Journal of Medical Systems, Vol. 26, No. 5, October 2002 (© 2002)

Decision Trees: An Overview and Their Use in Medicine

Vili Podgorelec,[1,2] Peter Kokol,[1] Bruno Stiglic,[1] and Ivan Rozman[1]

In medical decision making (classification, diagnosing, etc.) there are many situations where decision must be made effectively and reliably. Conceptual simple decision making models with the possibility of automatic learning are the most appropriate for performing such tasks. Decision trees are a reliable and effective decision making technique that provide high classification accuracy with a simple representation of gathered knowledge and they have been used in different areas of medical decision making. In the paper we present the basic characteristics of decision trees and the successful alternatives to the traditional induction approach with the emphasis on existing and possible future applications in medicine.

KEY WORDS: decision trees; classification; decision making; machine learning.

INTRODUCTION

Making the right decision is becoming the key factor for the successful achievement of our goals in all areas of our work. The ways of finding the right decision are as many as the number of people who have to make them. Nevertheless, the basic idea is the same for many of them: a decision is usually made as a combination of experiences from solving similar cases, the results of recent researches and personal judgment. The number of solved cases and new researches is increasing rapidly. It could be expected that newly made decisions will become better and more reliable but for the individuals and groups who have to make decisions it is actually becoming more and more complicated, because they simply can not process the huge amounts of data anymore. And there the need for a good decision support technique arises. It should be able to process those huge amounts of data and to help experts to make their decisions easier and more reliably. For this purpose, it is equally or even more important as suggesting the possible decision, to provide also an explanation of how and why the suggested decision was chosen. In this manner, an expert can decide whether the suggested solution is appropriate or not.

[1]University of Maribor – FERI, Smetanova 17, SI-2000 Maribor, Slovenia.
[2]To whom correspondence should be addressed; e-mail: vili.podgorelec@uni-mb.si.

0148-5598/02/1000-0445/0 © 2002 Plenum Publishing Corporation

As in many other areas, decisions play an important role also in medicine, especially in medical diagnostic processes. Decision support systems helping physicians are becoming a very important part in medical decision making, particularly in those situations where decision must be made effectively and reliably. Since conceptual simple decision making models with the possibility of automatic learning should be considered for performing such tasks, decision trees are a very suitable candidate. They have been already successfully used for many decision making purposes.

Objective and Scope of the Paper

In this paper, we present an overview of decision trees with the emphasis on a variety of different induction methods available nowadays. Induction algorithms ranging from the traditional heuristic based techniques to the most recent hybrids, such as evolutionary and neural network based approaches, are described. Basic features, advantages and drawbacks of each method are presented, biasing to the medical domain. For the readers not very familiar with the field of decision trees this paper should be a good introduction into this topic, whereas for more experienced readers it should broaden their perspective and knowledge.

DECISION TREES

Inductive inference is the process of moving from concrete examples to general models, where the goal is to learn how to classify objects by analyzing a set of instances (already solved cases) whose classes are known. Instances are typically represented as attribute-value vectors. Learning input consists of a set of such vectors, each belonging to a known class, and the output consists of a mapping from attribute values to classes. This mapping should accurately classify both the given instances and other unseen instances.

A decision tree[1] is a formalism for expressing such mappings and consists of tests or attribute nodes linked to two or more subtrees and leafs or decision nodes labeled with a class which means the decision. A test node computes some outcome based on the attribute values of an instance, where each possible outcome is associated with one of the subtrees. An instance is classified by starting at the root node of the tree. If this node is a test, the outcome for the instance is determined and the process continues using the appropriate subtree. When a leaf is eventually encountered, its label gives the predicted class of the instance.

The finding of a solution with the help of decision trees starts by preparing a set of solved cases. The whole set is then divided into (1) a training set, which is used for the induction of a decision tree and (2) a testing set, which is used to check the accuracy of an obtained solution. First, all attributes defining each case are described (input data) and among them one attribute is selected that represents a decision for the given problem (output data). For all input attributes specific value classes are defined. If an attribute can take only one of a few discrete values then each value takes its own class; if an attribute can take various numeric values then some characteristic intervals must be defined, which represent different classes. Each attribute can represent one internal node in a generated decision tree, also called an attribute node or a test node (Fig. 1). Such an attribute node has exactly as many

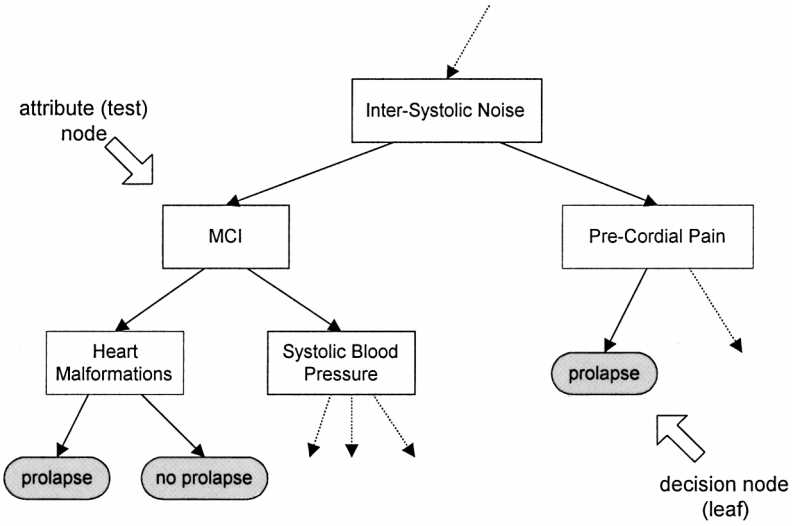

Fig. 1. An example of a (part of a) decision tree.

branches as its number of different value classes. The leaves of a decision tree are decisions and represent the value classes of the decision attribute – decision classes (Fig. 1). When a decision has to be made for an unsolved case, we start with the root node of the decision tree and moving along attribute nodes select branches where values of the appropriate attributes in the unsolved case matches the attribute values in the decision tree until the leaf node is reached representing the decision. Usually the members of a set of objects are classified as either positive or negative instances (for example ill and healthy patients), for the generality purpose this approach has to be extended with multiclass decision making, enabling one to differentiate between various decision classes (to determine patients as low, medium and high for example). An example of a simple decision tree can be seen on Fig. 1.

The decision tree is very easy to interpret. For example, from the tree shown in Fig. 1 we can deduce the following two rules:

1. if the patient has intersystolic noise and MCI and heart malformations then she/he has a prolaps, and
2. if the patient has intersystolic noise and MCI and no heart malformations then she/he does not have a prolaps.

A decision tree can be built from a set of training objects with the "divide and conquer" principle. When all objects are of the same decision class (the value of the output attribute is the same) then a tree consists of a single node—a leaf with the appropriate decision. Otherwise an attribute is selected which value is of at least two different decision classes and a set of objects is divided according to the category of the selected attribute. The selected attribute builds an attribute (test) node in a growing decision tree, for each branch from that node the inducing procedure is repeated upon the remaining objects regarding the division until a leaf (a decision) is encountered.

From a geometric point of view a set of x attributes defines a x-dimensional space, where each data object represents a point. A division of data objects regarding the attribute's class suits the definition of decision planes in the same space. Those planes are hyperplanes which are orthogonal to the selected attribute—a decision tree divides a search space into hyperrectangles, each of them represents one of the possible decisions; of course more rectangles can also represent the same decision.

CLASSICAL INDUCTION OF DECISION TREES

In 1986, Quinlan introduced an algorithm for inducing decision trees ID3.[2,3] In 1993 ID3 was upgraded with an improved algorithm C4.5[1] that is still regarded as the reference model to build a decision tree based on the traditional statistical approach. Both algorithms ID3 and C4.5 use the statistical calculation of information gain from a single attribute to build a decision tree. In this manner an attribute that adds the most information about the decision upon a training set is selected first, the next one is selected the most informative from the remaining attributes, etc.

The method for constructing a decision tree as paraphrased from Quinlan,[1] is as follows:

If there are k classes denoted $\{C_1, C_2, \ldots, C_k\}$, and a training set, T, then

- if T contains one or more objects which all belong to a single class C_j, then the decision tree is a leaf identifying class C_j.
- if T contains no objects, the decision tree is a leaf determined from information other than T.
- if T contains objects that belong to a mixture of classes, then a test is chosen, based on a single attribute, that has one or more mutually exclusive outcomes $\{O_1, O_2, \ldots, O_n\}$. T is partitioned into subsets $T_1, T_2, \ldots, T_n$, where T_i contains all the objects in T that have outcome O_i of the chosen test. The same method is applied recursively to each subset of training objects.

The most important aspect of a traditional decision trees induction strategy is the way in which a set is split, i.e. how to select an attribute test that determines the distribution of training objects into subsets upon which subtrees are built consequently. The C4.5 induction algorithm uses information theory[4] to evaluate splits. Two splitting criteria are implemented: gain criterion and gain ratio criterion.

The gain criterion[1] is developed in the following way: For any subset S of X, where X is the population, let freq(j_i, S) be the number of objects in S which belong to class i. Then consider the "message" that a randomly selected object belongs to class j_i. The "message" has probability freq(j_i, S)/$|S|$, where $|S|$ is the total number of objects in subset S. The information conveyed by the message (in bits) is given by

$$-\log_2(\text{freq}(j_i, S)/|S|).$$

Summing over the classes gives the expected information (in bits) from such a message:

$$\text{info}(S) = -\log_2\left(\frac{\text{freq}(C_j, S)}{|S|}\right).$$

When applied to a set of training objects, info(T) gives the average amount of information needed to identify the object of a class in T. This amount is also known as the entropy of the set T.

Consider a similar measurement after T has been partitioned in accordance with the n outcomes of a test X. The expected information requirement can be found as a weighted sum over the subsets $\{T_i\}$:

$$\text{info}_X(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \cdot \text{info}(T_i).$$

The quantity

$$\text{gain}(X) = \text{info}(T) - \text{info}_X(T)$$

measures the information that is gained by partitioning T in accordance with the test X. *The gain criterion*[1] selects a test to maximize this information gain.

The gain criterion has one significant disadvantage in that it is biased towards tests with many outcomes. The *gain ratio criterion*[1] was developed to avoid this bias. The information generated by dividing T into n subsets is given by

$$\text{split info}(X) = \pm \sum_{i=1}^{n} \frac{|T_i|}{|T|} \cdot \log_2 \left(\frac{|T_i|}{|T|} \right).$$

The proportion of information generated by the split that is useful for classification is

$$\text{gain ratio}(X) = \text{gain}(X)/\text{split info}(X).$$

If the split is near trivial, split information will be small and this ratio will be unstable. Hence, the *gain ratio criterion* selects a test to maximize the gain ratio subject to the constraint that the information gain is large.

This compares with CART's *impurity function* approach,[5] where impurity is a measure of the class mix of a subset and splits are chosen so that the decrease in impurity is maximized. This approach led to the development of the *gini index*.[5] The impurity function approach considers the probability of misclassifying a new sample from the overall population, given that the sample was not part of the training sample, T. This probability is called the *misclassification rate* and is estimated using either the *resubstitution estimate* (or training set accuracy), or the *test sample estimate* (test set accuracy). The node assignment rule selects i to minimize this misclassification rate. In addition, the *gini index* promotes splits that minimize the overall size of the tree.

Tests on Continuous Attributes

The algorithm for finding appropriate thresholds for continuous attributes[1,5,6] is as follows:

> The training objects are sorted on the values of the attribute. Denote them in order as $\{v_1, v_2, \ldots, v_m\}$. Any threshold value lying between v_i and v_{i+1} will have the same effect, so there are only m-1 possible splits, all of which are examined.

Some approaches for building decision trees use discretization of continuous attributes. Two ways are used to select discretized classes:

–equidistant intervals; the number of classes is selected first and then successive equidistant intervals are determined between absolute lower and upper bounds, and
–percentiles; again the number of classes is selected first and then successive intervals are determined based on the values of the appropriate attribute in the training set so that all intervals contain the same number of training objects.

Dynamic Discretization of Attributes

In MtDeciT 2.0 tool authors implemented an algorithm for finding subintervals,[7] where the distribution of training objects is considered and there are more than two subintervals possible. The approach is called dynamic discretization of continuous attributes, since the subintervals are determined dynamically during the process of building a decision tree. This technique first splits the interval into many subintervals, so that every training object's value has its own subinterval. In the second step it merges together smaller subintervals that are labeled with the same outcome into larger subintervals. In each of the following steps three subintervals are merged together: two "stronger" subintervals with one "weak" interval, where "weak" interval lies between those two "strong" subintervals. Here strong and weak applies to number of training objects in the subinterval tree (Fig. 2). In comparison to the previous two approaches the dynamic discretization

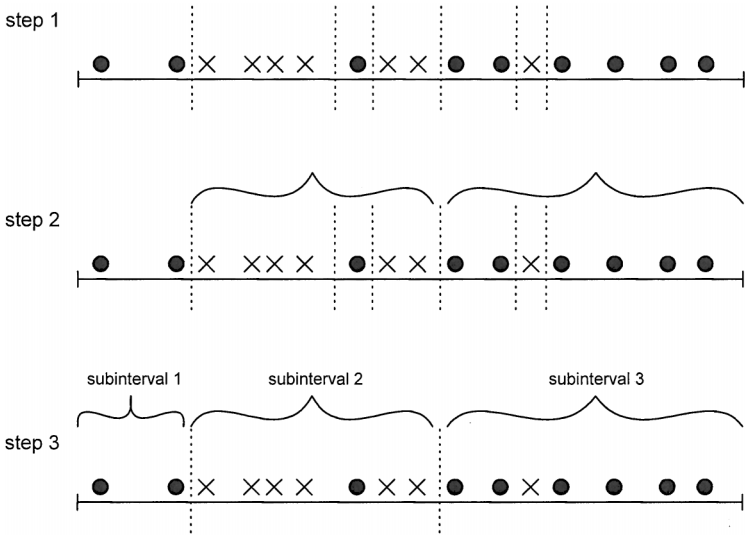

Fig. 2. Dynamic discretization of continuous attribute, which has values between 60 and 100. Shapes represent different attribute's values.

returns more natural subintervals, which result in better and smaller decision trees.

In general we differentiate between two types of dynamic discretization: general dynamic discretization and nodal dynamic discretization.

General dynamic discretization uses all available training objects for the definition of subintervals. That is why the general dynamic discretization is performed before the start of building a decision tree. All the subintervals of all attributes are memorized in order to be used later in the process of building of the decision tree.

Nodal dynamic discretization performs the definition of subintervals for all continuous attributes which are available in the current node of a decision tree. Only those training objects that came in the current node are used for setting the subintervals of the continuous attributes.

In a series of tests the authors showed that nodal dynamic discretization produces smaller decision trees with higher accuracy than decision trees built with general dynamic discretization.[7,8] Nodal dynamic discretization also outperforms classical discretization techniques in majority of cases.

Oblique Partitioning of Search Space

The so far presented algorithms are using univariate partitioning methods, which are attractive because they are straightforward to implement (since only one feature is analyzed at a time) and the derived decision tree is relatively easy to understand. Beside univariate partitioning methods there are also some successful partitioning methods which do not partition the search space axis-parallel based on only one attribute at a time but are forming oblique partition boundaries based on a combination of attributes (Fig. 3).

Oblique partitioning provides a viable alternative to univariate methods. Unlike their univariate counterparts, oblique partitions are formed by combinations of

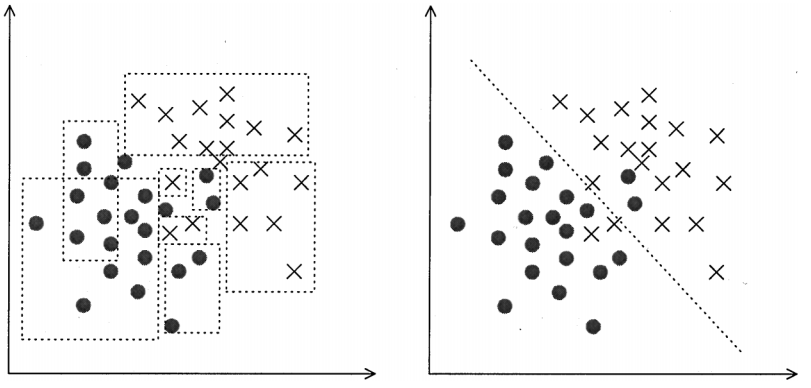

Fig. 3. Given axes that show the attribute values and shape corresponding to class labels (i) axis-parallel and (ii) oblique decision boundaries.

attributes. The general form of an oblique partition is given by

$$\sum_{i=1}^{d} \beta_i x_i \leq C$$

where β_i represents the coefficient of the i-th attribute. Because of their multivariate nature, oblique methods offer far more flexibility in partitioning the search space; this flexibility comes at a price of higher complexity, however. Consider that given a data set containing n objects described with d attributes, there can be $2\sum_{i=0}^{d}\binom{n-1}{i}$ oblique splits if $n > d$[9]; each split is a hyperplane that divides the search space into two nonoverlapping halves. For univariate splits, the number of potential partitions is much lower, but still significant, n.d.[10] In short, finding the right oblique partition is a difficult task.

Given the size of the search space, choosing the right search method is of critical importance in finding good partitions. Perhaps the most comprehensive reference[5] on this subject is on classification and regression trees (CART). Globally, CART uses the same basic algorithm as Quinlan in C4.5. At the decision node level, however, the algorithm becomes extremely complex. CART starts out with the best univariate split. It then iteratively searches for perturbations in attribute values (one attribute at a time) which maximize some goodness metric. At the end of the procedure, the best oblique and axis-parallel splits found are compared and the better of these is selected.

Although CART provides a powerful and efficient solution to a very difficult problem, it is not without its disadvantages. Because the algorithm is fully deterministic, it has no inherent mechanism for escaping from local optima. As a result, CART has a tendency to terminate its partition search at a given node too early. The most fundamental disadvantage of CART (and of traditional approach of inducing decision trees in general) is that the decision tree induction process can cause the metrics to produce misleading results. Because traditional decision tree induction algorithms choose what is locally optimal for each decision node, they inevitably ignore splits which score poorly alone, but yield better solution when used in combination. This problem is illustrated by Fig. 4. The solid lines indicate the splits found by CART. Although each split optimizes the impurity metric, the end product clearly does not reflect the best possible partitions (indicated by the dotted lines). However, when evaluated as individuals, the dotted lines register high impurities and are therefore not chosen. Given this, it is apparent that the sequential nature of decision trees can prevent the induction of trees that reflect the natural structure of the data.

Pruning the Decision Tree

Decision tree classifiers aim to refine the training sample T into subsets which have only a single class. However, training samples may not be representative of the population they are intended to represent. In most cases, fitting a decision tree until all leaves contain data for a single class causes *overfitting*. That is, the decision tree is designed to classify the training sample rather than the overall population and accuracy on the overall population will be much lower than the accuracy on the training sample.

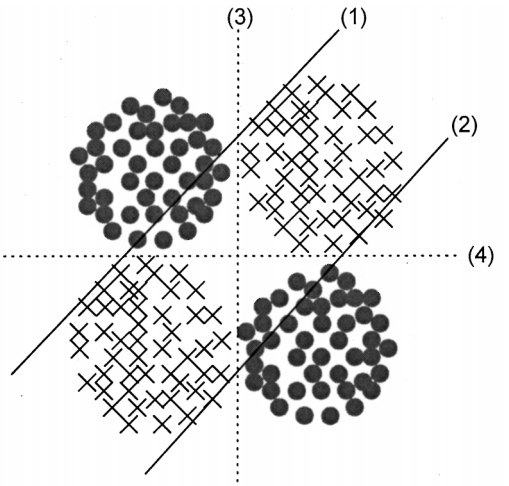

Fig. 4. CART generated splits (solid lines 1 and 2) minimize impurity at each decision node in a way that is not necessarily optimal regarding the natural structure of data (denoted by dotted lines 3 and 4).

For this purpose most of the decision tree induction algorithms (C4.5, CART, OC1) use pruning. They all grow trees to maximum size, where each leaf contains single-class data or no test offers any improvement on the mix of classes at that leaf, and then prune the tree to avoid overfitting. Pruning occurs within C4.5 when the predicted error rate is reduced by replacing a branch with a leaf. CART and OC1 use a proportion of the training sample to prune the tree. The tree is trained on the remainder of the training sample and then pruned until the accuracy on the pruning sample can not be further improved.

Variant Methods

Although algorithms such as ID3, C4.5, and CART make up the foundation of traditional decision tree induction practice, there is always room for improvement of the accuracy, size, and generalization ability of the generated trees. As would be expected, many researchers have tried to build on the success of these techniques by developing better variations of them.

Split Selection Using Random Search

Since random search techniques have proven extremely useful in finding solutions to nondeterministic polynomial complete (NP-complete) problems,[11] naturally they have been applied to decision tree induction. Heath[12,13] developed a decision tree induction algorithm called SADT which uses a simulated annealing process to find oblique splits at each decision node. Simulated annealing is a variation of hill climbing which, at the beginning of the process, allows some random downhill moves to be made.[14] As a computational process, simulated annealing is

patterned after the physical process of annealing,[15] in which metals are melted (at high temperatures) and then gradually cool until some solid state is reached.

Starting with an initial hyperplane, SADT randomly perturbs the current solution and determines the goodness-of-the-split by measuring the change in impurity (ΔE). If ΔE is negative (i.e. impurity decreases), the new hyperplane becomes the current solution; otherwise, the new hyperplane becomes the current split with probability $e^{-(\Delta E/T)}$ where T is the temperature of the system. Because simulated annealing mimics the cooling of metal, its initially high temperature falls with each perturbation. At the start of the process, the probability of replacing the current hyperplane is nearly 1. As the temperature cools, it becomes increasingly unlikely that worse solutions are accepted. When processing a given data set, Heath typically grows hundreds of trees, performing several thousands perturbations per decision node. Thus, while SADT has been shown to find smaller trees than CART, it is very expensive from a computational standpoint.[13]

A more elegant variation on Heath's approach is the OC1 system.[10] Like SADT, OC1 uses random search to find the best split at each decision node. The key difference is that OC1 rejects the brute force approach of SADT, using random search only to improve on an existing solution. In particular, it first finds a good split using a CART-like deterministic search routine. OC1 then randomly perturbs this hyperplane in order to decrease its impurity. This step is a way of escaping the local optima in which deterministic search techniques can be trapped. If the perturbation results in a better split, OC1 resumes the deterministic search on the new hyperplane; if not, it reperturbs the partition a user-selectable number of times. When the current solution can be improved no further, it is stored for later reference. This procedure is repeated a fixed number of times (using a different initial hyperplane in each trial). When all trials have been completed, the best split found is incorporated into the decision node.

Incremental Decision Tree Induction

The decision tree induction algorithms discussed so far grow trees from a complete training set. For serial learning tasks, however, training instances may arrive in a stream over a given time period. In these situations, it may be necessary to continually update the tree in response to the newly acquired data. Rather than building a new decision tree from scratch, the incremental decision tree induction approach revises the existing tree to be consistent with each new training instance. Utgoff implemented an incremental version of ID3, called ID5R.[16] ID5R uses an *E-Score* criteria to estimate the amount of ambiguity in classifying instances that would result from placing a given attribute as a test in a decision node. Whenever the addition of new training instances does not fit the existing tree, the tree is recursively restructured such that attributes with the lowest E-Scores are moved higher in the tree hierarchy. In general, Utgoff's algorithm yields smaller trees compared to methods like ID3, which batch process all training data. Techniques similar to ID5R include an incremental version of CART.[17] Incremental decision tree induction techniques result in frequent tree restructuring when the amount of training data is small, with the tree structure maturing as the data pool becomes larger.

Decision Forests

Regardless of the decision tree induction method utilized, subtle differences in the composition of the training set can produce significant variances in classification accuracy. This problem is especially acute when cross validating small data sets with high dimensionality.[18] Researchers have reduced these high levels of variance by using decision forests, composed of multiple trees (rather than just one). Each tree in a forest is unique because it is grown from a different subset of the same data set. For example, Quinlan's windowing technique[1] induces multiple trees, each from a randomly selected subset of the training data (i.e., a window). Another approach was devised by Ho,[19] who based each tree on a unique feature subset. Once a forest exists, the results from each tree must be combined to classify a given data instance. Such committee-type schemes for accomplishing this range from using majority rules voting[13,20] to different statistical methods.[21]

Deficiencies of Classical Induction

Decision trees have shown to be a powerful tool for decision support on different areas. Their effectiveness and accuracy of classification have been a surprise for many experts and their greatest advantage is in simultaneous suggestion of a decision and the straightforward and intuitive explanation of how the decision was made. Nevertheless, the classical induction approach also contains several deficiencies.

One of the most obvious drawback of classical decision tree induction algorithms is poor processing of incomplete, noisy data. If some attribute value is missing, classical algorithms do not perform well on processing of such object. For example, in Quinlan's algorithms before C4.5 such data objects with missing data have been left out of the training set—this fact of course resulted in decreased quality of obtained solutions (in this way the training set size and successively the information about the problem's domain have been reduced). Algorithm C4.5 introduced the technique to overcome this problem, but is still not very effective. In real world problems, especially in medicine, missing data is very usual; therefore, effective processing of such data is of vital importance.

The next important drawback of classical induction methods is the fact, that they can produce only one decision tree for a problem (when the same training set is used). In many real world situations it would be of a great benefit if more decision trees would be available and a user could choose the most appropriate one for a single case. As it is possible for training objects to miss some attribute value, the same goes also for a testing object—there can be a new case where some data is missing and is not possible to obtain it (unavailability of some medical equipment, for example, or invasiveness of a specific test for a patient). In this way another decision tree could be chosen that does not include a specific attribute test to make a decision.

Let us mention only one more disadvantage of classical induction methods, namely the importance of different errors. Between all decisions possible there are usually some more important than the others. Therefore, a goal is to build a decision tree is such a way that the accuracy of classification for those most important decisions is maximized. Once again, this problem is not solved adequately in classical decision tree induction methods.

ALTERNATIVE INDUCTION METHODS

Decision trees induction is a complex process. Therefore, deterministic induction approach is not necessarily optimal regarding the quality of obtained decision trees and includes several deficiencies. In last few years a lot of different alternative methods to the induction of decision trees have been introduced by various researchers, which try to overcome those deficiencies. Most of them are based on so-called soft methods, like evolutionary techniques or neural networks, sometimes several methods are combined in a hybrid algorithm.

A vast number of techniques have been also developed which help to improve only a part of the decision tree induction process. Such techniques include evolutionary algorithms for optimizing split functions in attribute nodes, dynamic discretization of attributes, dynamic subset selection of training objects, etc.

A Combination of Decision Trees and Neural Networks

When decision trees and neural networks are compared, one can see that their advantages and drawbacks are almost complementary. For instance knowledge representation of decision trees is easily understood by humans, which is not the case for neural networks; decision trees have trouble dealing with noise in training data, which is again not the case for neural networks; decision trees learn very fast and neural networks learn relatively slow, etc. Therefore, the idea is to combine decision trees and neural networks in order to combine their advantages. In this manner, different hybrid approaches from both fields have been introduced.[8,22,23]

Zorman in his MtDecit 2.0 approach first builds a decision tree that is then used to initialize a neural network.[8,24] Such a network is then trained using the same training objects as the decision tree. After that the neural network is again converted to a decision tree, that is better than the original decision tree.[25] The source decision tree is converted to a disjunctive normal form—a set of normalized rules. Then the disjunctive normal form serves as source for determining the neural network's topology and weights. The neural network has two hidden layers, the number of neutrons on each hidden layer depends on rules in the disjunctive normal form. The number of neurons in the output layer depends on how many outcomes are possible in the training set. After the transformation, the neural networks is trained using back-propagation. Mean square error of such network converges toward zero much faster than it would in the case of randomly set weights in the network. Finally, the trained neural network has to be converted into a final decision tree. The neural network is examined in order to determine the most important attributes that influence the outcomes of the neural network. A list containing the most important attributes is then used to build the final decision tree, that in majority of cases performs better than the source decision tree. The last conversion usually causes a loss of some knowledge contained in the neural network, but even so most knowledge is transformed into the final decision tree. If the approach is successful, then the final decision tree has better classification capabilities than the source decision tree.

Using Evolutionary Algorithms to Build Decision Trees

Evolutionary algorithms are generally used for very complex optimization tasks,[26] for which no efficient heuristic method exist. Construction of decision trees is a complex task, but heuristic methods exist which usually work efficiently and reliably. Nevertheless, there are some reasons justifying an evolutionary approach. Because of the robustness of evolutionary techniques they can be successfully used also on incomplete, noisy data (which often happens in medicine because of measurement errors, unavailability of proper instruments, risk to patients, etc.). Because of evolutionary principles used to evolve solutions, solutions can be found which can be easily overlooked otherwise. Also the possibility of optimizing the decision tree's topology and the adaptation of split tresholds is an advantage.

There are several attempts to build a decision tree with the use of evolutionary techniques,[27,28,29] one the most recent one and also very successful in medical applications is genTrees, developed by Podgorelec and Kokol.[20,28,30] First an initial population of (about one hundred) semirandom decision trees is seeded. A random decision tree is built by randomly choosing attributes and defining a split function. When a preselected number of attribute nodes is placed into a growing tree then a tree is finalized with decision nodes, which are defined by choosing the most fit decision in each node regarding the training set. After the initialization the population of decision trees evolves through many generations of selection, crossover and mutation in order to optimize the fitness function that determines the quality of evolved decision tree:

$$\mathrm{LFF} = \sum_{i=1}^{K} w_i \cdot (1 - \mathrm{acc}_i) + \sum_{i=1}^{N} c(t_i) + w_{\mathrm{u}} \cdot \mathrm{nu},$$

where

- K is the number of decision classes,
- N in the number of attribute nodes in a tree,
- acc_i is the accuracy of classification of objects of a specific decision class d_i,
- w_i is the importance weight for classifying the objects of a decision class d_i,
- $c(t_i)$ is the cost of using the attribute in a node t_i,
- nu is the number of unused decision (leaf) nodes, i.e. where no object from the training set fall into, and
- w_{u} is the weight of the presence of unused decision nodes in a tree.

According to LFF the best trees (the most fit ones) have the lowest function values—the aim of the evolutionary process is to minimize the value of LFF for the best tree. With the combination of selection that prioritizes better solutions, crossover that works as a constructive operator towards local optimums, and mutation that works as a destructive operator in order to keep the needed genetic diversity, the searching for the solution tends to be directed toward the global optimal solution. The global optimal solution is the most appropriate decision tree regarding the specific needs (expressed in the form of LFF). As the evolution repeats, more qualitative solutions are obtained regarding the chosen fitness function. The evolution stops when an optimal or at least an acceptable solution is found.

One step further from Podgorelec's approach has introduced Sprogar with his evolutionary vector decision trees – VEDEC.[31] In his approach the evolution of decision trees is similar to the one by Podgorelec, but the functionality of decision trees is enhanced in the way that not only one possible decision is suggested in the leaf, but several possible questions are answered with a vector of decisions. In this manner, for example, a possible treatment for a patient is suggested together with the diagnosis, or several diagnosis are suggested at the same time, which is not possible in ordinary decision trees.

The most recent approach to the evolutionary induction of decision trees-like methods is the AREX algorithm developed by Podgorelec and Kokol.[32] In this approach decision trees are extended by so-called decision programs which are evolved with the help of automatic programming. In this way a classical attribute test can be replaced by a simple computer program, which greatly improves the flexibility, at the cost of computational resources, however. The approach introduces a multilevel classification model, based on the difference between objects. In this manner "simple" objects are classified on the first level with simple rules, more "complex" objects are classified on the second level with more complex rules, etc.

APPLICATIONS IN MEDICINE

Decision trees are often used in medical and health care applications for more than 20 years. In this section we present an overview of some interesting recent applications of the use of decision trees in various medical fields reported on important international conferences and in international journals.

In more general articles, Cremilleux and Robert present the general framework for using decision trees in medicine.[33] Kokol *et al.* in their paper show certain limitation of decision trees in medical domain.[34] Zorman *et al.* evaluate different decision trees induction strategies on a hard real world problems of the orthopaedic fracture data with 2637 cases.[7] They tested various methods for building univariate decision trees in order to find the best induction strategy. They tested four classical approaches, one hybrid approach (neural networks and decision trees) and an evolutionary approach. The results show that all approaches had problems with either accuracy, sensitivity or decision tree size. The comparison shows that the best compromise in the hard real world problem decision trees building is the evolutionary approach.

In more specific papers, Tsien *et al.* show that decision trees can support early and accurate diagnosis of myocardial infarction.[35] A number of decision aids have been developed to assist with this problem but have not achieved general use. Machine learning techniques, including classification tree and logistic regression (LR) methods, have the potential to create simple but accurate decision aids. Both a classification tree (FT Tree) and an LR model (FT LR) have been developed to predict the probability that a patient with chest pain is having an MI based solely upon data available at time of presentation to the ER. Training data came from a data set collected in Edinburgh, Scotland. Each model was then tested on a separate Edinburgh data set, as well as on a data set from a different hospital in Sheffield, England. Unlike previous work this study indicates that classification trees, which have certain advantages over LR models, may perform as well as LR models in the diagnosis of

patients with MI. Babic *et al.* show the use of fuzzy decision trees in supporting decision making in breastfeeding.[36] Instead of crisp borders between values fuzzy borders are used, which are more general and user friendly. The use of decision trees in the identification of signals of possible adverse drug reactions is shown by Jones.[37] The emergence of large databases of adverse event data and the need to identify signals of new, unknown adverse effects of newly marketed drugs by regulators and pharmaceutical sponsors have coincided with the development of several methods of data mining for identifying new associations within all types of databases. He concludes that data mining based on decision trees methods show promise for the identification of new, unknown signals of AEs, especially in defined populations. In 1997, health authorities of the state of Sao Paulo, Brazil, designed a vaccination campaign against measles based on a decision model that utilized fuzzy logic.[38] The chosen mass vaccination strategy was implemented and changed the natural course of the epidemic in that state. Authors built a model using a decision tree and compare it to the fuzzy logic model. Using essentially the same set of assumptions about this problem, they contrasted the two approaches. The models identify the same strategy as being the best one, but exhibit differences in the ranking of the remaining strategies. Dantchev shows that computer based decision-making techniques like decision trees are still in the experimental stages and remains difficult to apply to clinical practice in psychiatry.[39] However they appear to be of a great interest, not only in communicating knowledge both in teaching and training, but in research as well. They allow us to view the results of epidemiological studies and clinical research from a more global perspective; to make evident the gray areas of our science and to determine new priorities in research.

Gambhir in his review focuses primarily on the methodology involved in properly reviewing the medical literature for performing a meta-analysis and on methods for performing a formal decision analysis using decision trees.[40] Issues related to performing a detailed meta-analysis with consideration of particular issues, including publication bias, verification bias and patient spectrum, are addressed. The importance of collecting conventional measures of test performance (e.g., sensitivity and specificity) and of changes in patient management to model the cost-effectiveness of a management algorithm is detailed. With greater utilization of the techniques discussed in this review, nuclear medicine researchers should be well prepared to compete for the limited resources available in the current health care environment. Furthermore, nuclear medicine physicians will be better prepared to best serve their patients by using only those studies with a proven role in improving patient management.

The high incidence of false alarms in the intensive care unit (ICU) necessitates the development of improved alarming techniques. A study aimed to detect artifact patterns across multiple physiologic data signals from a neonatal ICU using decision tree induction was performed by Tsien *et al.*[41] Approximately 200 h of bedside data were analyzed in the study. Artifacts in the data streams were visually located and annotated retrospectively by an experienced clinician. Derived values were calculated for successively overlapping time intervals of raw values, and then used as feature attributes for the induction of models trying to classify "artifact" versus "not artifact" cases. The results are very promising, indicating that integration of multiple

signals by applying a classification system to sets of values derived from physiologic data streams may be a viable approach to detecting artifacts in neonatal ICU data.

Bonner examined the application of the decision tree approach to collaborative clinical decision-making in mental health care in the United Kingdom.[42] While this approach to decision-making has been examined in the acute care setting, there is little published evidence of its use in clinical decision-making within the mental health setting. The complexities of dual diagnosis (schizophrenia and substance misuse in this case example) and the varied viewpoints of different professionals often hamper the decision-making process. This paper highlights how the approach was used successfully as a multiprofessional collaborative approach to decision-making in the context of British community mental health care.

Letourneau *et al.* used a decision tree approach in decision making for chronic wound care.[43] Data were collected from two groups of home care nurses in large urban centers. One group was measured after initial contact with the decision tree, and the other group was measured two years after implementation of the decision tree. The chronic wound management decision tree (CWMDT) was used, in combination with pictorial case studies. They conclude that a decision tree can assist with decision making by guiding the nurse through assessment and treatment options.

Although decision models can provide a formal foundation for guideline development and clinical decision support, their widespread use is often limited by the lack of platform-independent software that geographically dispersed users can access and use easily without extensive training. To address these limitations Sanders *et al.* developed a World Wide Web based interface for previously developed decision models.[44] They describe the use and functionality of the interface using a decision model that evaluates the cost-effectiveness of strategies for preventing sudden cardiac death. The system allows an analyst to use a web browser to interact with the decision model and to change the values of input variables within prespecified ranges, to specify sensitivity or threshold analyses, to evaluate the decision model, and to view the results generated dynamically. The web site also provides linkages to an explanation of the model, and evidence tables for input variables. The system demonstrates a method for providing distributed decision support to remote users such as guideline developers, decision analysts, and potentially practicing physicians. The web interface provides platform-independent and almost universal access to a decision model. This approach can make distributed decision support both practical and economical, and has the potential to increase the usefulness of decision models by enabling a broader audience to incorporate systematic analyses into both policy and clinical decisions.

The purpose of the study by Sims *et al.* was to determine whether decision tree-based methods can be used to predict cesarean delivery.[45] The study was a historical cohort study of women delivered of live-born singleton neonates in 1995 through 1997. The frequency of cesarean delivery was 17%; 78 variables were used for analysis. Decision tree rule based methods and logistic regression models were each applied to the same 50% of the sample to develop the predictive training models and these models were tested on the remaining 50%. Decision tree receiver operating characteristic curve areas were as follows: nulliparous, 0.82; parous, 0.93. Logistic receiver operating characteristic curve areas were as follows: nulliparous,

0.86; parous, 0.93. Decision tree methods and logistic regression methods used similar predictive variables; however, logistic methods required more variables and yielded less intelligible models. Among the six decision tree building methods tested, the strict minimum message length criterion yielded decision trees that were small yet accurate. Risk factor variables were identified in 676 nulliparous cesarean deliveries (69%) and 419 parous cesarean deliveries (47.6%). Decision tree models can be used to predict cesarean delivery. Models built with strict minimum message length decision trees have the following attributes: Their performance is comparable to that of logistic regression; they are small enough to be intelligible to physicians; they reveal causal dependencies among variables not detected by logistic regression; they can handle missing values more easily than can logistic methods; they predict cesarean deliveries that lack a categorized risk factor variable.

CONCLUSION

Decision trees are a reliable and effective decision making technique that provide high classification accuracy with a simple representation of gathered knowledge. When using decision trees, the decision-making process itself can be easily validated by an expert. Because of these reasons decision trees are especially appropriate to support decision-making process in medicine (see Table I).

Table I. Various Decision Tree Induction Approaches Summarized

	Description (method)	Induction approach	Discretization method	Space partitioning	Number of decision attributes
Ref. 1	Classical (ID3, C4.5, . . .)	Heuristic	Equidistant/ percentile/ dynamic	Orthogonal	One
Ref. 36	Fuzzy classical	Heuristic	Fuzzy	Orthogonal	One
Ref. 8	Hybrid (MtDecit 2.0)	Heuristic/ neural nets	Dynamic	Oblique	One
Ref. 5	Classical (CART)	Heuristic/ perturbations	Equidistant/ percentile/ dynamic	Oblique	One
Ref. 12	Classical (SADT)	Heuristic/ simulated annealing	Equidistant/ percentile/ dynamic	Oblique	One
Ref. 10	Classical (OC1)	Heuristic/ random	Equidistant/ percentile/ dynamic	Oblique	One
Ref. 16	Classical incremental (ID5R)	Heuristic/ incremental	Equidistant/ percentile	Orthogonal	One
Ref. 30	Evolutionary (Genetic Trees)	Genetic algorithms	Random	Orthogonal	One
Ref. 31	Evolutionary vector (VEDEC)	Genetic algorithms	Random	Orthogonal	Any
Ref. 32	Automatic programming (AREX)	Genetic algorithms/ genetic programming	Random	Oblique	One

There is a lot of decision tree induction methods which we tried to present briefly in this paper and are summarized in Table I. Naturally, none of the described methods is superior to others in all aspects of quality; each of the methods has some advantages and disadvantages. To make the choice of an induction method for a specific problem even more complicated, a method that performs very well for one task can result poorly for some other. In this manner, the best advice would be to use several methods instead of a single one when possible. The main objective of this paper is to inform a reader of the broad variety of methods available for inducing decision trees.

REFERENCES

1. Quinlan, J. R., *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Francisco, 1993.
2. Quinlan, J. R., Induction of decision trees. *Mach. Learn.* 1:81–106, 1986.
3. Quinlan, J. R., Simplifying decision trees, *Int. J. Man–Mach. Stud.* 27:221–234, 1987.
4. Shannon, C., and Weaver, W., *The Mathematical Theory of Communication*, University of Illinois Press, USA, 1949.
5. Breiman, L., Friedman, J. H., Olsen, R. A., and Stone, C. J., *Classification and Regression Trees*, Wadsworth, USA, 1984.
6. Paterson, A., and Niblett, T. B., *ACLS Manual*, Intelligent Terminals Ltd., Edinburgh, 1982.
7. Zorman, M., Podgorelec, V., Kokol, P., Peterson, M., and Lane, J., Decision tree's induction strategies evaluated on a hard real world problem. *Proc. 13th IEEE Symp. Comp.-Based Med. Syst.* (*CBMS-2000*) pp. 19–24, 2000.
8. Zorman, M., Hleb S., and Sprogar, M., Advanced tool for building decision trees MtDecit 2.0. *Proc. Int. Conf. Artif. Intellig.* (*ICA1-99*), 1999.
9. Tou, J. T., and Gonzalez, R. C., *Pattern Recognition Principles*, Addison-Wesley, Reading, MA, 1974.
10. Murthy, K. V. S., *On Growing Better Decision Trees from Data*, PhD dissertation, Johns Hopkins University, Baltimore, MD, 1997.
11. Neapolitan, R., and Naimipour, K., *Foundations of Algorithms*, D.C. Heath and Company, Lexington, MA, 1996.
12. Heath, D., Kasif, S., and Salzberg, S., k-DT: A multi-tree learning method. *Proc. Second Int. Workshop Multistrategy Learn.* pp. 138–149, 1993.
13. Heath, D., Kasif, S., and Salzberg, S., Learning oblique decision trees. *Proc. Thirteenth Int. Joint Conf. Artif. Intellig.* (*IJCA1-93*) pp. 1002–1007, 1993.
14. Rich, E., and Knight, K., *Artificial Intelligence* (2nd edn.), McGraw Hill, New York, 1991.
15. Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., Optimization by simulated annealing. *Science* 220:4598, 1983.
16. Utgoff, P. E., Incremental induction of decision trees. *Mach. Learn.* 4(2):161–186, 1989.
17. Crawford, S., Extensions to the CART algorithm. *Int. J. Man–Mach. Stud.* 31(2):197–217, 1989.
18. Dietterich, T. G., and Kong, E. B., Machine learning bias, statistical bias and statistical variance of decision tree algorithms. Technical Report, Oregon State University, 1995.
19. Ho, T. K., The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intellig.* 20(8):832–844, 1998.
20. Podgorelec, V., and Kokol, P., Evolutionary decision forests—decision making with multiple evolutionary constructed decision trees, *Problems in Applied Mathematics and Computational Intelligence*, pp. 97–103, WSES Press, 2001.
21. Shlien, S., Multiple binary decision tree classifiers. *Pattern Recogn. Lett.* 23(7):757–763, 1992.
22. Utgoff, P. E., Perceptron trees: A case study in hybrid concept representations. *Connect. Science* 1:377–391, 1989.
23. Craven, M. W., and Shavlik, J. W., Extracting tree-structured representations of trained networks. In *Advances in Neural Information Processing Systems, Vol. 8*, MIT Press, Cambridge, MA, 1996.
24. Zorman, M., Kokol, P., and Podgorelec, V., Medical decision making supported by hybrid decision trees. *Proc. ICSC Symp. Intellig. Syst. Appl.* (*ISA-2000*) ICSC Academic Press, 2000.
25. Banerjee, A., Initializing neural networks using decision trees. *Proc. Int. Workshop Comput. Learn. Nat. Learn. Syst.* pp. 3–15, 1994.
26. Goldberg, D. E., Genetic algorithms in search, optimization, and machine learning, Addison Wesley, Reading, MA, 1989.

27. Nikolaev, N., and Slavov, V., Inductive genetic programming with decision trees. *Intellig. Data Anal. Int. J.* 2(1):31–44, 1998.
28. Podgorelec, V., and Kokol, P., Induction f medical decision trees with genetic algorithms. *Proc. Int. ICSC Congr. Comput. Intellig. Methods Appl.* (*CIMA 1999*) 1999.
29. Cantu-Paz, E., and Kamath, C., Using evolutionary algorithms to induce oblique decision trees. *Proc. Genet. Evol. Comput. Conf.* (*GECCO-2000*) pp. 1053–1060, 2000.
30. Podgorelec, V., and Kokol, P., Towards more optimal medical diagnosing with evolutionary algorithms. *J. Med. Syst.* 25(3):195–219, 2001.
31. Sprogar, M., Kokol, P., Hleb, S., Podgorelec, V., and Zorman, M., Vector decision trees. *Intellig. Data Anal.* 4(3/4):305–321, 2000.
32. Podgorelec, V., *Intelligent Systems Design and Knowledge Discovery With Automatic Programming*, PhD thesis, University of Maribor, Oct. 2001.
33. Cremilleux, B., and Robert, C., A theoretical framework for decision trees in uncertain domains: Application to medical data sets. In *Lecture Notes in Artificial Intelligence, Vol. 1211*, pp. 145–156, Springer-Verlag, 1997.
34. Kokol, P., Zorman, M., Stiglic, M. M., and Malcic, I., The limitations of decision trees and automatic learning in real world medical decision making. *Proc. 9th World Congr. Med. Inform.* (*MEDINFO*-98) Vol. 52, pp. 529–533, 1998.
35. Tsien, C. L., Fraser, H. S. F., Long, W. J., and Kennedy, R. L., Using classification tree and logistic regression methods to diagnose myocardial infarction. *Proc. 9th World Congr. Med. Inform.* (*MEDINFO-98*) Vol. 52, pp. 493–497, 1998.
36. Babic, S. H., Kokol, P., and Stiglic, M. M., Fuzzy decision trees in the support of breastfeeding. *Proc. 13th IEEE Symp. Comp.-Based Med. Syst.* (*CBMS-2000*) pp. 7–11, 2000.
37. Jones, J. K., The role of data mining technology in the identification of signals of possible adverse drug reactions: Value and limitations. *Curr. Ther. Res.-Clin. Exp.* 62(9):664–672, 2001.
38. Ohno-Machado, L., Lacson, R., and Massad, E., Decision trees and fuzzy logic: A comparison of models for the selection of measles vaccination strategies in Brazil. *J. Am. Med. Inform. Assoc.* (Suppl.):625–629, September 2000.
39. Dantchev, N., Therapeutic decision frees in psychiatry. *Encephale-Revue De Psychiatrie Clinique Biologique Et Therapeutique* 22(3):205–214, 1996.
40. Gambhir, S. S., Decision analysis in nuclear medicine. *J. Nucl. Med.* 40(9):1570–1581, 1999.
41. Tsien, C. L., Kohane, I. S., and McIntosh, N., Multiple signal integration by decision tree induction to detect artifacts in the neonatal intensive care unit. *Artif. Intellig. Med.* 19(3):189–202, 2000.
42. Bonner, G., Decision making for health care professionals: Use of decision trees within the community mental health setting. *J. Adv. Nurs.* 35:349–356, 2001.
43. Letourneau, S., and Jensen, L., Impact of a decision tree on chronic wound care. *J. Wound Ostomy Continence Nurs.* 25:240–247, 1998.
44. Sanders, G. D., Hagerty, C. G., Sonnenberg, F. A., Hlatky, M. A., and Owens, D. K., Distributed decision support using a web-based interface: Prevention of sudden cardiac death, *Med. Decision Making* 19(2):157–166, 2000.
45. Sims, C. J., Meyn, L., Caruana, R., Rao, R.B., Mitchell, T., and Krohn, M., Predicting cesarean delivery with decision tree models. *Am. J. Obstet. Gynecol.* 183:1198–1206, 2000.