FUZZY DECISION TREE LEARNING FOR PREOPERATIVE CLASSIFICATION OF ADNEXAL MASSES

Emad Ahmadi
Digestive Diseases Research Center, Shariati Hospital, Tehran University of Medical Sciences, 14117 Tehran, Iran

Hoda Javadi
Medical Education and Development Center, Iran University of Medical Sciences, 14496 Tehran, Iran

Amin Khansefid
Department of Electrical and Computer Engineering, University of Tehran, 14396 Tehran, Iran

Atousa Asadi, Mohammad Mehdi Ebadzadeh
Department of Computer Engineering, Amirkabir University of Technology, 15875 Tehran, Iran

Dirk Timmerman
Department of Obstetrics and Gynecology, University Hospitals Leuven, Herestraat 49, B-3000 Leuven, Belgium

Keywords: Decision tree learning, Fuzzy logic, Adnexal diseases/diagnosis.

Abstract: The study problem was learning a fuzzy decision tree to classify patients with adnexal mass into either of benign or malignant class prior to surgery using patients' medical history, physical exam, laboratory tests, and ultrasonography. A learning algorithm was developed to learn a fuzzy decision tree in three steps. In the growing step, a binary decision tree was learned from a dataset of patients while fuzzy discretization was used in decision nodes testing continuous attributes. The best degree of fuzziness was automatically found by an algorithm based on optimization procedures. In the pruning step, the overfitted nodes were removed by an algorithm based on critical value post-pruning method. In the refitting step, the labels of the leaf nodes were optimized. The final resulted tree had 10 decision nodes and 11 leaf nodes. Performance testing of the tree gave AUC of ROC of 0.91 and mean squared error of 0.1. The tree was translated into a set of 11 fuzzy if-then rules and the clinical plausibility of the rules was assessed by domain experts. All rules were verified to be in agreement with medical knowledge in the domain. Despite the small learning set and the lack of some important input variables, this method gave accurate and, more importantly, clinically interpretable results.

1 INTRODUCTION

In the pool of more than 20 diseases causing adnexal mass, malignant lesions should be differentiated from benign lesions, because benign lesions should not undergo surgery unless being symptomatic or causing subfertility while malignant lesions should be removed by surgery (Hoffman, 2009). Ovarian cancers, comprising the majority of malignant adnexal masses, can spread quickly in the abdominal cavity and involve organs like diaphragm and bowel (Schaffer, 2008). Performing surgery on such organs is beyond the scope of general gynecology; therefore patients with malignant adnexal mass should be operated by gynecologic oncologists who have sufficient expertise in such operations (Mann et al., 2009). Thus, malignant and benign adnexal masses

Ahmadi E., Javadi H., Khansefid A., Asadi A., Mehdi Ebadzadeh M. and Timmerman D..
FUZZY DECISION TREE LEARNING FOR PREOPERATIVE CLASSIFICATION OF ADNEXAL MASSES.
DOI: 10.5220/0003179703640375
In *Proceedings of the International Conference on Health Informatics* (HEALTHINF-2011), pages 364-375
ISBN: 978-989-8425-34-8

should be differentiated prior to surgery in order to refer patients with malignant lesions to gynecologic oncologists as well as withholding surgery for innocent benign lesions.

No single imaging or laboratory study has been able to accurately differentiate malignant from benign adnexal masses (Myers et al.). In the meantime, there are few experts who can accurately differentiate malignant from benign masses prior to surgery using patient's history, physical exam, laboratory tests, and ultrasonography results. This observation leaded to the hypothesis that a combination of patient's data can accurately differentiate adnexal masses. A series of studies were tried to simulate experts' thinking process for classification of adnexal masses, but none have been implemented into routine clinical practice (Hoffman, 2009). The reason is that simple models like logistic regression are not accurate enough despite being easy to interpret by clinicians, while complex models like advanced kerned-based methods are not interpretable by clinicians despite being accurate. Ethical and legal issues do not allow clinicians to make their therapeutic decisions based on outputs coming out of black-box models without knowing how the outputs are made. Briefly, a model being both accurate and interpretable by clinicians is lacking. By overviewing medical textbooks and journals, it is revealed that combinatory if-then rules and decision trees are the most widely used medical decision making methods. Having all these facts, we tried to make a decision tree for preoperative classification of adnexal masses, which can then be translated into a set of if-then rules. Because the input data was nondeterministic for predicting malignancy, fuzzy inference was used to manage uncertainty associated with the data. The resulted fuzzy decision tree was then translated into a set of fuzzy if-then rules, which are interpretable by clinicians and can be criticized and amended based on medical knowledge in the domain.

The paper is organized as follows: Section 2 introduces the basic ID3 decision tree learning algorithm, its extension to use continuous attributes as input variables, the concept of nondeterministic data and overfitting in decision tree learning, and fuzzy decision trees. Section 3 defines the learning problem and the steps used to learn a fuzzy decision tree. Section 4 explains the exact methods by which the fuzzy decision tree was learned from the dataset. Section 5 explains the post-pruning method used to eliminate overfitting. Section 6 describes a refitting method used to further improve fuzzy decision tree classification generalizability. Section 7 reviews the dataset used in this study. Section 8 presents the final parameters chosen for implementing the learning task and the results of the final tree testing. Section 9 discusses strengths of this study, inductive bias associated with decision tree learning, and the conclusion of the study.

2 INTRODUCTION TO FUZZY DECISION TREE LEARNING

An algorithm which learns a decision tree from a dataset of patients is said to *learn* a decision tree from patients' data in the *training dataset*. The target function of this algorithm is the best decision tree which can classify cases into either of benign or malignant class. The simplest situation for making decision trees is when all attributes are binary, meaning each attribute can take only either value of 0 or 1. For making decision trees using such attributes, the decision tree learning algorithm follows these steps (Mitchell, 1997):

1- Create a tree by making a root node with one left child and one right child
2- Using the first attribute, send training examples with the value of 0 to the left child and examples with the value of 1 to the right child
3- Assess the pooled purity of left and right children for output classes
4- Redo 2 and 3 for all attributes, saving the pooled purity caused by each attribute
5- Assess which attribute has resulted in the maximum pooled purity; Assign that attribute to the root
6- If the left child is pure for one class, make it a leaf node and assign its label that class
7- If the left child is not pure for one class, go to 1 and start making a subtree rooted in the left child
8- If the right child is pure for one class, make it a leaf node and assign its label that class
9- If the right child is not pure for one class, go to 1 and start making a subtree rooted in the right child

The above steps are followed by all decision tree learning algorithms to *grow* the tree by making children for nodes recursively, until 6 or 8 is met, where the node is turned into a leaf and no more children are made for the node. A binary attribute selected for a node is eliminated from the list of attributes which can be used by the descendents of that node.

2.1 Nondeterministic Data and Overfitting

Other than the variables recorded in the dataset, there might be other variables affecting the output which are not recorded in the dataset. This is the usual case in modeling medical problems, where the model has to predict the output using some attributes while none of the attributes have a direct effect on the output (and not directly caused by the output). In this study, the only attribute which can directly determine the malignancy is the pathology results, but it cannot be used as an input in our model (because measuring this attribute needs surgery and is invasive). As a result, the model has to predict the malignancy by using attributes which are neither directly caused by the malignancy nor have direct effect on the malignancy, but are noticed to have interactions with the malignancy (e.g. malignant tumors often, but not always, have bigger sizes).

When data is not deterministic, the output cannot be absolutely predicted by any combination of the attributes. Thus, even the best models will have a degree of inaccuracy, called *residual error*.

When residual error is present, even the best attributes in the final test nodes cannot make absolutely pure children. If residual error is not recognized, the learning algorithm tries to make absolutely pure leaves while it is not possible by using any attributes. The learning algorithm continues to make children for nodes recursively, leading to small number of cases in the bottom nodes of an excessively grown tree. In this stage, because of the small number of cases in each test node, there is a high probability that one among all attributes has different values for cases of different classes, thus selecting this attribute for the node is associated with correct separation of cases in the training dataset who have reached that node (the cases are thus separated by chance, not by the selected attribute); but selecting this attribute for the node is associated with incorrect classification of cases reaching that node in subsequent testing of the tree on a separate dataset (because the same chance is improbable to occur again in testing). This learning algorithm will select irrelevant attributes for multiple bottom test nodes, resulting in an *overfitted* tree to the training dataset.

To prevent overfitting, the learner has to recognize residual error and turn the node into a leaf if a sufficient amount of purity, consistent with residual error, is met. Another approach is that the learning algorithm lets the tree to become overfitted, and then *post-prunes* the overfitted tree to make an optimal decision tree. This approach is used in this study and is introduced in section 5.

2.2 Crisp Discretization of Continuous Attributes

If attributes are continuous rather than binary, the second step of the learning algorithm becomes more elaborate. The learner should test multiple thresholds for the first attribute, sending cases with the attribute value of less than threshold to the left child, and cases with the attribute value of more than threshold to the right child. The pooled purity of children will be assessed for each threshold, and the best threshold is selected for that attribute.

Then the same process will be repeated for all attributes, assessing the pooled purity of children for each threshold of each attribute. The best attribute with its best threshold is finally assigned to the node.

2.3 Fuzzy Decision Trees

Assume a patient being classified by a conventional decision tree. In each test node, a single attribute is tested using a *single threshold* having two possible answers: less than threshold and more than threshold. The attribute space of the node (the *local attribute space*) is thus split into two non-overlapping subspaces, as shown in figure 1. Patients with value of the tested attribute less than threshold go to the left child, while cases with attribute value more than threshold go to the right child. To classify a new patient, it starts at the root node and is tested sequentially in multiple test nodes until it reaches a leaf. All patients reaching a leaf will be assigned to the same class corresponding to that leaf. In summary, each patient follows a *single path*, reaches a *single leaf*, and is assigned the class stored in that leaf.

Instead of defining crisp sets, we can define two fuzzy sets for members of the left and right children using a smooth and overlapping fuzzy discriminator function for continuous attributes tested in the test nodes (Olaru and Louis, 2003). Each fuzzy test node tests a single attribute using a pair of *two parameters* which characterize the *fuzzy discriminator function*. The two parameters are *threshold* which is the cutpoint, and *width* which defines the overlapping region of left and right children. The local attribute space is thus split into two overlapping subspaces. In a fuzzy decision tree, a case can be classified by being propagated through *multiple paths* in the tree and reaching *multiple leaves*, if the case is situated in the overlapping region of some test nodes. At the

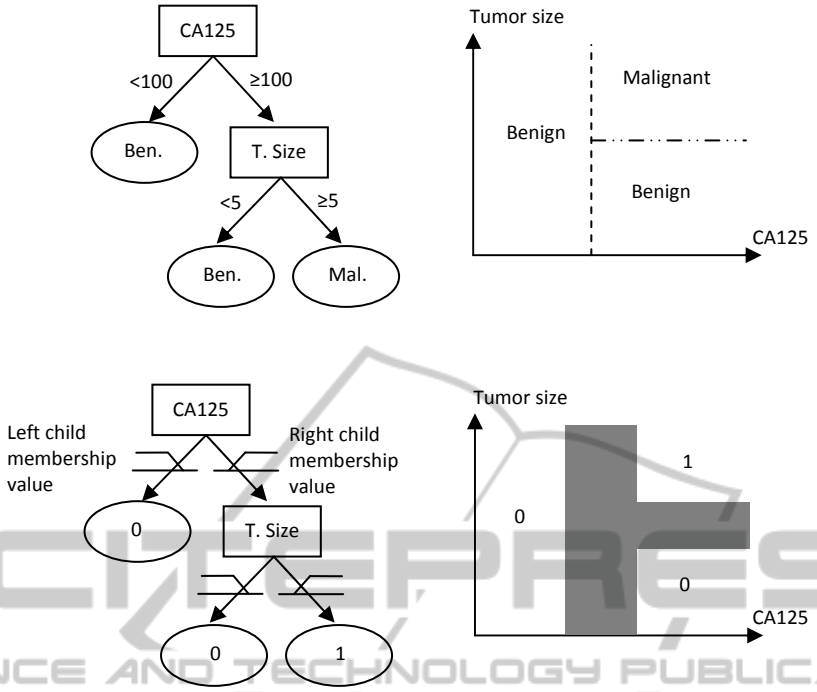

Figure 1: Conventional decision tree testing two attributes and its corresponding fuzzy decision tree. In conventional decision trees, the attribute space is partitioned into non-overlapping subspaces in which each case is assigned to a single class. In fuzzy decision trees, the attribute space is partitioned into overlapping subspaces by fuzzy boundaries. If a case is situated in the overlapping area, it may belong to both classes with different degrees of membership.

end, the case might have reached one or multiple leaves with different membership values. The class estimations given by all these leaves are then aggregated through some defuzzification process to determine the final estimated membership value of the case in each output class.

3 THE LEARNING PROBLEM

In this study, we have a concept learning problem. Let us use c to denote the malignancy concept. Then $c(p)$ denotes whether or not the patient p is a member of the malignancy class (section 5.1, equation 3), and $\hat{\mu}_C(p)$ denotes degree of membership of the patient in the malignancy class estimated by the tree.

In this study, membership-value weighted average of leaves labels was used to calculate patient's estimated membership value in the malignancy class:

$$\hat{\mu}_C(p) = \frac{\sum_{j \in leaves} \mu_j(p).L_j}{\sum_{j \in leaves} \mu_j(p)} \quad (1)$$

where $\mu_j(p)$ denotes patient's degree of membrership in the jth leaf and L_j denotes label of the jth leaf. The label of a node is the class estimation of that node for cases reaching that node. While the label of a leaf in a non-fuzzy decision trees is the name of one class, the label of a leaf in a fuzzy decision tree can be the fuzzy degree of membership in one class. When the number of output classes is only two, like in this study, we can define the label of each leaf as the fuzzy degree of membership in one class, like malignancy class in this study. The denominator equals patient's membership value in the root and is equal to one in this study.

A fuzzy decision tree is an approximation structure for computing the degree of membership of patients to a particular class, as a function of patients' attribute values. The term *attribute* is used to denote the input parameters used in the decision tree test nodes for classifying patients, the term *instances* denotes the set of all possible patients with any possible attributes values, which we denote by X, and the term *example* denotes the patient's attributes-output pairs provided in the training dataset. The process of fuzzy decision tree making was automatically done by supervised learning from

examples in the training dataset. The learning problem could then be defined as:

Learning Task. Classifying patients into either of benign or malignant class using patients' attributes

Target Function. The function c that maps the instance space to whether the patient is a member of the malignancy class:

$$c\colon X \to \{0\,,1\}$$

Target Function Representation. the function π equivalent to the fuzzy decision tree that maps the instance space to the patient's degree of membership in the malignancy class:

$$\pi\colon X \to [0\,,1]$$

Training Experience. Using examples in the training dataset to make a fuzzy decision tree to calculate $\hat{\mu}_C(p)$

Performance Measure. minimized squared error between the vector of outputs estimated by π and the vector of real outputs:

$$E_{prf} = \|[c] - [\pi]\|^2 = \big[[c] - [\pi]\big]^T.\big[[c] - [\pi]\big] \quad (2)$$

where E_{prf} denotes performance error, $[\pi]$ is the vector of the estimated class of patients estimated by π containing values in the interval [0,1], and $[c]$ is the vector of the real class of patients provided in the training dataset containing values from the set {0,1}.

For making an optimal fuzzy decision tree, a process of three steps was used (Olaru and Louis, 2003). First, a sufficiently large fuzzy decision tree was made in the *growing step*, using a subset of the dataset called the growing set (GS). In this step, test nodes are consecutively added in a top-down fashion, until one of the stopping criteria are met. At the end of this step, a large (and presumably overfitted) fuzzy decision tree is made.

Then, in the *pruning step*, the overfitted nodes of the grown tree were pruned in a bottom-up fashion. A cross-validation method was used for this step, using a separate subset of the dataset called the pruning set (PS).

Finally, in the *refitting step*, the labels of the leaves of the pruned tree were tuned to optimize the decision tree performance. This step used the whole learning set (LS), including all cases of both growing and pruning sets. At the end, the tree was tested to assess its performance on a dataset separate from the learning set.

All algorithms were coded in MATLAB programming language and implemented in MATLAB software (version 7.8.0.347 (R2009a), The MathWorks Inc).

4 GROWING METHOD

We used a modified version of the method introduced by Olaru et al (Olaru and Louis, 2003). Figure 2 shows the split of a tree node corresponding to a *fuzzy set* T into two fuzzy subsets L as the left child and R as the right child, based on the chosen attribute att in the node T. Each test node is associated with a discriminator function v which determines the degree of membership of each patient in the left child by using patient's attribute value $att(p)$. A widely used fuzzy discriminator function is the simple *linear piecewise* function, as shown in figure 3.

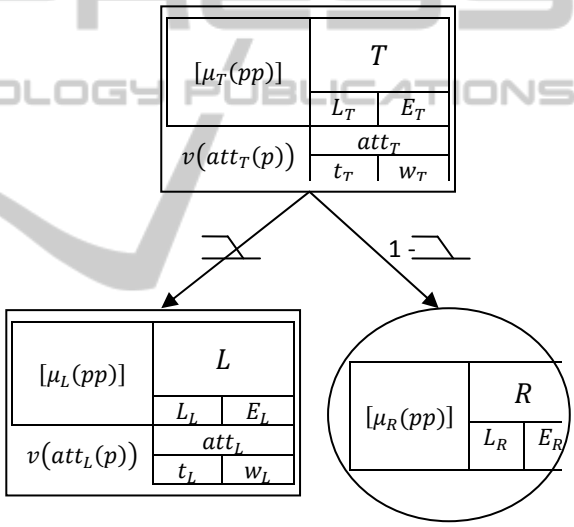

Figure 2: Fuzzy split of test node T into left and right children L and R. L_T: test node label; E_T: test node error; att_T test node selected attribute; t_T: test node attribute threshold; w_T: test node attribute width; $[\mu_T(pp)]$: patients' membership values in the test node; $v(att_T(p))$: discriminator function value for the patient □. Corresponding left and right children features are shown by subscripts L and R, respectively. The right child is a leaf and thus does not have attribute-related features. Note that $v(att_T(p))$ is not a feature stored in the test node, but can be calculated from the patient's attribute value and test node discriminator function parameters.

Left and right children membership values can then be calculated from patient's membership value in the test node $\mu_T(p)$ and patient's discriminator function value $att(p)$, which itself is dependent on the test node attribute threshold t and width w, and the

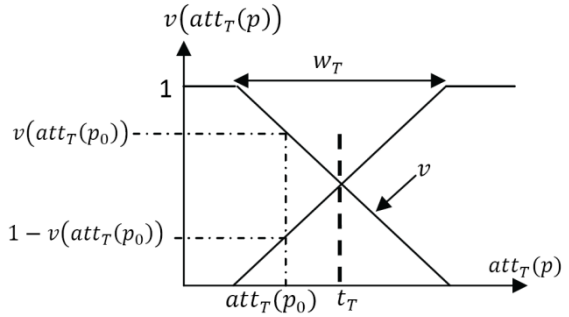

Figure 3: Linear piecewise discriminator function for test node T. v: discriminator function; $att_T(p)$: patient's attribute value; $v(att_T(p))$: left child discriminator function value; t_T: test node attribute threshold; w_T: test node attribute width. The inclining line characterizes right child discriminator function value, which can be calculated by subtracting left child discriminator function value from one.

patient's attribute value $att(p)$:

$$\mu_L(p) = \mu_T(p).v(att(p))$$

$$\mu_R(p) = \mu_T(p) - \mu_L(p) = \mu_T(p).[1 - v(att(p))]$$

$$v(att(p)) = \begin{cases} 1, & att(p) \le t - \frac{w}{2} \\ 0, & att(p) > t + \frac{w}{2} \\ \frac{t+\frac{w}{2}-att(p)}{w}, & t - \frac{w}{2} < att(p) \le t + \frac{w}{2} \end{cases}$$

The algorithm for making a fuzzy decision tree follows the basic steps described in section 2. However, some general concepts used there should be exactly defined to accommodate fuzzy concepts, including:

a. A method for selecting the best attribute for each test node, including selecting the optimum threshold and width of the fuzzy discriminator function for each continuous attribute

b. A method for assigning a label to each node

c. Exact definitions for stopping criteria

4.1 Selecting the Best Attribute for the Test Node, and Assigning Optimum Labels to Left and Right Children

Objective: given $[\mu_T(pp)]$, the vector of the membership values of all patients in node T, find the attribute att, threshold t and width w (parameters defining the discriminator function) together with left and right child labels L_L and L_R, so that the division error E_{div} is minimized in equation 3.

Because the parameters of v are still not fixed and the search algorithm has to search for the optimum values of t and w as well, the value of v for each patient is dependent on both the patient's attribute value $att(p)$ and the discriminator function parameters t and w. The concept of minimizing E_{div} in equation 3 is to select the attribute and discriminator function parameters so that the estimated class of all patients pooled in both children would have the minimum difference from the real class of all patients, if the patients are divided into left and right children by the discriminator function. If a decision tree is composed of a root with its left and right children being leafs, then E_{div} equals E_{prf}. For larger trees, E_{prf} cannot be directly minimized in each test node, and thus E_{div} was used as the closest possible approximation to E_{prf} which can be minimized in each test node.

The algorithm selects the first attribute, searches for multiple values of both t and w, reshapes the discriminator function according to them, and calculates $[\mu_L(pp)]$, the vector of membership values of all patients in the left child, and $[\mu_R(pp)]$, the vector of membership values of all patients in the right child. Then the algorithm calculates the optimum values of L_L and L_R for the selected att, t and w.

Assuming that att, t and w are selected and are temporarily fixed, the optimum values of L_L and L_R to minimize E_{div} are achieved by getting the partial derivative of E_{div} with respect to L_L and L_R and making them equal to zero:

$$\frac{\partial E_{div}}{\partial L_L} = 0 \qquad \frac{\partial E_{div}}{\partial L_R} = 0$$

Because of the quadratic shape of E_{div} as a function of L_L and L_R, solving the above equations will surely give the unique global minimum of E_{div}. Solving the above equations will give us equations 4 and 5. By solving this linear system in L_L and L_R, we will have the formulas for calculating the optimum values of L_L and L_R at each fixed t and w:

$$L_L = \frac{\gamma.\delta - \beta.\theta}{\beta^2 - \alpha.\beta} \qquad L_R = \frac{\alpha.\theta - \beta.\delta}{\beta^2 - \alpha.\beta}$$

where α, β, γ, δ and θ are all sums computed from $\mu_T(p)$, $c(p)$, and $v(att(p))$:

$$\alpha = \sum_{p \in GS} \mu_T(p).\, v\big(att(p)\big)^2$$

$$\beta = \sum_{p \in GS} \mu_T(p).\, v\big(att(p)\big).\big[1 - v\big(att(p)\big)\big]$$

$$\gamma = \sum_{p \in GS} \mu_T(p).\big[1 - v\big(att(p)\big)\big]^2$$

$$\delta = -\sum_{p \in GS} \mu_T(p).\, c(p).\, v\big(att(p)\big)$$

$$\theta = -\sum_{p \in GS} \mu_T(p).\, c(p).\big[1 - v\big(att(p)\big)\big]$$

$$E_{div} = \sum_{p \in GS} \mu_T(p).\{c(p) - [v(att(p), t, w).\, L_L + (1 - v(att(p), t, w)).\, L_R]\}^2 \quad (3)$$

$$-2 \sum_{p \in GS} \mu_T(p).\, v\big(att(p)\big).\{\mu_T(p) - [v(att(p)).\, L_L + (1 - v(att(p))).\, L_R]\} = 0 \quad (4)$$

$$-2 \sum_{p \in GS} \mu_T(p).\big(1 - v\big(att(p)\big)\big).\{\mu_T(p) - [v(att(p)).\, L_L + (1 - v(att(p))).\, L_R]\} = 0 \quad (5)$$

Then, E_{div} is calculated using the temporary values of $, w,\ L_L,$ and L_R . The parameters that minimize E_{div} are selected for the discriminator function of this attribute.

4.2 Searching for t and w

We developed an algorithm to find the optimum values of t and w for each attribute. This algorithm sorts the cases based on their attribute values (and eliminates duplicate values), assigns to t the mean of attribute values of two patients in the dataset while assigning to w the difference between the attribute values of the same two patients. The algorithm repeats this process for combinations of all two patients attribute values, and calculates $t, w,\ L_L$, L_R, and E_{div} for each of them. When the two selected patient's attribute values are picked up from a single patient, the splitting would be crisp (with w equal to 0); thus the algorithm does not have any tendency for fuzzy splitting.

This algorithm performs better than simply searching the interval of minimum to maximum values of the attribute (the attribute range) with changing t and w in small increments. In fact, we first tried to search for t and w by changing t from the minimum to maximum in small increments (ε), and changing w from 0 to $\min(t - \min(att), \max(att) - t)$ in small increments for each t value. Because various attributes have different units of measurement, the value of ε could not be defined as a single value, and it had to be defined as a fraction of the attribute range. Most attributes had some very big values for some patients. For example, while CA125 attribute value is less than 500 for most patients, its value is more than 5000 for few patients. If we have defined the value of $\varepsilon = \text{range}(CA125)/100$, then the value of ε could be more than 50, making the searching algorithm inefficient. We additionally tried to eliminate the outlier attribute values by eliminating the attribute values out of the interval:

$$\text{mean}(att) \pm 2\ \text{SD}(att)$$

where $\text{SD}(att)$ denotes standard deviation of the attribute values. This approach did not assign enough small values to ε either, because the distribution of most attributes were left skewed, causing this approach for eliminating outliers to be inefficient.

On the other hand, the algorithm we developed extensively searches for t in high density areas of the attribute distribution while minimal searching is done over low density areas of the distribution.

The final parameters selected for the discriminator function of the attribute were the ones minimizing E_{div}.

4.3 Defining the Stopping Criteria

The stopping criteria used in the non-fuzzy, binary ID3 algorithm (Mitchell, 1997) should be generalized to include fuzzy membership of patients in each node. The stopping criteria in the ID3 algorithm include limited number of members in the node, sufficient purity of the node members, and consumption of all attributes already in the ancestor nodes so that no attribute is remained for further splitting.

The cardinality of a fuzzy set is defined as the sum of membership values of all members:

$$|S| = \sum_{p \in S} \mu_S(p)$$

For having a measure for the purity of node members, *Node Error* is defined as:

$$E_N = \sum_{p \in GS} \mu_N(p)\,.\,[c(p) - L_N]^2$$

where E_N denotes node error, $\mu_N(p)$ denotes patient's membership value in the node, and L_N denotes node label.

The third criterion of ID3 algorithm can be conceptualized as inability of the best selected attribute to further purify the successor children. When all attributes are used in the ancestor nodes, it means that they can be used again but they will not further purify the successor children. This concept is close to the concept of E_{div}: if E_{div} of the best selected attribute for the node is high, it means that even the best attribute cannot further purify the successor children.

Finally, we can define the generalized stopping criteria to include fuzzy concepts:

a. $|N| \leq T_{|N|}$
b. $E_N \leq T_{E_N}$
c. $E_{div} \geq T_{E_{div}}$

where $|N|$ denotes cardinality of the set of node members, $T_{|N|}$ denotes cardinality threshold, E_N denotes node error, T_{E_N} denotes node error threshold, E_{div} denotes division error, and $T_{E_{div}}$ denotes division error threshold.

By decreasing the values of $T_{|N|}$ and T_{E_N} and increasing the value of $T_{E_{div}}$, the resulted tree will be bigger and, presumably, more overfitted. Because the grown tree would be pruned later, stopping criteria should be tuned such that the tree would grow large enough without concerning about the overfitting. However, stopping criteria should not be set such that the growing process would need a plentiful amount of computational time for making an unnecessarily overgrown tree.

5 PRUNING METHOD

Objective: given a grown fuzzy decision tree (FDT) and a pruning set, find the subtree of FDT among all subtrees which can be generated from FDT that has the minimum mean absolute error (MAE) on the pruning set:

$$MAE = \frac{E_{prf}}{|PS|} = \frac{\sum_{p \in PS} [c(p) - \hat{\mu}_C(p)]^2}{|PS|}$$

A subtree of the FDT is made by *contracting* one or several test nodes of the FDT. Contracting a test nodes means replacing the test node with a leaf (Mingers, 1989). The estimated output class of the node is assigned the label of the node. Nodes labels are already calculated in the growing step. The number of subtrees which can be made from a tree by contracting its test nodes increases exponentially with the number of test nodes of the tree; thus contracting all test nodes of a given tree one by one, and testing the resulted trees on the pruning set one by one, takes a considerable amount of computational time. Therefore, we used a modified version of the *critical value pruning* method in the following three steps:

1- *Test Nodes Sorting* by increasing order of their importance. The importance of each test node is determined by the node error (E_N) calculated in the growing step and saved in each node. The more E_N of a test node, the less pure the test node, and thus the more important the test node for differentiation of output classes in the successor nodes. If a node in this list is placed in a more important position than any of its ancestors, the node is removed from the list because it would be pruned together with the pruning of that ancestor. In the list, the most important test node is invariably the root.

2- *Subtrees Sequence Generation*: the previous list gives the order of the critical values for pruning. In critical value pruning, a critical value for the importance of test nodes is determined, and test nodes which are less important than the critical value are pruned, unless one of the successor test nodes reach the critical value. The larger the critical value selected, the greater the degree of pruning, and the smaller the resulted tree. In practice, a sequence of pruned trees is generated using increasing critical values. The previous list gives the order of critical values by which the test nodes are contracted.

At the first step, the first node in the list is contracted, and the resultant tree is saved in a tree sequence. The process is continued by contracting the next nodes in the list one by one, and saving the resulted trees in the tree sequence. At the end, we will have a sequence of trees in decreasing order of complexity.

Before contracting the first test node in the list, the complete tree is tested on the pruning set to calculate its MAE. For doing that, the pruning set patients are propagated through the tree and the membership value of all patients in each node is calculated and saved. Then the output class of all patients estimated by the complete tree is calculated and saved. Afterwards, MAE of the complete tree is calculated.

Then, every time a test node T from the sorted list of the important nodes is contracted, the output class of all patients of the pruning set estimated by the new subtree is recursively updated by removing

from $\hat{\mu}_C(p)$ the pooled output estimated by the successor leaves of the contracted node, and then adding to $\hat{\mu}_C(p)$ the output estimated by the contracted node which has become a new leaf:

a. $\forall j \; below \; test \; node \; T, and \; \forall p \in PS$:

$$\hat{\mu}_C(p) = \hat{\mu}_C(p) - \mu_j(p).L_j$$

b. $\forall p \in PS$: $\hat{\mu}_C(p) = \hat{\mu}_C(p) + \mu_T(p).L_T$

Where T is the test node being contracted. The above process for updating $\hat{\mu}_C(p)$ of all patients eliminates the necessity of propagating all patients through the pruned subtree in each step of subtrees generation. Each step of subtrees sequence generation is completed by calculating MAE of the resulted pruned subtree on the pruning set. Subtrees sequence generation is finished when there are no more nodes in the sorted list candidate for contracting.

3- *Best subtree selection*: finally, the best subtree in the sequence is selected. By having MEA of all subtrees in the sequence, the smallest subtree having the least MAE is selected.

6 REFITTING METHOD

In the growing step, structure of the tree is built and labels are assigned to each node, both based on local optimization strategies. In the pruning step, structure of the tree is amended by contracting the overfitted test nodes. In the refitting step, leaves labels are amended based on global optimization strategies (Olaru and Louis, 2003).

Let us consider the following definitions:

$$[c] = c(p_i), \quad \forall p_i \in LS$$
$$[\pi] = \hat{\mu}_C(p_i), \quad \forall p_i \in LS$$
$$[L] = L_j, \quad \forall j \in leaves$$
$$[M_{ij}] = \mu_j(p_i), \quad \forall j \in leaves, \forall p_i \in LS$$

where j is the index of leaves in the tree, and i is the index of patients in the learning set. $[L]$, $[c]$, and $[\pi]$ are defined as column vectors, and $[M]$ is defined as a matrix of $m \times n$ dimension while m denotes number of patients in the learning set and n denotes number of leaves (presumably $m > n$). Then:

Objective: given the pruned fuzzy decision tree and the learning set as the set of examples for refitting, find amended fuzzy decision tree leaves labels $[L^*]$ so that E_{prf} is minimized:

$$E_{prf} = \sum_{p \in LS} [c(p) - \hat{\mu}_C(p)]^2 = \|[c] - [M].[L]\|^2$$

$$[L^*] = \arg\min_{[L]} \|[c] - [M].[L]\|^2$$

By solving this problem, we will have the amended leaves labels, $[L^*]$. The solution of above optimization problem is as follow.

$$[L^*] = \left[[M]^T.[M]\right]^{-1}.M^T.[c]$$

7 THE DATASET

A dataset of 305 patients collected for the International Ovarian Tumor Analysis (IOTA) study was used in this project. The IOTA study is a multicenter collaborative project for preoperative differentiation of ovarian tumors based on predictive models.

Patients assessed with transvaginal ultrasonography and found to have an apparent persistent extrauterine pelvic mass were included in the IOTA study. Before surgery, clinical, laboratory, and ultrasonographic data was recorded to be used as input attributes. Patients then underwent surgical resection of the mass. All surgically removed tissues were extensively sampled for histologic examination. The histologic classification of the removed tissue (benign or malignant) was recorded to be used as output, as follows:

$$c(p) = \begin{cases} 1, & pathology(p) = malignant \\ 0, & pathology(p) = benign \end{cases} \quad (6)$$

The dataset contained the following variables: patient's age (years), menopausal status (premenopausal versus postmenopausal), serum CA125 level (units/mL), 8 sonographic morphologic variables, 5 color Doppler variables, and pathology results classification (benign or malignant). Ultrasonographic examination was done and reported based on the standard methods already published by IOTA group (Timmerman et al., 2000).

8 RESULTS

The size of growing, pruning, and testing sets was 200, 55, and 50, respectively. The values of the stopping criteria were set as $T_{|N|}$ equal to 4, T_{E_N}

equal to 4, and $T_{E_{div}}$ equal to 30. Smaller values resulted in reaching the maximum recursion limit of MATLAB without returning any trees. Using above algorithm parameters, a decision tree with 27 nodes was made at the end of the growing step. Pruning resulted in elimination of 6 overfitted nodes. Finally, the pruned tree was refitted and tested.

For testing the fuzzy decision tree, the following steps were done:

1- Patients of the testing set were propagated through the tree, and membership values of all patients in all leaves were calculated and saved. Then the output class of all patients of the testing set estimated by the complete tree was calculated according to equation 1.

2- The performance error, E_{prf}, was calculated:

$$E_{prf} = \sum_{p \in TS} [c(p) - \hat{\mu}_C(p)]^2$$

The performance error is presumably increased by larger sizes of the testing set, and thus should be adjusted by the size of the testing set:

$$MAE = \frac{E_{prf}}{|TS|}$$

where $|TS|$ denotes cardinality of the testing set equaling the number of patients in the testing set, and MAE denotes mean absolute error.

3- A Receiver Operator Curve (ROC) was developed for analyzing the performance of the tree in terms of area under curve (AUC) of the ROC as well as finding the optimal cutoff value for the estimated output class.

The optimal point of the ROC was found by using the built-in function $perfcurve$ in MATLAB programming language. The function plots the true positive rate (TPR) versus false positive rate (FPR) while the resultant ROC is parameterized as a function of cutoff values:

$$[x, y] = \left(\text{FPR}(cutoff), \text{TPR}(cutoff)\right)$$

where x and y denote coordinates of each point of the plot. The optimal point of the ROC was found by moving a straight line with the slope of one from the upper left corner of the ROC (FPR=0, TPR=1) down and to the right until it intersects the ROC. Using the coordinates of this point (x=FPR, y=TPR), the optimum cutoff for the estimated output was then computed.

4- Using the coordinates of the optimum point of the ROC, sensitivity and specificity were calculated:

$$sensitivity = TPR \qquad specificity = 1 - FPR$$

Then, using sensitivity and specificity, the likelihood ratios for positive (malignant) and negative (benign) results of the tree were calculated:

$$LR^{+} = \frac{sensitivity}{1 - specificity} \qquad LR^{-} = \frac{1 - sensitivity}{specificity}$$

The results of the final fuzzy decision tree testing on the testing set are summarized in table 1. To ensure that the small size of the testing set had not biased the testing results, all growing, pruning, and refitting steps were repeated for 10 times, allocating all patients to GS, PS, and TS again each time, and the resulted trees were compared. Except for 2 times

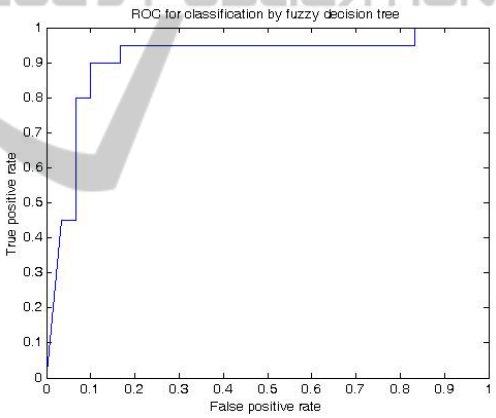

Figure 4: ROC for classification of cases in the testing set by fuzzy decision tree.

where the resulted trees were different in some bottom nodes, the resulted trees were almost the same in structure and leaves labels. Testing the resulted trees on their corresponding testing sets resulted in AUCs ranged from 0.89 to 0.96. The ROC plot is shown in figure 4.

9 DISCUSSIONS

The performance measures of the final resulted tree were acceptable for clinical utilization. The positive likelihood ratio of near 10 as well as the negative likelihood ratio of near 0.1 is indicative of the high

accuracy of the resulted fuzzy decision tree. More importantly, the tree had the capacity of being translated into the equivalent set of fuzzy if-then rules; each rule is made by conjunction of all conditions from the root to each leaf. Because the decision tree had 11 leaves, the decision tree was translated into 11 fuzzy if-then rules. Then the rules were interpreted and criticized by domain experts. This process of interpretation and amendment of the rules by clinicians is the main advantage of this method. A sample of rules extracted from the tree is:

Table 1: Decision tree testing results.

Performance measure	Value
Mean squared error	0.1195
AUC of ROC curve	0.9092
Output class cutoff	0.3596
Positive likelihood ratio	9.0000
Negative likelihood ratio	0.1111
Sensitivity	90.00%
Specificity	90.00%

"If patient's CA125 level is *low*, and the lesion internal wall is not smooth, and color score is *low*, and the cyst content is hemorrhagic, then the lesion is *benign*."
where italic words are linguistic fuzzy variables defined over continuous attributes. The comment of a domain expert on the above rule was:
"This rule is right, because a hemorrhagic cyst having low blood flow (low color score) and low CA125 level would be a hemorrhagic functional ovarian cyst which is a benign lesion."
Likewise, all rules were interpreted by clinicians, and some rules were amended by clinicians based on clinical knowledge.

9.1 Inductive Bias

An approximation to the inductive bias of decision tree learning is (Mitchell, 1997):
"Smaller trees are preferred over larger trees. Trees that place highly purifying attributes closer to the root are preferred over those that do not."
While selecting attribute for a test node, decision tree learning algorithms can just think of the immediate consequences of this selection, but cannot think of further consequences in successor nodes. Additionally, when the learning algorithm faces the consequences of the bad selections in the ancestor nodes (such as facing nodes which cannot be further purified using any attribute), it never backtracks to reconsider its previous choices. Therefore, these learning algorithms are susceptible to the usual drawback of simple-to-complex searching for hypotheses without backtracking: selecting locally optimal solutions which are not globally optimal.
All backfitting algorithms designed for decision tree learning can just tune parameters of decision trees, but cannot amend their structure. Pruning algorithms are just able to contract overfitted nodes, but are not able to reconsider various generations of nodes to find the globally optimal decision tree. In fact, amending the structure of decision trees by simultaneously considering various generations of nodes for finding a globally optimal decision tree is too complex to be done by simple algorithms.
While inductive bias of decision tree learning cannot be easily solved by simple algorithms, *it can be solved by using the aid of human experts to amend the tree or its equivalent rules*. Currently, the complex thinking process of a clinician cannot be simulated by any algorithm. The main advantage of decision trees is their explicit and easy-to-understand nature, as well as their ability to be translated into the equivalent if-then rules. The whole point was making a decision tree by an artificial learner, because of the perfect abilities of artificial learners in analyzing high-dimensional data; and then amending the built tree (or its equivalent rules) by human experts, because of the perfect abilities of the human brain in interpreting and criticizing rules.

10 CONCLUSIONS

Decision trees are easy-to-interpret for clinicians, and fuzzy reasoning is a more general approach for managing uncertainty than probability theory. We proposed that a combination of decision trees and fuzzy reasoning would result in a robust and accurate classification method.
The performance results of the tree are acceptable, with positive likelihood ratio of near 10 and negative likelihood ratio of near 0.1 for diagnosing malignancy. This model has minimal restriction bias, the problem of overfitting is eliminated in the pruning step, and the problem of model preference bias was minimized by getting aid from human experts to amend the extracted rules.
Eleven fuzzy if-then rules were extracted from the tree and were interpreted and amended by clinicians. These rules are ready to be used in clinical practice guidelines as well as being implemented into some expert system for management of patients with adnexal mass.

REFERENCES

Hoffman, M. S. 2009. Overview of the evaluation and management of adnexal masses. *In:* mann, W. J. & goff, B. (eds.) *Uptodate.* 17.3 ed. Waltham: uptodate inc.

Mann, W. J., chalas, e. & Valea, F. A. 2009. Epithelial ovarian cancer: initial surgical management. *In:* goff, b. (ed.) *Uptodate.* 17.3 ed. Waltham: uptodate inc.

Mingers, J. 1989. An empirical comparison of pruning methods for decision tree induction. *Mach learn,* 4**,** 227-43.

Mitchell, T. M. 1997. Decision tree learning. *In:* Mitchell, T. M. (ed.) *Machine learning.* 1st ed. Columbus: mcgraw-hill.

Myers, E. R., bastian, L. A., Havrilesky, L. J., Kulasingam, S. L., Terplan, M. S., Cline, K. E., Gray, R. N. & Mccrory, D. C. Management of Adnexal Mass. Evidence report/technology assessment no.130 (prepared by the duke evidence-based practice center under contract no. 290-02-0025.) Ahrq publication no. 06-e004. Rockville, md: agency for healthcare research and quality. Feb 2006.

Olaru, C. & Louis, W. 2003. A complete fuzzy decision tree technique. *Fuzzy set syst,* 138**,** 221-54.

Schaffer, J. I. 2008. Epithelial ovarian cancer. *In:* schorge, J. O., schaffer, J. I., halvorson, L. M., hoffman, B. L., bradshaw, K. D. & cunningham, F. G. (eds.) *Williams gynecology.* 1st ed. Dallas: mcgraw-hill.

Timmerman, D., Valentin, L., Bourne, T. H., collins, W. P., Verrelst, H. & Vergote, I. 2000. Terms, definitions and measurements to describe the sonographic features of adnexal tumors: a consensus opinion from the international ovarian tumor analysis (iota) group. *Ultrasound obstet gynecol,* 16**,** 500-5.