Using Impurity and Depth for Decision Trees Pruning

D. Fournier and B. Crémilleux

GREYC, CNRS - UPRESA 6072
Université de Caen
F-14032 Caen Cédex France
{fournier,cremilleux}@info.unicaen.fr

Abstract

Most pruning methods for decision trees minimize a classification error rate. In uncertain domains, some sub-trees which do not lessen the error rate can be relevant to point out some populations of specific interest or to give a representation of a large data file. We propose here a new pruning method (called DI pruning) which takes into account the complexity of sub-trees and which is able to keep sub-trees with leaves yielding to determinate relevant decision rules, even when keeping these ones does not increase the classification efficiency.

1 Introduction

Data mining and Knowledge Discovery in Databases (KDD) are fields of increasing interest combining databases, artificial intelligence, machine learning and statistics. Broadly speaking, the purpose of KDD is to extract from large amounts of data non trivial "nuggets" of information in an easily understandable form. Such discovered knowledge may be for instance regularities or exceptions.

In this context, with the growth of databases, methods which explore data - like for example decision trees - are required. Such methods can give a summary of the data (which is easier to analyze than the raw data) or can be used to build a tool (like for example a classifier) to help a user for many different decision makings.

Briefly, 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 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) [1] 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 [9] and CART [1].

Nevertheless, in many areas, like in medicine, data are uncertain: that means that there are always some examples which escape from the rules. Translated in the context of decision trees, that means these examples seem similar but in fact differ from their classes. In these situations, it is well-known (see [1], [3]) that decision tree algorithms tend to divide nodes having few examples and that the resulting trees tend to be very large and overspecified. Some branches, especially towards the bottom, are present due to sample variability and are statistically meaningless (one can also say that they are due to noise in the sample). Pruning methods (see [1], [8], [9]) try to cut such branches in order to avoid this drawback.

In uncertain domains, the understanding of the mechanism of these methods is a key point for their use in practice: in order to achieve a fruitful process of extraction of information, these methods requires declarativity during treatments. We will see in Section 3 that we include this point in our pruning strategy.

This paper is organized as follows. Section 2 outlines existing decision trees pruning methods and sets out the question of pruning in uncertain domains. We will see that the principal pruning methods are based on a classification error rate and that it may be a drawback in uncertain domains. So, in Section 3, we propose a quality index (called DI for Depth-Impurity quality index) which is a trade-off between the depth and the impurity of nodes of a tree. From this index, we infer a new pruning method for decision trees (denoted DI pruning) which is appropriate in uncertain domains: unlike usual methods, this method is not bound to the possible use of a tree for classification. It is able to give an efficient description of a data file oriented by a priori classification of its elements and to highlight interesting sub-populations, even if the classification error rate does not decrease. We think that it is a major point for the use of decision trees in uncertain domains. In Section 4, we present examples in a real world domain where we use DI pruning as a tool to optimize a final decision tree.

2 Motivations

The principal methods for pruning decision trees are examined in [6], [8] and [10]. 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. The latter is estimated with a test file or using statistical methods such as cross-validation or bootstrap.

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. Let us consider the sub-tree depicted in Fig. 1. D is the class and it is here bivalued. 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.

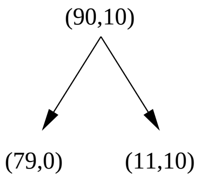

Figure 1: A tree which could be interesting although it doesn't decrease the number of errors.

Our aim is to propose a pruning method which does not systematically discard a sub-tree which classification error rate is equal to the rate of the root. The following section describes a quality index of a tree which is able to highlight such interesting - in our opinion - sub-populations. This method is not based on a classification error rate and we claim that it is a major point in uncertain domains.

3 Depth-Impurity quality index and pruning

3.1 Framework for a quality index

Let us formulate the question of a quality index. We claim that the quality index of a tree T has its maximum value if and only if the two following conditions are satisfied:

i) All the leaves of T are pure.

ii) Depth of T is 1.

We notice that these conditions are part of the usual framework to properly define suitable attribute selection criteria to build decision trees ([1], [4]). This framework states that a theoretical level, criteria derived from an impurity measure ([7]) perform comparably ([1], [2] and [4])[1]. Such criteria are called 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 [11]) and the Gini criterion (in CART algorithms [1]), are C.M. criteria.

The idea is then to use known properties (based on an impurity measure) of C.M. criteria to define a quality index. In order to better understand the genesis of the quality index that we present below, we have to glance at the question of attribute selection criteria in decision trees and its usual notations. Let us call D the class of the examples of a data set, with values $d_1, \ldots, d_k$, Ω a node and Y any attribute defined on Ω, with values $y_1, \ldots, y_m$. Let us note Ψ a C.M. criterion (see Fig. 2).

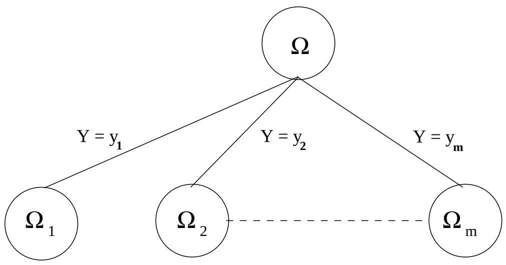

Figure 2: Splitting of a node Ω using an attribute Y.

We know (see [1], [7] and [4]) that $\Psi(Y) = \sum_i^m \alpha_i \varphi(\Omega_i)$ where $\Omega_1, \ldots, \Omega_m$ are the sub-nodes

[1] We used here the term of impurity because it is the most usual in the machine learning community ([1] [7]).

yielded by Y, α_i is the rate of Ω_i in Ω, and φ is an impurity measure of D. $\Psi(Y)$ can be viewed as a combined measure of impurity of the sub-nodes induced by Y. The value of the criterion in a node reflects how appropriately the chosen attribute divides the data: it is a way to quantify the quality of a tree of depth 1. For example, theoretical results on C.M. criteria claim that the minimum value of $\Psi(Y)$ is reached if and only if the sub-nodes induced by Y are pure with respect to D, that $\Psi(Y)$ has its maximum value if and only the frequency distributions of D in Ω and in the sub-nodes induced by Y are equal.

In fact, these criteria, used to quantify the quality of a splitting (that means of a tree of depth 1) can be straightforwardly extended in order to evaluate the quality of a tree of any depth: the sub-nodes of depth 1 induced by Y are replaced by the leaves of any depth of the tree. Furthermore, we shall not forget that our aim is to offer a suitable pruning method in uncertain domains. In this context, we claim that a deep tree is less relevant than a small: the deeper a tree, the less understandable and reliable. For example, in Fig. 3, we prefer the tree which has the root denoted Ω_3 to the one with root Ω_{12} (even if the frequency distributions of D are the same on all leaves). On the other hand, both trees with root Ω_3 and Ω_4 have the same number of miss-classified examples, but we prefer the tree with root Ω_3 because it is simpler.

So, we have to properly define a quality index which takes into account both impurities and depths of leaves. The next section presents this index.

3.2 The quality index DI

We propose here a quality index called DI (for Depth-Impurity) which corresponds to a trade-off between depth and impurity of each leaf of a tree.

Equation 1 defines the quality index DI (for Depth-Impurity) of a tree T (we note either $DI(T)$ or $DI(\Omega)$, the quality index of the tree T of root Ω):

$$DI(T) = \sum_{i}^{m} \alpha_i (1 - \varphi(\Omega'_i)) f(depth(\Omega'_i)) \quad (1)$$

where $\Omega'_1, \ldots, \Omega'_m$ are the leaves of T, α_i is the rate of Ω'_i in Ω, and φ is an impurity measure normalized between $[0,1]$. Let us note that as φ is an impurity measure, $(1-\varphi)$ defines a purity measure. The depth of a leaf is computed from the root of the whole tree (and not from the root of T). For instance, in Fig. 3, to calculate the quality of tree of root Ω_3, the depth of the leaf Ω'_7 is with respect Ω_1. If we come back to the example in Fig. 3 given at the end of the previous section (comparison between trees of root Ω_3 versus the one of root Ω_{12}), we notice that the quality index of the tree of root Ω_{12} is lower because it is deeper. This choice to compute the depth allows comparisons between any nodes. By introducing a damping function (denoted f), DI is able to take into account the depth of the leaves: the deeper a leaf, the lower its quality.

We are led to address now the question of defining the damping function f. The argument of f is a depth of a node (denoted d). A minimal set of constraints is:

1 f is decreasing.

2 $f(1) = 1$

Constraint 1 corresponds to the damping process and 2 is necessary to satisfy condition **ii)** of our framework. If we choose an impurity measure normalized between $[0,1]$, as $\sum_i^m \alpha_i = 1$, we will see immediately with these two constraints that the value of DI is between $[0,1]$. The higher the value of DI, the better quality of T.

Furthermore, we add the three following constraints:

3 $f(d) \simeq 0$ when d tends towards the total number of attributes.

4 $0 \leq f(d) \leq 1$

5 $f(d+1) = \beta.f(d)$ where $\beta \in \mathbb{R}^+$ (in practice, $\beta \in [0,1[$ to respect constraint 1)

Constraint 3 means that a leaf which has a depth similar to the total number of attributes is likely to be unreliable, so it seems sensible that the value of its quality is close to the minimum value of DI. Constraint 4 allows the values of the damping function and of the purity (or impurity) of a leaf to have same rough estimates. Over the achievement of a linear damping, constraint 5 is suggested by algorithmic considerations: we will see that it allows to have a linear computation of DI as indicated by the remark below. So, as the number of elementary steps in the whole process is limited, the computational cost of DI pruning (see Section 3.3) is particularly low and it is tractable even with large databases. Moreover, the choice of β does not modify the result of the pruning process (see Section 3.3).

Translated into a mathematical form, this set of constraints leads to choose an exponential function for f.

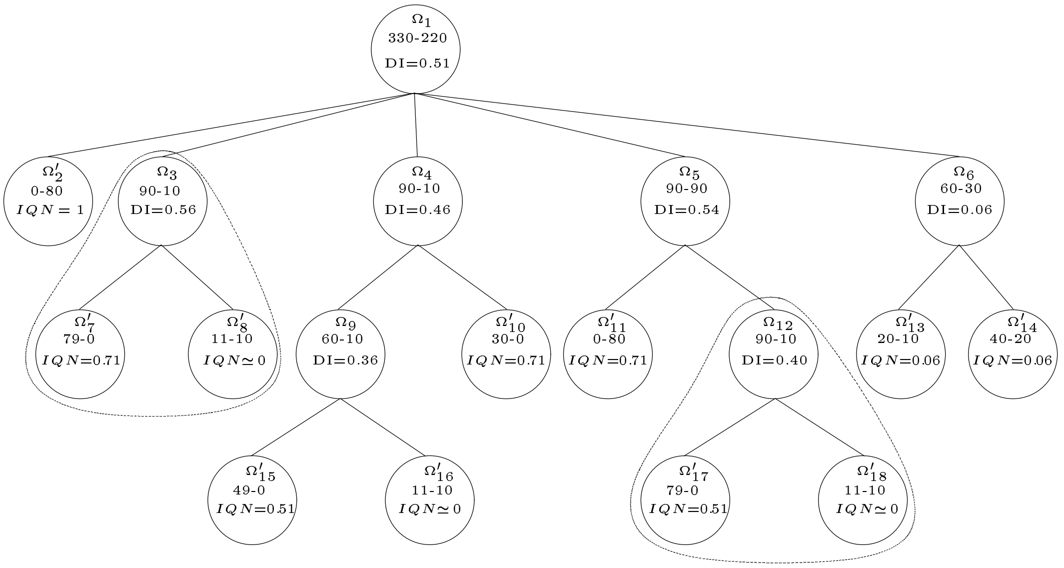

Figure 3: Examples of DI values

Proof. From $f(d+1) = \beta.f(d)$, we deduce: $f(d) = \beta^{(d-1)}.f(1)$. We thus get from 2: $f(d) = \beta^{(d-1)}$. This equality implies that f is an exponential function.

We thus choose the following function f:

$$f(d) = e^{-\frac{d-1}{N}} \tag{2}$$

where N is the the total number of attributes (let us note that any exponential function can be chosen).

If we come back to the tree given in Fig. 3, we notice that the tree which has the root denoted Ω_3 has a greater DI value than the one with root Ω_4: impurities of leaves of these trees are equal, but the latter is more complex than the former. This has been taken into account by DI.

Remark: with regard to the algorithmic point of view, computation of DI is not expensive. We can easily write $DI(T)$ according to the DI values of sons of T: in other words, the computation of $DI(T)$ is the weighted average of the DI values of the sons of T.

Proof. As used previously, $\Omega'_1, \ldots, \Omega'_m$ are the leaves of a tree T of root Ω. Let us call Ω_k a son of a node Ω. We have:

$$DI(\Omega_k) = \sum_j \alpha_k(\Omega'_j)(1-\varphi(\Omega'_j))e^{-\frac{depth(\Omega'_j)-1}{N}} \tag{3}$$

$$\text{where } \alpha_k(\Omega'_j) = \frac{\alpha(\Omega'_j)}{\alpha(\Omega_k)} \tag{4}$$

With (4), (3) becomes :

$$DI(\Omega_k) = \sum_j \frac{\alpha(\Omega'_j)}{\alpha(\Omega_k)}(1-\varphi(\Omega'_j))e^{-\frac{depth(\Omega'_j)-1}{N}} \tag{5}$$

It follows easily that the definition of DI (see Equation 1) can be rewritten:

$$DI(\Omega) = \sum_k \alpha(\Omega_k)DI(\Omega_k) \tag{6}$$

By ensuring that DI is computed for each node in one step, this point will allow a low computational cost for the DI pruning method. We will now move to this subject.

3.3 DI pruning

DI leads to a straightforward way to prune sub-trees: the main idea is to compare $DI(T)$ with the purity of its root. When $DI(T)$ is lower than the purity of its root, we infer that the "cost" of T (due to its complexity) is less useful than the benefit of the explanation generated by T. So, T is cut and replaced by its root which becomes a leaf of the original tree. The cost-complexity method of Breiman [1] uses a similar process (that means a trade-off between a "cost" and a "benefit" to cut sub-trees), but the cost is based on a classification error rate.

So, a straightforward pruning method (that we call DI pruning because it goes with DI) consists in pruning all sub-trees T of root Ω of the original tree which satisfy the following condition:

$$DI(T) \leq 1 - \varphi(\Omega) \tag{7}$$

From experiments, we noticed that the degree of pruning is quite bound to the uncertainty embedded in data. In practice, that means the damping process has to be adjusted according to data in order to obtain, in all situations, a relevant number of pruned trees. For that, we introduce a parameter (denoted k) to control the damping process: the higher k, the more extensive the pruning stage (i. e. the more sub-trees are cut). Equation (8) gives the damping function updated by this parameter (as usual, N is the total number of attributes):

$$f_k(d) = e^{-\frac{k(d-1)}{N}} \text{ with k } \in \mathbb{R}^+ \tag{8}$$

With $k = 1$, we find again Equation 2. By varying k, DI pruning produces a family of pruned trees spreading from the initial large tree to the tree restricted to its root. In practice, it should be not easy to select automatically the "best" pruned tree (but it is not the main aim of this stage of this work). Nevertheless, curves of DI as a function of k and as a function of the number of pruned nodes give a pragmatic method to stop the pruning process. Furthermore, if we are eager to obtain automatically a single "best" pruned tree, we can use a procedure requiring a test file [1] [10] and the final tree should be the one which maximizes DI (we will indicate this way in Conclusion).

Fig. 4 shows the pruned tree obtained (with $k = 1$) from the whole tree indicated in Fig. 3. Two sub-trees (of root Ω_4 and Ω_{12}) have been cut. Although the sub-trees of root Ω_3 and Ω_{12} are identical, only the latter is cut because it is deeper than the former. Moreover, let us note that the frequency distributions of D in Ω_3 and in Ω_4 are equal, the sub-tree of root Ω_4 is removed (and *not* the sub-tree of root Ω_3), because Ω_4 is more complex than Ω_3.

4 Experiments

We have designed an induction software called UnDeT (for Uncertain Decision Trees) which produces decision trees. Two paradigms of attribute selection criteria are available in UnDeT (the choice of one of these depends on the kind of data and the aim wished by the user: see [5]). UnDeT indicates DI quality index of each node and prunes trees with DI pruning. UnDeT is available on the web at the following address: "http://cush.info.unicaen.fr/UnDeT/". UnDeT is included in a more general data mining tool-box in which another major part is devoted to the treatment of missing values [12].

In this section we describe the results obtained by running UnDeT on a real world database. We perform UnDeT with the gain criterion (Shannon entropy) [11] because it is the most commonly used. The data set is a medical database coming from the University Hospital at Grenoble (France) and runs on child's meningitis.

4.1 Child's meningitis

Faced with a child with a case of acute meningitis, the clinician must quickly decide which medical course should be taken. Briefly, the majority of these cases are viral infections for which a simple medical supervision is sufficient, whereas about one quarter cases are caused by bacteria and need treatment with suitable antibiotics.

In typical cases, the diagnosis could be considered as obvious and a few simple rules enable a quasi certain decision. However, nearly one third of these cases are presented with non-typical clinical and biological data: the difficulty of the diagnosis lies in the fact that the observed attributes, considered separately, have little diagnostic signification. The aim of this section is to study the relevance of DI pruning in such domains.

The used data set is composed of 329 instances, described by 23 (quantitative or qualitative) attributes. The class is bivalued (viral versus bacterial).

4.2 Results

The initial large tree (see Fig. 5) has 29 nodes with pure leaves. Its quality (0.848) is high, especially for a medical domain. This result is due to the relevance of the used attributes.

On the tree depicted in Fig. 5, let us call T_4 the sub-tree of root labelled `-4-` and T_{10} the sub-tree of root labelled `-10-`. T_4 has an impurity in its root equal to 0.54 (2 miss-classified instances among 16)

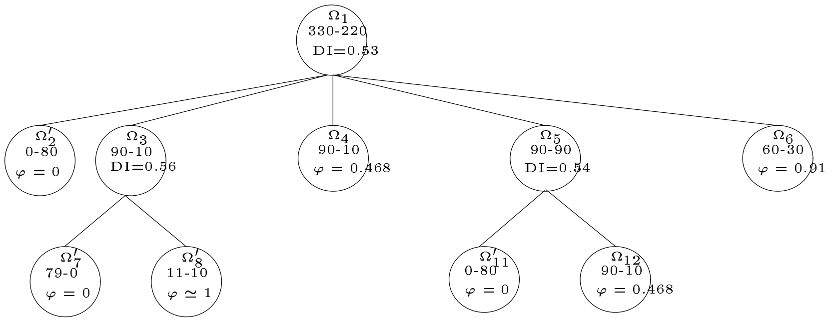

Figure 4: Examples of DI values

```
 DI=0.848 -1- (30/245) gram <= 0:  DI=0.818
| -3- (16/243) prot <= 0.95:  DI=0.826
| | -5- (9/223) purpura = 0:  DI=0.818
| | | -10- (7/24) age <= 1.66:  DI=0.722
| | | | -12- (0/14) vs <= 78:  I=0 Class = vir
| | | | -13- (7/10) vs > 78:  DI=0.631
| | | | | -14- (0/7) age <= 0.58:  I=0 Class = vir
| | | | | -15- (7/3) age > 0.58:  DI=0.514
| | | | | | -16- (1/3) fever <= 39.7:  I=0.811 Class = vir
| | | | | | -17- (6/0) fever > 39.7:  I=0 Class = bact
| | | -11- (2/199) age > 1.66:  DI=0.832
| | | | -18- (0/119) lung = 0:  I=0 Class = vir
| | | | -19- (0/54) lung = 1:  I=0 Class = vir
| | | | -20- (0/20) lung = 2:  I=0 Class = vir
| | | | -21- (2/6) lung = 3:  DI=0.797
| | | | | -22- (0/1) season = winter:  I=0 Class = vir
| | | | | -23- (2/0) season = spring:  I=0 Class = bact
| | | | | -24- (0/5) season = summer:  I=0 Class = vir
| | -6- (1/10) purpura = 1:  DI=0.873
| | | -26- (0/10) cytol <= 400:  I=0 Class = vir
| | | -27- (1/0) cytol > 400:  I=0 Class = bact
| | -7- (0/10) purpura = 2:  I=0 Class = vir
| | -8- (4/0) purpura = 3:  I=0 Class = bact
| | -9- (2/0) purpura = 4:  I=0 Class = bact
| -4- (14/2) prot > 0.95:  DI=0.685
| | -28- (2/2) cytol <= 600:  I=1 Class = bact
| | -29- (12/0) cytol > 600:  I=0 Class = bact
-2- (54/0) gram > 0:  I=0 Class = bact
```

Figure 5: Initial large tree

and T_{10} has an impurity in its root equal to 0.77 (7 miss-classified instances among 31). Nevertheless, $DI(T_{10})$ (which equal to 0.722) is higher than $DI(T_4)$ (0.685). Furthermore, T_{10} is deeper than T_4, which also penalized the quality of T_{10}. $DI(T_{10})$ is better, because it better explains instances (T_{10} finally leads to a single miss-classified instance), but T_{10} is complex (we will come back below to this point with the pruning stage).

Fig. 6 represents the pruned tree with $k = 1$. The sub-tree of root labelled `-11-` becomes a leaf. This pruning introduces 2 miss-classified instances. This number is not high regarding the 201 instances of the node. Furthermore, in order to properly classify these 2 instances, the initial large tree had to build a complex sub-tree with deep leaves. Such leaves appear not to be very reliable in uncertain domains.

```
 DI=0.829
-1- (30/245) gram <= 0:  DI=0.796
| -3- (16/243) prot <= 0.95:  DI=0.803
| | -5- (9/223) purpura = 0:  DI=0.792
| | | -10- (7/24) age <= 1.66:  DI=0.722
| | | | -12- (0/14) vs <= 78:  I=0 Class = vir
| | | | -13- (7/10) vs > 78:  DI=0.631
| | | | | -14- (0/7) age <= 0.58:  I=0 Class = vir
| | | | | -15- (7/3) age > 0.58:  DI=0.514
| | | | | | -16- (1/3) fever <= 39.7:  I=0.811 Class = vir
| | | | | | -17- (6/0) fever > 39.7:  I=0 Class = bact
| | | -11- (2/199) age > 1.66:  I=0.080 Class = vir
| | -6- (1/10) purpura = 1:  DI=0.873
| | | -26- (0/10) cytol <= 400:  I=0 Class = vir
| | | -27- (1/0) cytol > 400:  I=0 Class = bact
| | -7- (0/10) purpura = 2:  I=0 Class = vir
| | -8- (4/0) purpura = 3:  I=0 Class = bact
| | -9- (2/0) purpura = 4:  I=0 Class = bact
| -4- (14/2) prot > 0.95:  DI=0.685
| | -28- (2/2) cytol <= 600:  I=1 Class = bact
| | -29- (12/0) cytol > 600:  I=0 Class = bact
-2- (54/0) gram > 0:  I=0 Class = bact
```

Figure 6: Pruned tree ($k = 1$)

Finally, Fig. 7 indicates the pruned tree with $k = 2$. The sub-tree of root labelled `-5-` becomes a leaf (in this case, the sub-tree T_{10} is cut: its complexity to classify a single instance is not dependable).

It is important to notice that the sub-tree of root labelled `-4-` is *not* destroyed: even if it does not decrease the number of miss-classified examples (which is 2 on both root and leaves), this sub-tree highlights a reliable sub-population when the attribute "cytol" is higher than 600 (this result is checked by the medical expert). It is typically the situation that we presented in our motivations (see Sect. 2).

```
 DI=0.762
-1- (30/245) gram <= 0:  DI=0.716
| -3- (16/243) prot <= 0.95:  DI=0.718
| | -5- (9/223) purpura = 0:  I=0.237 Class = vir
| | -6- (1/10) purpura = 1:  DI=0.873
| | | -26- (0/10) cytol <= 400:  I=0 Class = vir
| | | -27- (1/0) cytol > 400:  I=0 Class = bact
| | -7- (0/10) purpura = 2:  I=0 Class = vir
| | -8- (4/0) purpura = 3:  I=0 Class = bact
| | -9- (2/0) purpura = 4:  I=0 Class = bact
| -4- (14/2) prot > 0.95:  DI=0.685
| | -28- (2/2) cytol <= 600:  I=1 Class = bact
| | -29- (12/0) cytol > 600:  I=0 Class = bact
-2- (54/0) gram > 0:  I=0 Class = bact
```

Figure 7: Pruned tree ($k = 2$)

5 Conclusion

We have proposed a quality index DI for decision trees for uncertain domains which realizes a trade-off between the impurities and the depth of leaves of a tree. Stemming from DI, we present a pruning method which is able to keep sub-trees which do not lessen an error rate but which can point out some populations of specific interest; yet usual methods are based on a classification error rate. We claim that it is a major point in uncertain domains.

Further work has to normalize $DI(T)$ by taking into account the number of instances of T with regard to the root of the whole tree: this improvement will allow to compare properly trees with same frequency distributions of D in the nodes but different number of instances. Another direction is to use a test file to improve the choice of the "best" pruned tree. We will move then to an other stage of our work, which will be to select a pruned tree which reflects - *in general* - the sound knowledge of the studied data.

Acknowledgements

The authors wish to thank Dr P. François (University Hospital of Grenoble, France) for providing data on child's meningitis and many valuable comments.

References

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

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

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

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

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

[10] J. R. Quinlan. Simplifying decision trees. *International Journal of Man-Machine Studies*, 27:221–234, 1987.

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

[12] A. Ragel and Crémilleux B. Mvc - a preprocessing method to deal with missing values. *Knowledge-Based Systems*, pages 285–291, 1999.