Building Intelligent Decision-Tree Algorithm for Classification Problems

Pei–Lei Tu[*]
National Cheng-Chi University
Taipei, Taiwan
peilei@watson.ibm.com

Jen–Yao Chung
IBM Thomas J. Watson Research Center
Yorktown Heights, New York 10598, USA
jychung@watson.ibm.com

Abstract

Classification is a general problem-solving approach that takes a collection of examples from multiple groups as input and identifies the characteristic pattern or property for each group. The traditional approach to classification problems has been focused on the use of statistical methods. But, the usefulness of these statistical methods is limited since some classification problems may not meet the required statistical requirements. Our approach to classification is to use decision trees and it requires no statistical assumptions but instead uses the frequency ratios computed directly from the data. We study the current decision-tree classification algorithm, analyze its anomalies, and present new concepts to overcome these anomalies.

Key Areas: Artificial Intelligence & Expert Systems

[*]Pei-Lei Tu is currently a visiting scientist at IBM Thomas J. Watson Research Center Yorktown Heights, New York 10598, USA

1 Introduction

Classification is a general problem-solving approach that takes a collection of examples from multiple groups as input and identifies the characteristic pattern or property for each group. The traditional approach to classification problems has been to use statistical methods, such as *multiple discriminant analysis* (MDA). One drawback of this method is that it makes statistical assumptions on the input data. For example, MDA requires that data should have a multivariate normal distribution, each group should have the same variance-covariance matrix, and each group needs to be totally distinct from the others. The use of these statistical methods is limited since some classification problems may not meet these required statistical requirements. For example, a classification problem with a mixture of both discrete and continuous variables surely does not have the normal distribution structure. Also, the output obtained from these statistical methods is not explicitly useful unless further interpretations are made.

One approach to classification, as an alternative to statistical methods, is to use decision trees. This approach requires no statistical assumptions but uses the frequency ratios computed directly from the data. The output generated from this approach is a decision tree that partitions and classifies the input events into their groups in a divide-and-conquer approach. Many decision problems have been analyzed using decision trees. These include applications in medical diagnosis, financial distress or bankruptcy prediction, manufacturing testing, and chess endgames [10].

The use of decision trees for classification is best exemplified by the emerging technologies of knowledge acquisition in knowledge-based systems. One major type of knowledge representation used in knowledge-based systems is the if-then decision rule. Since the decision tree itself is a set of nested if-then rules, a decision-tree method can be viewed as an automatic knowledge acquisition procedure that follows the learning strategy, *learning from examples* [9]. One implementation of this learning strategy using decision-tree is the machine learning program, ID3 [10], which has been applied widely.

There are issues which are either ignored or unclear from the literature of machine learning. They are the anomalous behaviors of ID3 algorithm that may lead to an inferior decision tree requiring more decisions to classify multiple group objects. A common search procedure used by this algorithm is to search for the first best attribute, continuing to the next one by combining it with the previously selected attribute, and so on. This approach has an anomaly, however. That is, even if attributes are the best locally, they may not be the best globally. As a result, the tree generated based on local optimum may not be able to generate globally optimal trees.

Based on findings in the above analyses, alternative concepts, such as the neighborhood search and the nearest-neighbor dependence, will be used in a new heuristic algorithm, *intelligent decision-tree* algorithm (IDA) which overcomes these anomalies to produce better (or simpler) classification trees.

The remainder of this paper is organized as follows. Section 2 studies the current decision-tree classification algorithm, analyzes its anomalies, and present new concepts to overcome these anomalies. Section 3 defines the intelligent decision-tree algorithm for discrete attributes, IDA. It also summarizes the comparative study of IDA and ID3. Section 4 describes the equivalent probability blocks for continuous attributes. Section 5 extends the IDA for continuous attributes. Section 6 describes future work.

2 Study of the Decision-Tree Classification Algorithm in Machine Learning

Although the decision-tree classification algorithm has been widely used as the machine learning technique in artificial intelligence, there has been little research toward evaluating the time and computational complexity of constructing the optimal (smallest) decision tree for classification, and investigating the performance or quality and anomalies of the current classification algorithm.

2.1 Complexity of Constructing an Optimal Classification-Tree

By using a known NP-Complete problem, 3-exact cover, the complexity of constructing an optimal decision tree has been proved to be NP-Complete [11]. Therefore, there exists no computationally efficient classification algorithm for the optimal decision tree but heuristic algorithms.

2.2 Anomalies of classification-tree algorithms using greedy search

Methodologies of ID3 algorithm are inherited from theory in information theory and pattern recognition. The program uses the information measure function *entropy* as the criterion function. Given a two-group classification problem with each object a m-dimensional vector $\mathbf{x} = (f_1, \ldots, f_m)$, the entropy function to measure the uncertainty of the two possible classification outcomes, group w_1

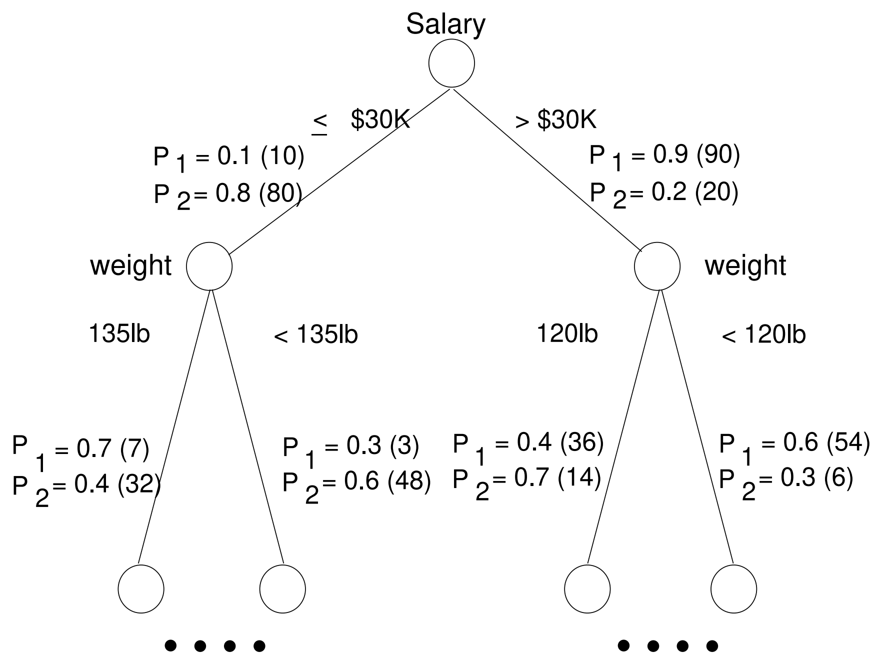

Figure 1: A classification tree with poor attributes

or w_2, can be defined as

$$H(\mathbf{x}) = \sum_{\mathbf{x}} -P(w_1 \,|\, \mathbf{x}) \log_2 P(w_1 \,|\, \mathbf{x}) - P(w_2 \,|\, \mathbf{x}) \log_2 P(w_2 \,|\, \mathbf{x}),$$

where $P(w_i \,|\, \mathbf{x})$ is the *a posteriori* probability of $\mathbf{x}$ in population w_i.

ID3 algorithm follows the greedy search procedure. First, each of the m attributes is evaluated using the entropy function and the one with the smallest entropy function $H(f_i)$ $(i = 1, \ldots, m)$ is selected. Second, given the selected attribute f_i, the next attribute f_j should be the one with the smallest entropy $H(f_i, f_j)$ where $f_j = f_1, \ldots, f_m$ and $f_j \neq f_i$ if they are both discrete attributes.

Given two groups of people, 100 women and 100 men, there are two decision trees for determining their sex as shown in Figures 1 and 2. The tree in Figure 2 is definitely better than the one in Figure 1, because more females are differentiated from males. The decision tree generated by ID3 algorithm is the one with poorly selected attributes in Figure 1 instead of the one with good classification attributes in Figure 2.

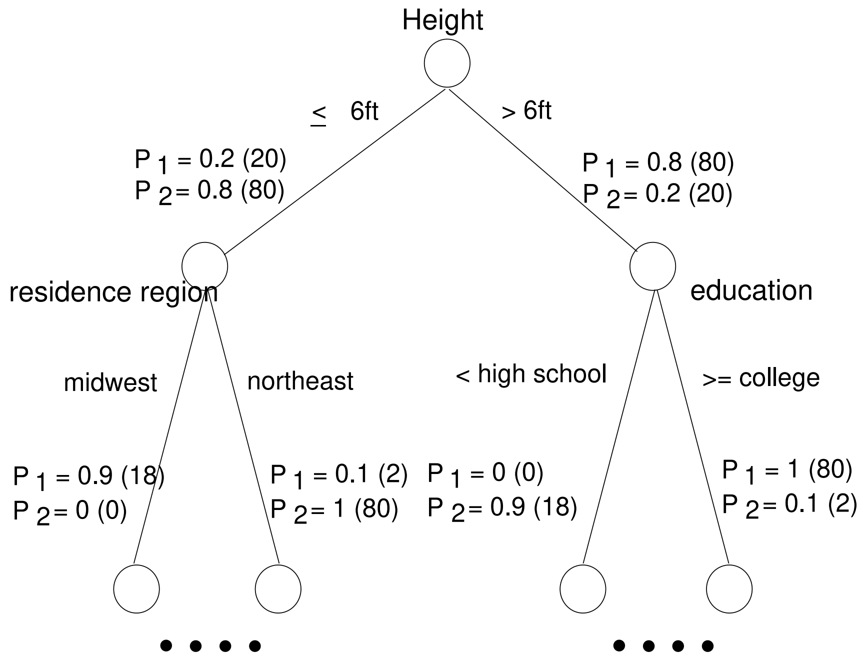

Figure 2: A classification tree with good attributes

2.3 Complexity of the greedy search for classification-tree algorithms

Since the optimal classification tree can be found only through the exhaustive search, the search space is usually enormous. The greedy heuristic search used in ID3 algorithm is to select the locally best attribute and will not backtrack to previous eliminated candidates. The overall search space decreases drastically by concentrating on that particular attribute only. As shown in Figure 3, the search space at level 1 is m, and becomes $2(m-1)$ at level 2. The general form of the search space at level i is $2^{i-1}(m-(i-1))$.

Therefore, in the worst case when every attribute is used, the total search space from the top (level 1) to the bottom (level m) is

$$\sum_{i=1}^{m} 2^{i-1}(m-(i-1)).$$

Such an algorithm also requires exponential computer time. But in contrast to the exhaustive search algorithm, the greedy algorithm will reduce the computer time drastically.

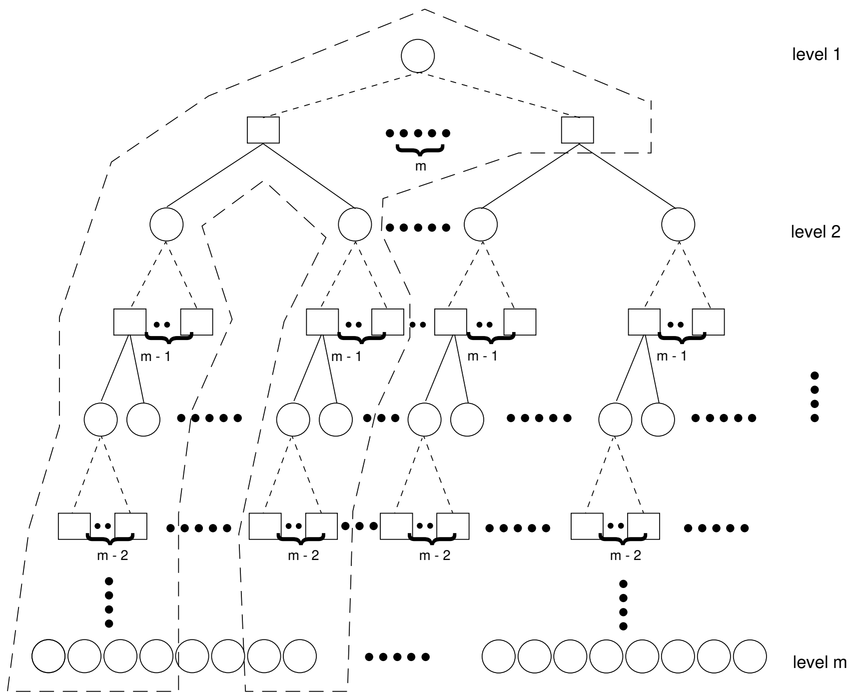

Figure 3: The search space for the greedy search

2.4 The neighborhood search for classification algorithms

Although the greedy algorithm works for some computationally intractable problems [4], it may not work for classification problems. By concentrating on the locally best attribute will overlook the effect from later added attributes, and this in turn may result in an inferior classification tree. Therefore, instead of focusing on a single local solution (e.g., the locally best attribute), we need to obtain more information from the given data and try to improve the initial feasible solution from the viewpoint of enhancing the global classification performance. If a locally best solution does not satisfy the goal of enhancing the global performance, it can be substituted by other solutions in the neighborhood which may have worse local optimization but may have more global enhancement. Similar procedures have been used to find "good" solutions for traveling salesman problems [6,7].

An approximated search space for the neighborhood search for classification-tree algorithms is shown in Figure 4. The size of this search space is definitely smaller than the one of the exhaustive search space. But the comparison between this search space and the one for the greedy search depends on the neighborhood we choose. A large neighborhood produces more alternatives, but at the same time costs more computer time. Suppose we measure the global dependency structure before tree splitting and use this to determine the neighborhood candidates in each node later

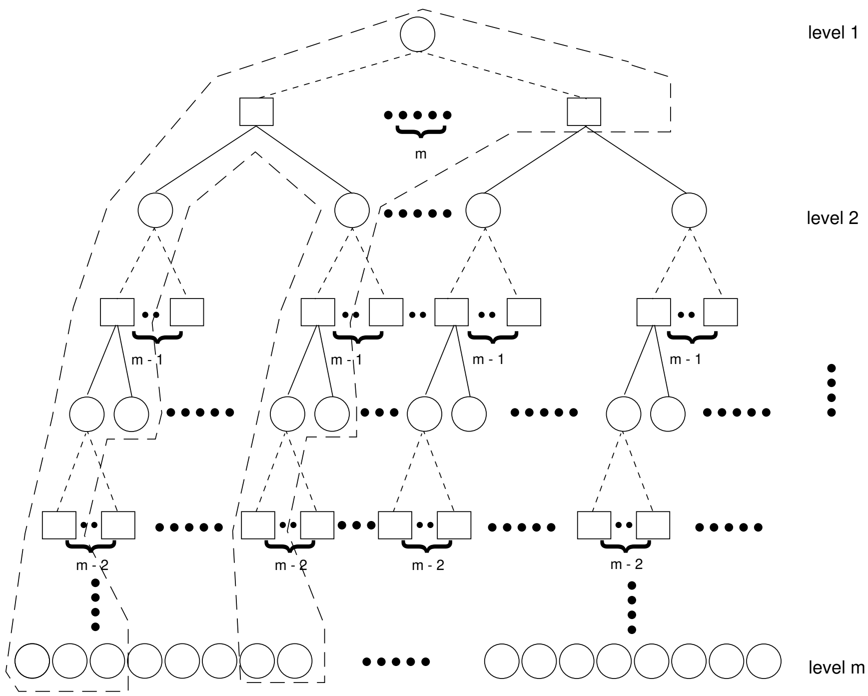

Figure 4: The search space for the neighborhood search

on. Assume only half of all possible candidates will be in the neighborhood. Contrasted to the search space for the greedy search, the general form for the number of alternatives at level i is $2^{i-1}[(m-i+1)/2]$ or $2^{i-2}(m-i+1)$. In the worst case when every attribute is used, the total search space will be

$$\sum_{i=1}^{m} 2^{i-2}(m-i+1).$$

The search space for the neighborhood search is only one half of the one for the greedy search. Besides the smaller search space and less complexity, the neighborhood search procedure will be demonstrated to have better classification performance than ID3 in a newly created classification algorithm IDA.

3 The intelligent decision-tree algorithm (IDA)

Instead of using entropy function, the IDA algorithm uses the *divergence measure* [5,8], another classification criterion developed in information theory. To consider the dependency relation of attributes f_i and f_j and their combined classification performance, the divergence measure of these two attributes, given a value of attribute f_i, can be defined as

$$Div(f_j \mid f_i) = \begin{cases} P(f_j \mid f_i)(P_1(f_j \mid f_i) - P_2(f_j \mid f_i)) \log \frac{P_1(f_j \mid f_i)}{P_2(f_j \mid f_i)}, & P_1(f_j \mid f_i) \text{ and } P_2(f_j \mid f_i) > 0 \\ K \times P(f_j \mid f_i), & P_1(f_j \mid f_i) \text{ or } P_2(f_j \mid f_i) = 0 \end{cases}$$

where K is an arbitrarily large number, and $P(f_j \mid f_i)$ is the average conditional probability of f_j given f_i.

IDA utilizes the global dependency structure obtained from every two attributes and uses this as a criterion to select attributes so the dependency between attributes can be taken into consideration in the classification. The global dependency structure is obtained from the entire data set in the beginning and measured by the conditional divergence function. It is then used in the neighborhood search procedure in IDA to identify other locally potential solutions. The locally best solutions may be substituted by other neighboring alternatives to enhance the classification performance globally.

Besides being used to measure the neighborhood in the neighborhood search, the global dependency structure is also used by IDA for look-ahead. The selection of attributes is based on not only its individual classification effects but also its associated classification effects with every other attribute. In this way, individually best attributes won't be selected if its combined classification effects with other attributes are poor. The look-ahead approach adopts the concept of nearest-neighbor dependency [2]. The interactions between a new attribute and the already selected attributes can be approximated by considering only the interactions between this new attribute and those nearest-neighbor attributes on the parent and grandparent nodes in the tree, and other interactions can be ignored. Therefore the look-ahead approach used by IDA only considers relations of attributes at the three nearest levels in the tree.

Given a collection of n objects $\mathbf{x} = (f_1, \ldots, f_m, c)$ where c is the class to which an object belongs. The IDA algorithm is described below. Right now, this algorithm only handles the discrete attributes. Details of this algorithm are as follows:

Intelligent Decision-tree Algorithm :

(1) For each distinct value f_{ir} of attribute f_i, compute its conditional divergence measures $Div(f_j \mid f_{ir})$ with the remaining attributes f_j. Select the largest divergence measure $Div(f_j^* \mid f_{ir})$.

(2) For each attribute f_i, compute the average divergence measure $E_{f_{ir}}\left[Div(f_j^* \mid f_{ir})\right] = \sum_{f_{ir}} P(f_{ir}) Div(f_j^* \mid f_{ir})$.

(3) Select the attribute f_i with the largest average divergence measure $E[Div(f_j^* \mid f_i)]$ as the first node in the tree. Subnodes are created for each value f_{ir} of f_i. A subnode f_{ir} is a terminal subnode, if objects satisfying this subtree are from the same class, otherwise a nonterminal subnode.

(4) The next attribute for each nonterminal subnode f_{ir} will be the attribute f_j with the largest divergence measure $Div(f_j^* \mid f_{ir})$, which in turn creates subnodes f_{js}. Check if any subnode is a terminal subnode.

(5) Generate the next attribute f_k for each nonterminal subnode f_{js}. This attribute is checked against its grandparent node f_{ir}. If attributes f_k and f_i are the same attribute or attribute f_k is too closely related to attribute f_i, go to Step (6) to find other neighboring alternatives, otherwise continue to Step (7). We choose to determine that attribute f_k is too closely dependent on f_{ir} if $Div(f_k \mid f_{ir}) < \gamma_i$, where γ_i is the *median* of $\{Div(f_k \mid f_{ir})\ k = 1, \ldots m\}$.

(6) Try other attributes whose divergence measures are in the *neighborhood*. This includes all conditional divergence measures greater than or equal to γ_i. Start with the nearest neighbor with the closest divergence measure. If this attribute has not been used and it is not too closely dependent on its grandparent node, continue to Step (7); otherwise continue the next nearest neighbor until an applicable attribute is found. If no neighboring attribute is applicable, choose the last attempted attribute.

(7) Create the subnodes and mark any terminal subnodes.

(8) Repeat Step (5) for each of the remaining nonterminal nodes.

3.1 A Comparison of IDA and ID3

Since both IDA and ID3 algorithms are heuristic algorithms, it is of interest to compare their effectiveness and efficiency. Their computational complexity is analyzed using the elementary steps of both algorithms: (1) a *computation* for the criterion function, (2) a *comparison* between two values in the search procedure, (3) a *substitution* for an initial solution. The result in [11] showed that when the size of input instance is large the computational complexity of ID3 algorithm is twice as much as that of the IDA algorithm. The major reason is that the greedy search used by the ID3 algorithm needs to check every candidate in each node to determine the locally best attribute.

As the results show, IDA outperforms ID3 in most instances of the variance-covariance matrix. But the performance of IDA increases drastically as the variance-covariance matrix is altered from no interaction to high interaction. This finding is consistent with the methodology of IDA to consider the dependency structure of variables. A further analysis of variances also shows that two parameters, the number of variables and the number of levels, have interaction effects on classification. When both of them are small, IDA is better. When either of them increases, the comparative advantage of IDA declines to the extent that IDA and ID3 perform equally. But ID3 is more computationally inefficient.

4 Equivalent Probability Blocks for Continuous Attributes

Previous sections mainly focus on discrete attributes. In this section we will introduce the continuous attributes into IDA algorithm. A continuous attribute is split based on the value which has the most discriminating power in comparison to the rest of the values for that attribute. From the current decision-tree technique, it is determined by the criterion function *information measure* or the *impurity* of each value of this attribute. There are several drawbacks of this approach:

1. The number of distinct values of a continuous attribute could be enormous which makes the approach inefficient. The worse case is that there are totally n objects and a continuous attribute has n distinct values.

2. In this way, varying the sample set can easily alter the results, especially when there are noisy data or the sample size is not sufficiently large.

Therefore, we need an efficient splitting rule on continuous attributes and the sample-based results should have a tolerable region to tolerate the sensitivity caused by noisy data or sample size. The idea of *statistically equivalent probability blocks* [1,3] is a simple but justifiable approach which can be applied to partition the values of a continuous attribute into several equivalent probability blocks.

Given n distinct values of a continuous attribute, the procedure of determining the equivalent probability blocks of this attribute is described below.

(1) Rank the n distinct values in increasing order.

(2) Let

$$q = \lceil n^{\frac{1}{3}} \rceil \text{ and } \quad p = \lceil \frac{n^{\frac{1}{2}}}{q} \rceil.$$

Partition the above ordering into p blocks by making $(p - 1)$ cuts. Space the cuts across these n distinct values as evenly as possible.

(3) Each block b_k, $1 \leq k \leq p$, is delimited by its lower and upper boundaries lb_k and ub_k, respectively.

Here p is totally determined by the number of values n.

5 The Intelligent Decision-Tree Algorithm with Continuous Attributes

The intelligent decision-tree algorithm described in Section 3 assumes each attribute is discrete and here this algorithm will be modified by adding the concept of equivalent probability blocks for continuous attributes. Details of the algorithm are described below.

Input:

A collection of n objects; each object $\mathbf{x} = (f_1, \ldots, f_m, c)$ has m continuous attributes along with the class c it belongs to.

Output:

A classification tree with each leaf containing objects from the same class.

Processes:

(1) Generate the equivalent probability blocks for each continuous attribute. Now the number V_i of all possible distinct values for each continuous attribute F_i is reduced to the number of blocks. At the same time, value f_{ir} for attribute F_i is an upper bound of a block.

For each distinct value f_{ir} $(1 \leq r \leq V_i)$ of attribute F_i $(1 \leq i \leq m)$, compute the conditional divergence measures

$$Div(F_j \mid F_i \, \phi \, f_{ir}) \;\; = \;\; \sum_{f_{js}} P(F_j \, \phi \, f_{js}) Div(F_j \, \phi \, f_{js} \mid F_i \, \phi \, f_{ir})$$

$$= \sum_{f_{js}} P(F_j\,\phi\,f_{js})\Big(P_1(F_j\,\phi\,f_{js}\,|\,F_i\,\phi\,f_{ir}) - P_2(F_j\,\phi\,f_{js}\,|\,F_i\,\phi\,f_{ir})\Big)$$
$$\log\frac{P_1(F_j\,\phi\,f_{js}\,|\,F_i\,\phi\,f_{ir})}{P_2(F_j\,\phi\,f_{js}\,|\,F_i\,\phi\,f_{ir})}$$

where f_{js} $(1 \le s \le V_j)$ is any value of attribute F_j, $1 \le j \le m$, and ϕ is the $>, <, \le$, or $\ge$ operator. Note that ϕ is equal to "=" in Section 3 for discrete attributes.

If $F_j = F_i$, contradictions between $F_j\,\phi\,f_{js}$ and $F_i\,\phi\,f_{ir}$ should be avoided. For example, if the test $F_1 > 2$ is used before, then $F_1 \le 1$ can not be used.

(2) For each value f_{ir} of attribute F_i, select the largest $Div(F_j^*\,|\,F_i\,\phi\,f_{ir})$ over $F_j, j = 1, \ldots, m$, and compute

$$Div(F_j\,|\,F_i) = \sum_{f_{ir}} P(F_i\,\phi\,f_{ir}) Div(F_j^*\,|\,F_i\,\phi\,f_{ir}).$$

(3) Let F_i be the attribute with the largest $Div(F_j\,|\,F_i)$. This attribute will be selected and the value f_{ir} where the largest $Div(F_j^*\,|\,F_i\,\phi\,f_{ir})$ occurs will be the place to split. Two subnodes are produced with one satisfying $F_i\,\phi\,f_{ir}$ and the other one satisfying $F_i\,\overline{\phi}\,f_{ir}$. The second attribute selected after each subnode is the attribute F_j which maximizes

$$\{Div(F_j\,|\,F_i\,\phi\,f_{ir}),\ j = 1, \ldots, m\}.$$

This attribute in turn generates two subnodes $F_j\,\phi\,f_{js}$ and $F_j\,\overline{\phi}\,f_{js}$

(4) Check if these subnode satisfies either of the following terminal conditions. If so, mark it as a terminal node, otherwise a nonterminal node.

Two *terminal conditions* are

(a) either $P_1(F_j\,\phi\,f_{js}\,|\,F_i\,\phi\,f_{ir})$ or $P_2(F_j\,\phi\,f_{js}\,|\,F_i\,\phi\,f_{ir})$ is equal to zero, or

(b) objects in this node are from the same class.

(5) Select the attribute F_k for each nonterminal subnode. Attribute F_k will be checked against its grandparent node $F_i\,\phi\,f_{ir}$. If it is not *applicable*, go to the neighborhood search procedure in Step 5a to find alternatives.

Attribute F_k is not *applicable* if it is too closely dependent on its grandparent node $F_i\,\phi\,f_{ir}$. Let γ_i denote the *median* divergence measure of the set $\{Div(F_k\,|\,F_i\,\phi\,f_{ir}),\ k = 1, \ldots, m\}$. If $Div(F_k\,|\,F_i\,\phi\,f_{ir}) < \gamma_i$, it means the attribute F_k is too closely dependent on its grandparent node to be applicable.

(5a) Try other attributes with divergence measures in the *neighborhood*. This includes all conditional divergence measures greater than or equal to γ_i. Start from the nearest neighbor with the closed conditional divergence and if it is applicable, return; otherwise, continue the next nearest neighbor until an applicable attribute is found.

If no attribute in the neighborhood is applicable, it means there is no room to improve the performance. If the original attempted attribute has not been used, return this attribute; otherwise return the next neighboring attribute which has not been used before.

(6) Check each subnode $F_k \phi f_{kt}$. If it satisfies either of the following terminal conditions, it will be a terminal node.

Two *terminal conditions* are

(a) either $P_1(F_k \phi f_{kt} \mid F_j \phi f_{js})$ or $P_2(F_k \phi f_{kt} \mid F_j \phi f_{js})$ is equal to zero, or

(b) objects in this node are from the same class.

(7) Repeat Step 5 for each nonterminal node.

In summary, attributes are evaluated not only by their individual discrimination capability but also by their dependency relations with the other attributes. Only in this way the problem that newly added variables may deteriorate the performance can be avoided.

We have discussed the procedures of IDA algorithm for either discrete or continuous attributes, but not both. The algorithm can be easily modified to apply to the cases with a mixture of both discrete and continuous attributes by combining the above two procedures.

6 Future work

In real world applications, classification is an important but sophisticated task. Although we have raised and discussed several important issues in this paper, more work needs to be done: (1) Noisy data can occur in real world situations where the same event occurs in more than one class, or there exists initial misclassifications in the sample set. Classification results based on events with noisy data may not be reliable in classifying new events later. It depends on the classification algorithm itself. If the algorithm is totally data-driven, it may lead to an inferior tree that cannot be improved by adding extra processes, e.g., pruning steps. If the algorithm is not too sensitive to data, the effect of noisy data may be ignored. (2) More work need to be conducted on the performance of continuous data and the mixture of continuous and discrete data.

REFERENCES

[1] Anderson, T. W., "Some nonparametric multivariate procedure based on statistically equivalent blocks," *Proc. of the Int. Symp. on Multivariate Analysis,* New York, Academic Press, 1966.

[2] Chow, C. K., "A recognition method using neighbor dependence," *IRE Transactions on Electronic Computers*, vol. 11, 1962, pp. 683-690.

[3] Gessaman, M. P., and Gessaman, P. H., "A comparison of some multivariate discrimination procedures," *Journal of the American Statistical Association*, vol. 67, June 1972, pp. 468-472.

[4] Hu, T. C., *Combinatorial Algorithms*, 1981.

[5] Kullback, S., *Information Theory and Statistics*, John Wiley and Sons, Inc., 1959.

[6] Lin, S., "Heuristic programming as an aid to network design," *Networks*, vol. 5, 1975, pp. 33-43.

[7] Lin, S., and Kernighan, B. W., "An effective heuristic algorithm for the traveling salesman problem," *Operations Research*, vol. 21, no. 2, 1973, pp. 498-516.

[8] Marill and Green, "On the effectiveness of receptors in recognition systems," *IEEE Transactions on Information Theory*, January 1963, pp. 11-17.

[9] Michalski, R. S., Carbonell, J. G., and Mitchell, T. M. (Eds.), *Machine Learning: An Artificial Intelligence Approach*, Palo Alto: Tioga Publishing Company, 1983.

[10] Quinlan, J. R., "Learning efficient classification procedures and their application to chess endgames," in *Machine Learning: An Artificial Intelligence Approach*, Michalski, R. S., Carbonell, J. G., and Mitchell, T. M. (Eds.), Palo Alto, Tioga Publishing Company, 1983.

[11] Tu, P.-L., and Chung, J.-Y., "A new decision-tree classification algorithm for machine learning," *IEEE 4th International Conference on Tools for Artificial Intelligence,* Arlington, Virginia, Nov. 10-13, 1992.