

Figure 2: Membership functions with different sharpness

3 Measure of fuzziness

In this section a few basic results are mentioned about the general class of fuzzy operators and the measure of fuzziness.

There are two main classes of associative, subidempotence conjunction operators in the fuzzy theory: the class of strictly monotone and the class of nilpotens operators[6]. Monotone conjunction operators may be written in form $c(x,y) = f^{-1}(f(x)+f(y))$ where $f(x) : (0,1] \rightarrow \Re^+$, $\lim_{x\to 0} = \infty$, $f(1) = 0$ and $f(x)$ is a strictly decreasing function; and nilpotens operators in form $c(x,y) = f^{-1}([f(x)+f(y)])$ where $[x]$ is the cut function

$$[x] = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } 0 \leq x \leq 1 \\ 1 & \text{otherwise} \end{cases}$$

and $f(x) : [0,1] \rightarrow [0,1]$, $f(0) = 1$, $f(1) = 0$ is a strictly decreasing function. These general forms give us a lot of fuzzy conjunction operators: one for each suitable generator function $f(x)$. Disjunction operators could be characterized in the same manner with a bit different assumptions for generator function $f(x)$.

By the help of conjunction operators we are able to derive a function that measures the fuzziness of a (membership) function. For example in figure 2 some different membership functions are shown. The first one makes a very sharp switch from 0 to 1, so it is not fuzzy, on the second picture the dotted one is a less fuzzy while the other one is a more fuzzy function. On the third picture a total fuzzy function is shown. This property is measured with the following formula (in discrete case)[6]:

$$S = \frac{1}{n}\sum_{i=1}^{n} F(x_i)$$

where $c(x,y)$ is a conjunction operator (as above) and

$$F(x) = \frac{1}{c\left(\frac{1}{2},\frac{1}{2}\right)} \cdot c(x, 1-x).$$

This function will be useful in the next section to measure how separated the elements of the training set are.. It is also interesting to note that the entropy function, used in the ID3 method, may be obtained from this measure of fuzziness in a special case.

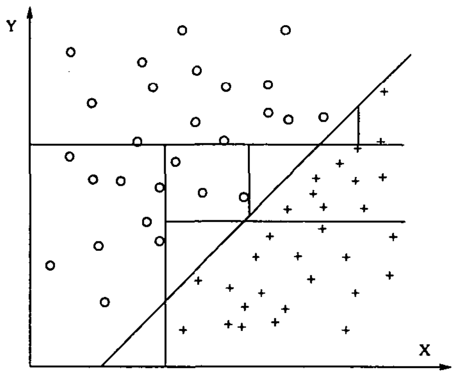

Figure 3: Geometric interpretation of classification

4 The new test selection criteria

Throughout the following sections we assume that elements of the training set belong to only two classes: $'+'$ and $'-'$!

First, we will mention the geometric interpretation of classification. Consider a set of elements described by two continuous attributes as shown in figure 3. In the C4.5 method the training set is separated at a threshold of a continuous attribute. The effect of such a decision might be interpreted as an axis-parallel line in $\Re^2$ (see figure 3). Quite a lot of C4.5-type decisions are needed for correct separation of all training elements, although one line in general position would be enough as well. Our goal is to allow arbitrary figures like hyperplane, sphere and ellipsoid to separate elements of the training set.

In our new method the starting point is the pliant operator, which is a monotone weighted fuzzy operator from Dombi[6]:

$$c_\lambda(x,u;y,v) = \frac{1}{1 + \left(u\left(\frac{1-x}{x}\right)^\lambda + v\left(\frac{1-y}{y}\right)^\lambda\right)^{\frac{1}{\lambda}}}$$

It is derived from the generalized form of mean values[7], which is closely related to the general form of fuzzy operators:

$$f^{-1}\left(\sum_{i=1}^{m} w_i f(x_i)\right), \text{ where } \sum_{i=1}^{m} w_i = 1.$$

The above mentioned operator is obtained from this general form when

$$f(x) = \left(\frac{1-x}{x}\right)^\lambda.$$

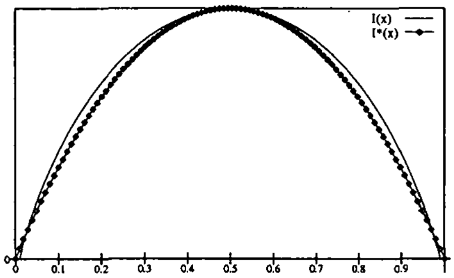

Figure 4: Entropy function for two classes and the new $4x(1-x)$ measure

We use the pliant operator when $u = v = \frac{1}{2}$ and $\lambda = 1$. In this special case it is easy to get the following measure of fuzziness:

$$I^*(x) = \frac{1}{c\left(\frac{1}{2}, \frac{1}{2}\right)} \cdot c(x, n(x)) = 4x(1-x).$$

This formula is much simpler than the entropy function and it behaves very similarly. The graph of the original and the new measure is shown on figure 4. It is easy to prove, that these two functions lead to the same result (in case of concept learning), since on interval $[0,1]$ both functions have maxima at $\frac{1}{2}$, minima at 0 and 1, they are strictly monotone on $[0, \frac{1}{2}]$ and on $[\frac{1}{2}, 1]$ and they never intersect each other. We are going to demonstrate their work on the following example, where the value of entropy function and the value of this new measure is calculated parallel.

This example, where the 14 elements of the training set is described by four attributes (two discrete and two continuous ones) and the examples belong to two classes, is taken from [1] (see table 1). First we have to calculate the entropy of the whole training set, that is

$$I(S) = -\frac{9}{14}\log\frac{9}{14} - \frac{5}{14}\log\frac{5}{14} = 0.9403.$$

With the new $I^*(x)$ measure we get

$$I^*(S) = 4 \cdot \frac{9}{14} \cdot \frac{5}{14} = 0.9184.$$

Then we try to separate the training set using the attribute 'outlook', so first we have to calculate the entropy of the three subsets (sunny, overcast, rain) $I(S_{sunny}) = -\frac{2}{5}\log\frac{2}{5} - \frac{3}{5}\log\frac{3}{5} = 0.9709$, $I(S_{overcast}) = -\frac{4}{4}\log\frac{4}{4} - 0 = 0$, $I(S_{rain}) = -\frac{3}{5}\log\frac{3}{5} - \frac{2}{5}\log\frac{2}{5} = 0.9709$ then the weighted sum of these values $E(outlook, S) = \frac{5}{14}0.9709 + \frac{4}{14}0 + \frac{5}{14}0.9709 = 0.6935$. Therefore $Gain(outlook) = 0.9403 - 0.6935 = 0.2468$. It says that we gain 0.2468 if we select attribute 'outlook' as a test.

If we apply the new measure we get the following: $I^*(S_{sunny}) = 4 \cdot \frac{2}{5} \cdot \frac{3}{5} = 0.96$, $I^*(S_{overcast}) = 4 \cdot \frac{4}{4} \cdot \frac{0}{4} = 0$, $I^*(S_{rain}) = 4 \cdot \frac{3}{5} \cdot \frac{2}{5} = 0.96$ and the weighted

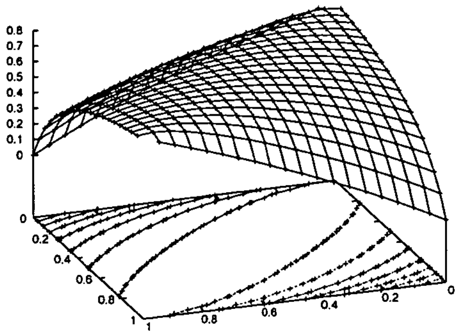

Figure 5: Test selection criteria

sum of these values is $E^*(outlook, S) = \frac{5}{14}0.96 + \frac{4}{14}0 + \frac{5}{14}0.96 = 0.6857$. Therefore $Gain^*(outlook) = 0.9184 - 0.6857 = 0.2327$. It is close to the previous result.

Now, we have to calculate the gain of discrete attribute 'windy' and the gain of continuous attributes 'temp' and 'humidity'. The selection criteria is maximization of gain, so finally we choose attribute 'outlook'. It is fairly a good choice, because nearly third of the training set is classified correctly. If we continue the algorithm on the first and third branch of the tree, we get the decision tree shown in figure 1.

5 The CDT method

In the previous section we introduced a new $I^*(x)$ measure instead of information entropy on the special case, if elements of training set belong to only two classes. Using this measure we are able to build up formulas similar to the ones in the ID3 method to compare the gain of possible tests. The result of our calculations for $E^*(t, S)$ is shown in figure 5. This works only for tests with two outcomes but decisions with geometric figures behave in this manner.

Our next problem is to describe the required geometric figures such as hyperplane, sphere and ellipsoid with bounded parameters. The geometric interpretation of classification shows that, if the training examples are described by n continuous attributes, the elements might be represented by points in the $\Re^n$ space. So we can calculate the bounding box of training examples as shown in figure 6. Denote R_{max} the distance of the farthest points of this bounding box from the center of it. Each point P in the sphere (with origin O and radius R_{max}) determines a hyperplane, with normal vector $\overrightarrow{OP}$ and one point P, that separates points of the training set:

$$f_s(\mathbf{p}, \mathbf{o}, \mathbf{x}) = \mathbf{n}(\mathbf{x} - \mathbf{p}) = \frac{\mathbf{p} - \mathbf{o}}{\|\mathbf{p} - \mathbf{o}\|_E}(\mathbf{x} - \mathbf{p}) = 0$$

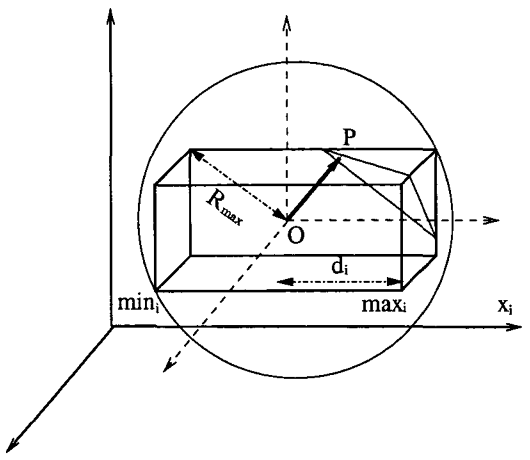

Figure 6: Description of hyperplane

where $p_i \in (O_i - R_{max}, O_i + R_{max})$. Other geometric figures may be described similarly.

Our last task is to count examples on one side of a geometric figure. Instead of taking strict bounds we use the well-known sigmoid function:

$$\sigma_1(\mathbf{x}, \mathbf{a}) = \frac{1}{1 + e^{-\lambda f(\mathbf{x}, \mathbf{a})}}$$

where $f(\mathbf{x}, \mathbf{a}) = 0$ describes the geometric figure (hyperplane, sphere or ellipsoid), vector $\mathbf{a}$ contains the parameters of the figure and λ gives the sharpness of the bound. This $\sigma_1(\mathbf{x}, \mathbf{a})$ function helps us counting examples belong to class C in one side of function $f(\mathbf{x}, \mathbf{a})$:

$$S_1^C(\mathbf{a}) = \frac{1}{|C|_T} \sum_{c(\mathbf{x}_i)=C} \sigma_1(\mathbf{x}_i, \mathbf{a}).$$

Now, we are ready to separate elements of the continuous space with arbitrary geometric figures, counting elements of the training set continuously (using the sigmoid function) and compare figures with different parameters (and compare different figures) using the *Gain*-type decision criteria. Finally, we should note that the parameters of the searched figures are found by global maximization of the *Gain* function, using stochastic global optimization technique with a specific local optimization method.

6 Examples

In this section we consider some two-dimensional examples to demonstrate the work of our new method introduced in the previous section. First, take the training set

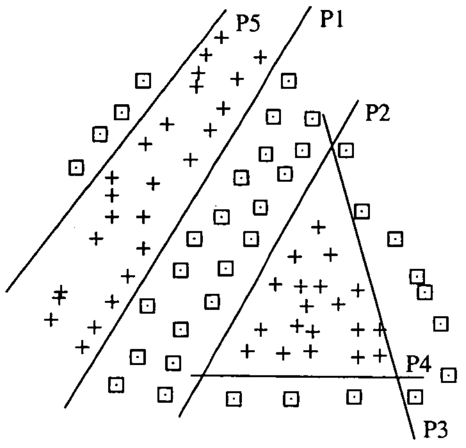

Figure 7: The extended triangle problem

shown in figure 7. This is the triangle problem extended with a few extra points. In this example we use only lines to separate elements of training set. The most interesting part of this example is the separation of the triangle. It is solved by three lines correctly, while in the C4.5 method (where only axis-parallel lines are allowed) about ten decisions are needed. (This example is also examined in [5]).

Another well-known hard task is the problem of two spirals on figure 8. The decision tree on the right side of the figure shows only the first few steps of the tree building method because this problem gives a large decision tree. As it can be seen, the elements of the training set have been quite well separated and the structure of the problem has already been realized in these very first steps. It is easy to imagine that the C4.5 method gives a very complex decision tree to solve this problem.

A further example could be a training set where most of the points are class '+' and only a few points are class '−' in a small group. Then the '−' elements are taken out by a circle decision in our CDT method, while in C4.5 minimum four decisions are required to solve this problem.

7 Conclusions

Our new method is the generalization of the C4.5 algorithm in the following sense:

In C4.5 the continuous decision space is separated only with hypercubes whose edges are parallel with the coordinate axes, while in our new method arbitrary geometric figures like hyperplane, sphere and ellipsoid are allowed. The advantage of this improvement has been demonstrated on the previous examples. Of course

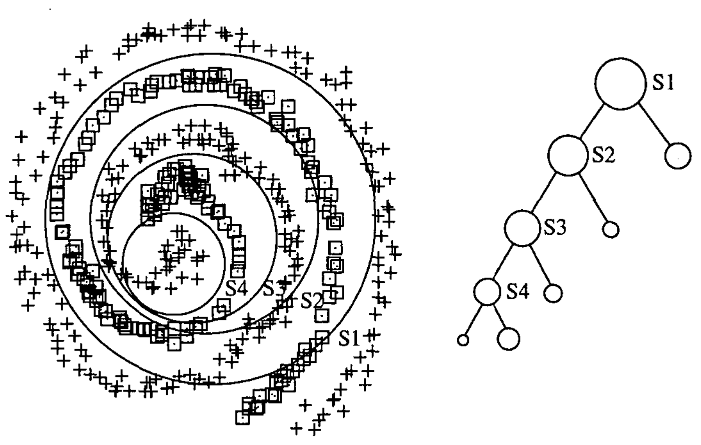

Figure 8: The problem of two spirals

it increases the computational complexity of the algorithm, but it also enlarges the variety of possible tests.

The entropy function is replaced with a simpler fuzzy measure (coming from a monotone fuzzy operator) and the test selection criteria is derived from this simpler measure. This simplification reduces the running time of the algorithm.

In our new method the elements of the training set are counted continuously, which means that the elements close to the bound of the figure belong to both regions (with different weights). Therefore, the decision criteria is a continuous function of the parameters of figure. This property makes the global optimization problem (finding parameters of geometric figure) simpler.

There are some restrictions in the current version of our new method: first, we use only continuous attributes. It is also possible to handle ordered discrete attributes but obviously it is impossible to handle unordered ones in this way. Second, we assumed, that all elements of the training set belong to two classes, so only concept learning is examined. Third, all the tests we use separate the training set into two smaller groups. Multivalued tests are impossible.

Finally, a few more improvements are going to be mentioned. First, it is worth to replace the general purpose global optimizer used in this method with a problem specific one, because it will enlarge the efficiency of the algorithm. Second, the running time on large datasets might be rather long, so in this case a sample of the training examples should be used to build up a tree. Our experiments shows that the algorithm is quite quick on a few hundred or thousand examples, but rather slow on larger (more hundred thousand elements) dataset. Third, sometimes the built up decision trees are too complex, so it is worth to apply some pre- or post-pruning[8, 9] method to simplify the result tree.

References

[1] J. R. Quinlan, *C4.5 Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

[2] J. R. Quinlan, Improved Use of Continuous Attributes in C4.5, *Journal of Artificial Intelligence Research*, **4** (1996) 77-90.

[3] L. Hyafil and R. L. Rivest, Constructing optimal binary decision trees is NP-complete, *Information Processing Letters*, **5** (1976) 15-17.

[4] Stuart J. Russel and Peter Norvig, *Artificial Intelligence, a Modern Approach*, Prentice Hall, 1995.

[5] Krzysztof J. Cios and Ning Liu, A Machine Learning Method for Generation of a Neural Network Architecture: A Continuous ID3 Algorithm, *IEEE Transactions on Neural Networks*, **3** (1992) 280-290.

[6] Dombi J., A general class of fuzzy operators, the DeMorgan class of fuzzy operators and fuzziness measures induced by fuzzy operators, *Fuzzy Sets & Systems*, **8** (1982) 149-163.

[7] G. H. Hardy, J. E. Littlewood and G. Pólya, *Inequalities*, Cambridge University Press, 1934.

[8] Floriana Esposito, Donato Malerba and Giovanni Semeraro, A Comparative Analysis of Methods for Pruning Decision Trees, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19** (1997) 476-491.

[9] Jim Kay, Comments on Esposito et al.,*IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19** (1997) 492-493.

[10] Michael J. Kearns and Umesh V. Vazirani, *An Introduction to Computational Learning Theory*, The MIT Press, 1994.

[11] Tom M. Mitchell, *Machine Learning*, The McGrawn-Hill Companies, Inc., 1997.