Use of Attribute Selection Criteria in Decision Trees in Uncertain Domains

B. Crémilleux[1], **C. Robert**[2]

[1]GREYC, CNRS - UPRESA 6072
Université de Caen
F-14032 Caen Cédex France
cremilleux@info.unicaen.fr
[2]Institut de Recherche en Mathématiques Appliquées
Université Joseph Fourier
F-38041 Grenoble Cédex France
Claudine.Robert@imag.fr

Abstract

There are many attribute selection criteria involved in the induction of decision trees. We present criteria derived from an impurity measure in a unified framework and we use the same background to describe ORT criterion [12]. We compare the theoretical backgrounds of C.M. criteria with those of the ORT criterion and we illustrate their differences. We set lazy decision trees with regards to these families of criteria, and we debate their use with uncertain data sets. Finally, we clarify the choice of these approaches and we separate their domains of application.

1 Introduction

Decision trees have been used successfully for many different decision making and classifications tasks in many domains like medicine. Broadly speaking, a decision tree is built from a set of training data having attribute values and a class name. The result of the process is represented as a tree in which nodes specify attributes and branches specify attribute values. Leaves of the tree correspond to sets of examples with the same class or to elements in which no more attributes are available. Construction of decision trees is described, among others, by Breiman et al. (1984) [3] who present an important and well-known monograph on classification trees. A number of standard techniques have been developed, for example like the basic algorithms ID3 [19], CART [3] and the questionnaire theory [16]. A survey of different methods of decision tree classifiers and the various existing issues are presented in Safavian and Landgrebe (1991) [22].

In induction of decision trees various attribute selection criteria are used to estimate the quality of attributes in order to select the best one to split on. Section 2 briefly sets out two main paradigms to generate decision trees: the C.M. criteria which are stemmed from an impurity measure [3][5] and the ORT criterion [12] which is based on a canonical scalar product. We present C.M. criteria in an unified framework [9] to better confront these paradigms and we compare the theoretical backgrounds of C.M. criteria with those of the ORT criterion. We connect the question of the advantages and disadvantages of each of these families of criteria with the presence or not of uncertainty in the used data and the necessity to prune or not the tree to avoid overspecified information. Section 3 briefly illustrates differences which may happen in practical situations between C.M. and ORT criteria. In Section 4, we set lazy decision trees with regards to these families of criteria and we debate their use in uncertain domains. To conclude, we will see that the choice of a paradigm, according to us, depends whether the used data sets embed uncertainty or not, whether the phenomenon under study admits deterministic causes, and what level of intelligibility is required.

2 C.M. and ORT criteria

2.1 C.M. criteria

Many possible attribute selection criteria for building decision trees exist, but we know at a theoretical level that criteria derived from an impurity measure perform comparably [3][5][9]. We used here the term of impurity because it is the more usual in the machine learning community [3][12]. We called such criteria C.M. criteria (concave-maximum criteria) because an impurity measure, among other characteristics, is defined by a concave function. The most commonly used criteria which are the Shannon entropy (in the family of C4.5 software [20]) and the Gini criterion (in CART algorithms [3]), are C.M. criteria. We present criteria derived from an impurity measure in an unified framework [9] in order to better understand C.M. and ORT criteria.

To compare these families of criteria, it is necessary now to recall the basic constraints of the problem [3][9]. Let us call D the class of the examples of a data set, with values $d_1, \ldots, d_k, \Omega$ a node and Y any attribute defined on Ω, with values $y_1, \ldots, y_m$. Let us consider an attribute selection criterion denoted Ψ (see Fig. 1).

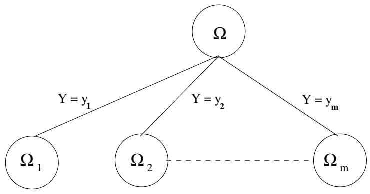

Figure 1: Splitting of a node Ω using an attribute Y.

i. The minimum value of Ψ is reached if and only if the sub-nodes induced by Y are pure with respect to D, that is if and only if two examples with the same value of Y imply that they have the same value of D.

ii. Let P (respectively $P_i, i = 1, \ldots, m$) be the frequency distribution of D in Ω (respectively on the set $Y = y_i, i = 1, \ldots, m$). Ψ has its maximum value if and only if $P_1 = \ldots = P_m$ (which implies that $P = P_i$ for all i).

iii. $\Psi(Y)$ can be viewed as a combined measure of impurity of the sub-nodes induced by Y. If we want Ψ to take into account the respective sizes of the sub-nodes Ω_i, the simplest form for Ψ is: $\Psi(Y) = \sum_i^m \alpha_i \varphi(\Omega_i)$ where $\Omega_1, \ldots, \Omega_m$ are the sub-nodes yielded by Y, α_i is the rate of Ω_i in Ω, and φ is a function that quantifies the impurity of D in the node Ω_i.

Thus we are led to address the question of defining an impurity measure. A minimal set of constraints is:

1. The impurity of D in Ω depends only on the frequency distribution $P = (p_1, \ldots, p_k)$ of D in Ω. Thus a measure of impurity is a function φ defined on the set of $k - uples$ with positive coordinates, the sum of which is 1. We will note either $\varphi(P)$ or $\varphi(\Omega)$ the impurity of a node Ω.

2. φ doesn't depend on the way we code D. Thus φ has to be equal over all permutations of the components of P. The mathematical term for this is: φ is a symmetric function.

3. φ reaches its minimum value minφ if and only if D is a constant function on Ω.

4. φ reaches its maximum value maxφ if all values of D are equally frequent.

5. Combining groups tends to increase the impurity. For example, when we combine r groups Ω_i, D might be constant on each of them, while D might not be constant on the whole group; from i) $\varphi(\Omega_i) = \min \varphi$ and $\varphi(\Omega) \geq \min \varphi$ with $\Omega = \Omega_1 \cup \ldots \cup \Omega_r$.

One can then show [9], that if φ is strictly concave and is symmetric, then it satisfies constraints (1) to (5) and any element of the set $\mathcal{C}$ of symmetric strictly concave functions is a proper function to define an impurity measure. So, from this set of constraints, we have a whole set of possible criteria, which will be called concave minimum criteria (for example, the information gain and the Gini criterion are C.M. criteria). The value of the criterion in a node reflects how appropriately the chosen attribute divides the data. With impurity measures, the smaller the value of Ψ, the better the split. Let us recall that, from iii), $\Psi(Y)$ can be viewed as a combined measure of impurity of the sub-nodes induced by Y: we will see that this point is of great interest to explain differences between ORT and C.M. criteria.

Many domains are uncertain and in such areas, like medicine, we are sure a priori that it is impossible to build a tree that correctly classifies all the examples. In such situations, decision tree algorithms tend to divide nodes having few examples and a main drawback appears (see [3][18][6][23]): the resulting trees tend to be very large and overspecified. Some branches, especially towards the bottom, are due to sample variability and are statistically meaningless (one can also say that they are present due to noise in the sample). Such branches must either not be built or be pruned. If we do not want to build them, we have to set out rules to stop the building of the tree.

We know it is better to generate the entire tree and then to prune it (see for example [3][14]).

With C.M. criteria, one can define a straightforward pruning method (which is called C.M. pruning because it goes with using a C.M. criterion to build the tree) perfectly coherent with the building of the tree [8]. We will see below that C.M. pruning is not tied to the use of the pruned tree as a classifier and it is suitable in uncertain domains. C.M. pruning builds a new attribute binding the root of a tree with its leaves, the attribute's values corresponding to the branches leading to a leaf. It permits computation of the global quality of a tree. It corresponds to the difference between the impurity of the root of the tree and the mean impurity of its leaves, this difference being normalized to a value in [0, 1]. The global quality of a tree is equal to 1 if and only if all its leaves are pure and it is equal to 0 if and only if the frequency distributions of D in its root and in all its leaves are identical. The best sub-tree for pruning is the one that yields the highest quality pruned tree. C.M. pruning produces a family of nested trees spreading from the initial large tree to the tree restricted to its root. The curve of the global quality index as a function of the number of pruned tree gives a pragmatic method to stop the pruning process [8]. The computational cost of C.M. pruning is particularly low and it is tractable even with large databases.

The principal methods for pruning decision trees are examined in Mingers and others [17][11]. Most of these pruning methods are based on minimizing a classification error rate when each element of the same node is classified in the most frequent class in this node. These pruning methods are inferred from situations where the built tree will be used as a classifier and they systematically discard a sub-tree which doesn't improve the used classification error rate. The resulting pruned tree is different from that produced when one uses C.M. pruning. For example C.M. pruning doesn't systematically discard a sub-tree, the classification error rate of which is equal to the rate of the root. In Figure 2, D is bivalued and in each node the first (resp. second) value indicates the number of examples having the first (resp. second) value of D. This sub-tree doesn't lessen the error rate, which is 10% both in its root or in its leaves; nevertheless the sub-tree is of interest since it points out a specific population with a constant value of D while in the remaining population it's impossible to predict a value for D. The global quality index of this sub-tree is 0.55, which means that it explains 55% of the initial impurity.

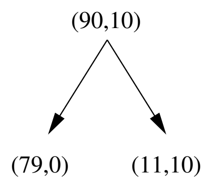

Figure 2: A tree which could be interesting although it doesn't decrease the number of errors.

2.2 The ORT criterion

In order to better confront C.M. and ORT criteria, we present now ORT criterion from the set of constraints previously cited.

Let us come back to conditions i), ii), and iii). Condition i) states that Ψ reaches its minimum value exclusively when Y induces pure sub-nodes. However, purity of all leaves is an ideal situation which is unheard-of in uncertain domains. So we can replace condition i) by another one, favouring attributes that simply separate the different values of D (the class). Thus we consider the following condition:
iv) $\Psi(Y)$ is minimum if and only if $Y(\omega) \neq Y(\omega')$ implies $D(\omega) \neq D(\omega')$.

If D has two values, Y three values and $P_1 = (1,0), P_2 = (1,0), P_3 = (0,1)$, all leaves induced by Y are pure while iv) implies $\Psi(Y) > \min \varphi$, which is not sensible. This drawback disappears if we consider attributes Y with fewer values than D. Fayyad and Irani [12] claim that grouping values of attributes and building binary trees yield better trees; we agree with them for many situations, but nevertheless, for example in medical areas, physicians do not always agree to group values since it yields meaningless trees and thus non-binary trees must not be definitively discarded. However, in this section, in order that condition iv) implies that $\Psi(Y)$ is minimum as soon as all sub-nodes are pure, we consider attributes Y with two values (this doesn't hold true for D which can still have more than two values).

There are also situations where taking into account the respective sizes of the sub-nodes is not necessary. For example, let us suppose that we try to clear up strategies for therapy in the case of a well diagnosed disease and that the choice of therapy relies on p dichotomous attributes; we can consider two methods:

1. We collect a sample of real cases where D codes the prescribed therapy and then scrutinize the tree built with this data set.

2. We consider the 2^p possible values of $(Y_1, \ldots, Y_p)$; we ask an expert to write down for each of the 2^p situations the therapy he recommends. This method yields trees which are useful tools in building expert systems or in consensus conferences.

In method 1), it is essential to take into account the respective sizes of the sub-nodes and thus to retain condition iii). In method 2), there is no reason to do so. Therefore we discard condition iii) in this section and we consider criteria satisfying ii) and iv). As frequency distributions are also elements of R^k (where k is the number of values of D), the cosine of (P_1,P_2) is the simplest function to consider. Let us note $P_1.P_2$ the canonical scalar product: $P_1.P_2 = \sum_{j=1}^{k} p_{1j}p_{2j}$ with $P_1 = (p_{11}, \ldots, p_{1k})$ and $P_2 = (p_{21}, \ldots, p_{2k})$ being the frequency distributions of D in the sub-nodes induced by Y. A selection attribute criterion is defined by the function Ψ, the minimum of which will define the best attribute to select:

$$\Psi(Y) = \cos(P_1, P_2) = \frac{P_1.P_2}{||P_1||||P_2||}$$

with $||P_i||^2 = \sum_{j=1}^{k} p_{ij}^2 \quad i = 1, 2$

This criterion is, up to a sign $-$, the ORT criterion [12] (in the ORT criterion one has to maximize $1 - \Psi(Y)$).

3 ORT versus C.M. criteria

To show the differences between C.M. and ORT criteria, let us consider the data set given in the appendix and the trees induced from this data depicted in Figure 3: a tree built with a C.M. criterion is represented at the left and the tree built with the ORT criterion at the right. ORT rapidly comes out with the pure leaf $Y_2 = y_{21}$ while C.M. criterion splits it and arrives later at the split leaves. We give here just a simple example, but some others both in artificial and real world domains are detailed in [10]: they show that ORT criterion produces more often than C.M. criteria trees with small leaves at their top. We will also see in [10] that overspecified leaves with C.M. criteria tend to be small and at the bottom of the tree (thus easy to prune) while leaves at the bottom of ORT trees can be large. Thus, in the example of therapy strategies, ORT criterion may perform better in method 2) while C.M. criteria are to be used in method 1).

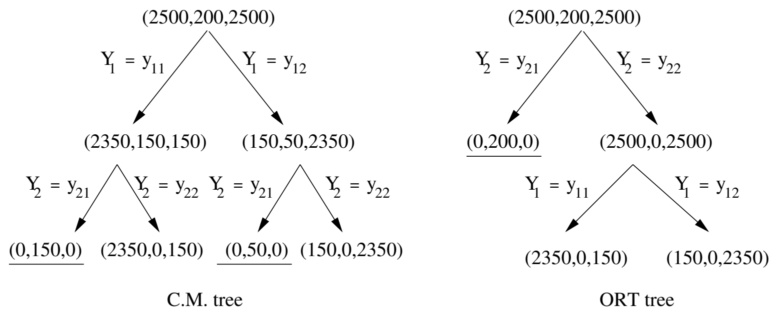

Figure 3: An example of C.M. and ORT trees.

The issue of taking into account the number of examples covered by a rule is debated in Clark and Boswell [7]. They claim rules established from few examples are unreliable in domains with uncertainty. Let d_1 and d_2 be two values of the class. They prefer for instance a rule $(R1)$ covering 1000 examples having the value d_1 and 1 example of d_2 to a rule $(R2)$ covering 5 examples having the value d_1 and 0 example of d_2. They upgrade their CN2 system in order to avoid producing highly specific rules by using the Laplacian error estimate. The accuracy of the Laplace estimate for predicting the value of D is 99.8% for R1 and 85.7% for R2. Experimental comparison on twelve real world domains shows this strategy significantly improves CN2 accuracy.

So, ORT favours attributes that simply separate the different classes without taking into account the number of examples of nodes so that ORT produces trees with small pure (or nearly pure) leaves at their top more often than C.M. criteria. Nevertheless, in uncertain domains, such leaves may be irrelevant and it is difficult to prune them without destroying the tree. However pruning is perhaps useless in the domains where the ORT criterion is preferred.

4 Lazy decision trees

4.1 Motivations

A single decision tree built from a training set is the best (according to the used attribute selection criterion) *on average* on all examples. But, there may be better attributes choices for classifying specific instances.

Let us consider the following example. Suppose a data set of patients which we would like to classify as *sick* or *healthy*. Let us assume that this data set includes a boolean attribute Human Immuno-deficiency Virus (called HIV). In this situation, if a patient has the value *true* for HIV, his class is likely to be *sick*. That means that, for this patient, the HIV attribute is highly informative for the classification task. But, even if we suppose that all instances having HIV positive are classified as *sick*, if the proportion of patients having HIV positive is small, it is likely that a C.M. tree doesn't choose this attribute at the top of the tree and the HIV-positive instances will be fragmented throughout the nodes. Using ORT criterion, we may hope that HIV attribute will be chosen at the top of the tree, which seems sensible in this situation, since this attribute gives prominence to a deterministic situation.

Nevertheless, let us note that we have to choose here between perfectly classifying few patients with a single attribute or to start explaining the class of many patients but with the cost of building a more complicated structure. Moreover, what happens if several attributes - like HIV - are highly informative each one to classify few instances? If a single tree is built, it is necessary to select one attribute at the root and the final tree will be unable to take advantage of special characteristics of an instance that may give rise to an extremely short explanation to classify it.

Lazy trees [13] have been introduced - among other things - to solve this question. The main idea of this strategy is to highlight the "best" path in a virtual decision tree for each instance to classify. A key point is to use all the available information for *each instance*. If we come back to our previous example, we know that HIV attribute seems to be a good choice at the root of the tree for patients having HIV positive. For an other instance, we will see in the next section that lazy approach may lead to select another attribute to get again a short path specific to this instance to classify it. If an instance has no value for an attribute, lazy approach will construct the best tree with remainder attributes. Thus, we will see below that it is a natural mean to handle missing values.

Next sections outline lazy trees and compare the advantages and disadvantages of this approach to those of usual decision trees built with C.M. and ORT criteria.

4.2 Overview of the method

In lazy approach, the first part of the induction process (i.e., building a classifier) is non-existent: all the work is done during the classification step.

The basic algorithm has in input a training set of labelled instances (i.e., class is known) and a new instance to classify (i.e., a test instance). It predicts in output the class value of the test instance. Aside from stop conditions, the core of the process is to select an attribute and apply it recursively on the sub-problem containing data with the same outcome of the attribute as the test instance. The overall effect is that of tracing a path in an virtual tree made specifically for the test instance.

Such an approach has been described by White [25] in order to deal with noisy data. The latter adds conditional probabilities (probabilities which are conditional on the full set of attributes) in order to relax assumptions between attributes. The process of classifying a new case proceeds by following a path through the tree from its root down to its leaf where the estimated probabilities of class membership are located. The combinatorial explosion inherent in generating large classification trees is circumvented by only building the particular path required at each stage. White estimates that this approach is suitable for all domains where there is some level of uncertainty as intrinsic to the nature of the problem.

In a more recent work, Friedman et al. [13] set lazy trees approach in relation to ordinary trees. They present a new lazy algorithm - LAZYDT - that conceptually constructs the "best" decision tree for each test instance. In practice, only a path is generated, and a caching scheme makes the algorithm fast.

The heart of the algorithm is the selection of an attribute at each recursive call. Unlike ordinary decision trees, the input of lazy approach has an extra information, the test instance, and one would like to use that information to choose the appropriate attribute. LAZYDT uses, as attribute selection criterion, a normalization of the difference between the entropies for the current node and for the node where the test instance would be branched to. This normalization avoids negative gain and allows to take into account the actual class values and not only the frequencies. By re-weighting the instances at every node, normalization leads to have equal weight for each value of the class. Furthermore, to avoid fragmentation of data, LAZYDT chooses a binary split on a single value (for attributes with more than two values, only splits on any value that is not equal to the test instance's value are allowed). According to us, normalization and binary split, which concur the aim of these stated by the ORT framework, are the cornerstone of the process which highlights quickly a deterministic situation appropriate to the test instance, even if few instances are affected.

4.3 Discussion

Unlike C.M. and ORT frameworks, which generate a single tree from a data set, lazy decision trees are able to produce the "best" tree appropriate to each instance to classify. The criterion used in LAZYDT is able to produce short paths giving prominence to deterministic situations, even with few instances. In this sense, this criterion is quite similar to ORT. But, as the criterion used in lazy trees relies on the node where the test instance will be sent, generated paths depend on the specific values of the test instance and we can hope to build the "best" path for *each* test instance: here is a difference with ORT.

In uncertain domains, missing values and noise are unavoidable and a great interest of lazy approach is its flexibility. Lazy trees use the available information so that missing values are naturally handled by considering only splits on attributes values that are known in the test instance. So, Friedman et al. [13] claim that this approach is robust with respect to missing values and doesn't need to resort to the complicated methods usually seen in induction trees (e.g., sending portions of instances down different branches like in C4.5 [20] or using surrogate attributes like in CART [3]). This method has been explicitly proposed by Brunet in [4]. Nevertheless, Ragel notes that this approach doesn't solve missing values in the training set (with LAZYDT, instances with unknown values are excluded when their values are missing for an attribute in a path). Ragel [21] proposes a pre-processing method to deal with missing values which temporarily ignored them, without deleting data. This one is based on a search of relevant associations within data to predict missing values. The core of the used algorithm stems from associations rules [1] (we will come back briefly to this method in conclusion).

But, with lazy trees, the user has to cope with multi-paths and not with a single decision tree. The latter is a graphical structured summary of a data set. Experience in induction tasks shows that this result is a key point of the success of decision trees. It is the foundations of a fruitful iterative process of discovery: building a tree brings a new light on the studied phenomenon which suggests the construction of a new tree, and so on. By this mean, the user may reach a goal that he didn't suspect. With lazy decision trees, we miss the point and its lack of intelligibility is a clear drawback. Let us note that the use of a tree as a classifier is highlighted in the artificial intelligence field. We claim that its use as an efficient data-mining description tool, oriented by an *a* priori classification of its elements, is as important as its use as a classifier.

There is also not a simple analogy between pruning in lazy decision trees and pruning in ordinary ones. According to Friedman [13], it is the weakest point of LazyDT. Experiments on the usual data sets used in the community (U. C. Irvine repository) show that the lack of pruning of lazy trees is a clear drawback with corrupted data [13]. As we have seen in Section 2.1, pruning is a fundamental point in uncertain domains. Until this question of pruning in lazy approach is not be clarified and resolved, it is obvious that it will slow down the use of lazy decision trees in such domains.

5 Conclusion

The C.M. attribute selection criteria and the C.M. pruning can be performed consistently within the same theoretical framework. It is a general framework since on the one hand there are no specific conditions on the attributes and on the other hand it doesn't rely on any specific use of the pruned tree. We have seen that C.M. pruning is suitable in uncertain domains because it allows to keep sub-trees with leaves yielding the determination of relevant decision rules, even when keeping the sub-trees doesn't increase the classification efficiency.

We can say that both ORT and C.M. criteria favour keeping in the same subset elements with the same value of the class. The C.M. criteria, taking into account the size of the nodes, favour trees with big leaves at the top and small leaves (easy to prune without destroying the tree) at the bottom. ORT might track a statistical random effect at the top of the tree rather than a characteristic of the phenomenon under study. Conversely if we know a priori that the phenomenon under study admits deterministic causes in some rare situations, the ORT criterion may find the description of these situations while C.M. criteria do not. Thus these criteria have different fields of applications. Let us remark that if we do not specialize in binary trees then C.M. criteria always work while the generalization of ORT criterion is not a simple task [15].

If classification is the main focus of the induction work, by constructing a path which takes into account characteristics of each instance, lazy decision trees is a very attractive approach. It is a suitable mean to highlight rare but interesting situations which imply few cases. In this sense, lazy decision trees are powerful than ORT criterion. But the latter, as well as the C.M. criteria framework, by offering a single decision tree, allows the user to understand the phenomenon embedded in data and has the advantage of the intelligibility. Furthermore, in uncertain domains, we know that situations with few cases may be due to random effects and that pruning is a key point: this step is not clarified in lazy decision trees yet.

To conclude, we can point out that lazy decision trees is a more sophisticated approach than ordinary decision trees: the search is dedicated for each example and not on average on all the data. This concept can be further developed: for example, association rules method [1] is a recent and efficient approach to extract all the associations of large databases. With this approach, classification task comes down to a search dedicated to each value. The counterpart to this exhaustive search is the huge amount of rules that are produced. If it is not a drawback for an automatic process, on the other hand the user has some difficulties to interpret the rules and to understand the phenomenon. Hence, there are some works on generalization of association rules [24] in order to summarize them. When such a synthesis of association rules will be available, it may be a new mean to run more effective data summaries as decision trees, and the question of the used attribute selection criterion would be obsolete.

References

[1] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. *Fast discovery of association rules*, chapter 12. Advances in Knowledge Discovery and Data Mining. AAAI/MIT Press, 1996.

[2] L. Breiman. Some properties of splitting criteria. *Machine Learning 21*, pages 41–47, 1996.

[3] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and regression trees.* Statistics probability series. Wadsworth, Belmont, 1984.

[4] T. Brunet. Le problème de la résistance aux valeurs inconnues dans l'induction: une forêt de branches. In *proceedings des* 8^{eme} *journées francophones sur l'apprentissage*, Saint-Raphaël (France), 1993.

[5] W. Buntine and T. Niblett. A further comparison of splitting rules for decision-tree induction. *Machine Learning 8*, pages 75–85, 1992.

[6] J. Catlett. Overpruning large decision trees. In *proceedings of the Twelfth International Joint Conference on Artificial Intelligence IJCAI 91*, pages 764–769, Sydney, Australia, 1991.

[7] P. Clark and R. Boswell. Rule induction with CN2: some recent improvements. In Y. Kodratoff, editor, *proceedings of European Working Sessions on Learning EWSL 91*, Lecture notes in artificial intelligence. N^o 482, pages 151–163, Porto (Portugal), 1991. Springer-Verlag.

[8] B. Crémilleux and C. Robert. A pruning method for decision trees in uncertain domains: Applications in medicine. In *proceedings of the workshop Intelligent*

Data Analysis in Medicine and Pharmacology, ECAI 96, pages 15–20, Budapest, Hungary, 1996.

[9] B. Crémilleux and C. Robert. A theoretical framework for decision trees in uncertain domains: Application to medical data sets. In E. Keravnou, C. Garbay, R. Baud, and J. Wyatt, editors, *6th Conference on Artificial Intelligence In Medicine Europe (AIME 97)*, Lecture notes in artificial intelligence. N^o 1211, pages 145–156, Grenoble (France), 1997. Springer-Verlag.

[10] B. Crémilleux, C. Robert, and M. Gaio. Uncertain domains and decision trees: Ort versus c.m. criteria. In *7th Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems*, pages 540–546, Paris (France), 1998. EDK.

[11] F. Esposito, D. Malerba, and G. Semeraro. Decision tree pruning as search in the state space. In P. B. Brazdil, editor, *proceedings of European Conference on Machine Learning ECML 93*, Lecture notes in artificial intelligence. N^o 667, pages 165–184, Vienna (Austria), 1993. Springer-Verlag.

[12] U. M. Fayyad and K. B. Irani. The attribute selection problem in decision tree generation. In *proceedings of Tenth National Conference on Artificial Intelligence*, pages 104–110, Cambridge, 1992. MA: AAAI Press/MIT Press.

[13] J. H. Friedman, R. Kohavi, and Y. Yun. Lazzy decision trees. In *proceedings of the thirteenth National Conference on Artificial Intelligence AAAI 96*, pages 717–724, Portland (Oregon), 1996.

[14] S. B. Gelfand, C. S. Ravishankar, and E. J. Delp. An iterative growing and pruning algorithm for classification tree design. *IEEE Transactions on Pattern Analysis and Machine Intelligence 13(2)*, pages 163–174, 1991.

[15] G. H. John. *Enhancements to the data mining process*. PhD thesis, Stanford University, 1997.

[16] J. Lorigny. Questionnaire theory applied to wording recognition. In *proceedings of the IEEE Conference on Information Theory*, volume 29, pages 67–82, Les Arcs (France), 1982. CNRS.

[17] J. Mingers. An empirical comparison of pruning methods for decision-tree induction. *Machine Learning 4*, pages 227–243, 1989.

[18] T. Niblett. Constructing decision trees in noisy domains. In *proceedings of 2nd European Working Sessions on Learning EWSL 87*, pages 67–78, Bled (Yugoslavia), 1987. Sigma Press.

[19] J. R. Quinlan. Induction of decision trees. *Machine Learning 1*, pages 81–106, 1986.

[20] J.R Quinlan. *C4.5 Programs for machine learning*. Morgan Kaufmann, San Mateo, Californie, 1993.

[21] A. Ragel. Preprocessing of missing values using robust associations rules. In J. M. Zytkow and M. Quafafou, editors, *proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery PKDD 98*, Lecture notes in artificial intelligence. N^o 1510, pages 414–422, Nantes (France), 1998. Springer-Verlag.

[22] S. R. Safavian and D. Landgrebe. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics 21(3)*, pages 660–674, 1991.

[23] C. Schaffer. Overfitting avoidance as bias. *Machine Learning 10*, pages 153–178, 1993.

[24] R. Srikant and R. Agrawal. Mining generalized association rules. In *proceedings of the 21st International Conference on Very Large Data Bases VLDB 95*, pages 407–419, 1995.

[25] A. P. White. Probabilistic induction by dynamic path generation in virtual trees. In M. A Bramer, editor, *proceedings of the 6th conference Expert Systems ES 86*, pages 35–46, Brighton (Great Britain), 1987. Cambridge University Press.

Appendix

We give here the data file used to build trees for Figure 3. Let us recall that D denotes the class that we want to explain, and $Y_1, \ldots, Y_p$ are the attributes.

	D	Y_1	Y_2
1	d_1	y_{11}	y_{22}
$\ldots$	$\ldots$	$\ldots$	$\ldots$
2350	d_1	y_{11}	y_{22}
2351	d_1	y_{12}	y_{22}
$\ldots$	$\ldots$	$\ldots$	$\ldots$
2500	d_1	y_{12}	y_{22}
2501	d_2	y_{11}	y_{21}
$\ldots$	$\ldots$	$\ldots$	$\ldots$
2650	d_2	y_{11}	y_{21}
2651	d_2	y_{12}	y_{21}
$\ldots$	$\ldots$	$\ldots$	$\ldots$
2700	d_2	y_{12}	y_{21}
2701	d_3	y_{11}	y_{22}
$\ldots$	$\ldots$	$\ldots$	$\ldots$
2850	d_3	y_{11}	y_{22}
2851	d_3	y_{12}	y_{22}
$\ldots$	$\ldots$	$\ldots$	$\ldots$
5200	d_3	y_{12}	y_{22}

Table 1: Data file used to induce decision trees of Figure 3.