Explaining the result of a Decision Tree to the End-User

Isabelle Alvarez [1] [2]

Abstract. This paper addresses the problem of the explanation of the result given by a decision tree, when it is used to predict the class of new cases. In order to evaluate this result, the end-user relies on some estimate of the error rate and on the trace of the classification. Unfortunately the trace doesn't contain the information necessary to understand the case at hand. We propose a new method to qualify the result given by a decision tree when the data are continuous-valued. We perform a geometric study of the decision surface (the boundary of the inverse image of the different classes). This analysis gives the list of the tests of the tree that are the most sensitive to a change in the input data. Unlike the trace, this list can easily be ordered and pruned so that only the most important tests are presented. We also show how the metric can be used to interact with the end-user.

1 INTRODUCTION

Real-world applications of decision trees (DT) are used as decision support system in various domain [13]. DT algorithms are also integrated in software for data mining or decision support purpose (see for instance software lists on http://www.kdnuggets.com or http://www.mlnet.org/). They often offer many possibilities to build, prune, manipulate or validate decision trees. However, when it comes to the final use of the DT, to classify real cases and make a decision, end-users find little information to assess the relevance of the result. This kind of information is generally available by the mean of error rates or probability estimators. [4] [11] [7]. In practice, these estimators are not always available, since they are developed for the construction of the tree and not for the end-user's need (see examples in [15] and [6]). They are also not necessarily accurate ([11]; [10]). Besides, little information is developed to help the end-user to link the result to the input data, to assess the relevance of the result. Actually, it's a difficult problem, since it depends on both the user and the system. Works on tree intelligibility are an attempt to answer this question. This is done mainly by pruning methods (see [8] for a review). Works on feature selection (see [20]) contribute also to this objective. It is also one of the main objectives of fuzzy DT [17]. But with these methods, intelligibility is sought for the tree itself, considered as a model. The relevance of a particular result is only available by the mean of the trace of the classification, that is the path followed in the tree, the list of the tests passed by the case from the root to the leaf that finally gives the class.

Unfortunately the trace doesn't hold the right information that is necessary to understand the situation of a case. The change of some tests that are in the trace can have no consequences on the result. Conversely, a little change of the value of an attribute that doesn't appear in the the trace can lead to a modification of the resulting class. The fact is that the trace doesn't exploit the information that is embedded in the partition realized by the DT in the input space.

We propose a geometric method to take into account the complete partition of the input space, when it possible to define a metric. This method is based on the study of the decision surface (DS), that is the boundary of the inverse image of the different classes in the input space. We consider that the position of a case relatively to the DS can give a good description of the situation to the end-user. It allows to identify the tests of the DT that are the most sensitive to a change in the input data. Contrary to the trace, this list of tests is relevant to explain the particular classification of a case, since if the tests of the list aren't verified any more, the class changes.

The paper is organized as follow: Section 2 presents the drawbacks of the trace as an explanation support. Simple geometric examples show why they cannot be bypassed by any processing of the trace. The same examples suggest a geometric method to identify more relevant tests to describe the situation of a case. Section 3 presents the geometric sensitivity analysis method, some interesting properties of the sensitive tests (uniqueness, robustness, ordering relation) and general results. Section 4 focuses on one example and studies the role of the metric. Possible complementary viewpoints are discussed in the concluding section.

2 LIMITS OF THE TRACE AS AN EXPLANATION SUPPORT

Software that integrates decision trees algorithms generally allows the user to visualize the trace of the classification of a new case. But it is not easy to read, all the more so as it grows in size. Moreover it has similar drawbacks to the trace of reasoning in rule based system (see [5]), since it is easy to translate a decision tree into an ordered list of rules (by following every path from the root to the different leaves). In fact, works on trace of reasoning finally directed toward reconstructive explanation [14], [18], [9]. The following examples illustrate why the trace cannot be used to provide to the end-user relevant information about the case. We consider binary linear decision trees (LDT): a test consists in computing the algebraic distance h of a new case (the point P) to a hyperplane H. The point P passes the test depending on the sign of $h\,(P, H)$. So the area classified by a leaf is the intersection of halfspaces $E(H)$. The tree induces a partition of the input space, and we call decision surface the union of the boundaries of the different areas corresponding to the different classes. In the case of LDT, it consists in pieces of hyperplanes. Figures 2 and 3 show examples of partitions induced by the trees in Figure 1. We consider the trace of the classification given by the trees for several points. DT1 classifies P_1 at the first test, so the trace of the classification of P_1 is limited to $h(P_1, H_1) \leq 0$. But we can see in Figure 2 and 3 that if P_1 moves slightly and crosses the hyperplane H_1 so that $h(P_1, H_1) > 0$, its class remains the same. The fact is that H_1

[1] LIP6, Paris VI University, Paris, France email: isabelle.alvarez@lip6.fr
[2] Cemagref, Aubière, France

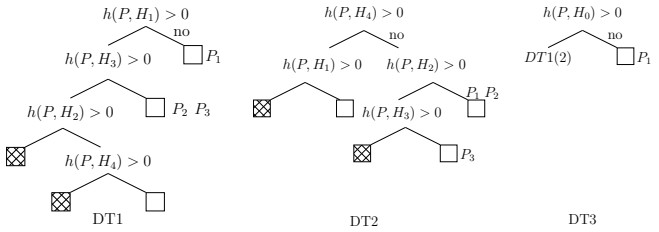

Figure 1. Linear decision trees (LDT)

contributes to the definition of the decision surface in a place very far from P_1. The case of DT3 is worse: the only hyperplane involved in the trace of the classification of P_1, H_0, doesn't contribute to the definition of the DS. In these cases, the trace of the classification says something true about P_1, but it is quite useless to describe why a particular class (among all the possible classes) was assigned to this point. In the case of point P_3, all the hyperplanes except H_1 are involved in the trace of its classifications by DT2 and DT3. However, if we consider a small neighborhood of P_3, the hyperplane H_3 is the only one necessary to describe the situation: We can see in Figure 2 and 3 that in the open ball B_3 centered at P_3, H_3 separates the classes, so the sign of $h(P, H_3)$ determines the class of P. But this information is lost in the trace. Conversely the trace of classification of point P_2 by DT1 involves H_1 and H_3. H_1 is not relevant to describe the situation, for the same reason as for point P_1, and if we consider a small neighborhood of P_2, we can see that H_3 is not sufficient to describe the situation. P_2 verifies the test $h(P_2, H_3) \leq 0$. But we can find near P_2 an area of points that don't verify this test but still have the same class as P_2. So the situation is not adequally described by the trace (the same problem arises with the trace of DT2). In fact we can see that in the open ball B_2 centered at P_2, the decision surface involves both H_2 and H_3. Both tests (the sign of $h(P, H_2)$ and of $h(P, H_3)$) have to be considered to classify the points in B_2. But the trace doesn't contain this information.

These simple geometric examples show that the trace doesn't necessarily contain the tests that are meaningful to describe the situation of a case in its neighborhood. In two of the traces of P_1, there is none. In the traces of P_2 and P_3, only one test is important, morover in the case of P_2, another essential test is always missing. Since DT1 and DT2 are well-pruned trees regarding both partitions of Figures 2 and 3, this problem cannot be solved by the pruning phase (although it can be made worse, as in DT3). The underlying reason is that when a class area is not convex, it is divided into several leaves. So the boundaries of a leaf are not always meaningful with regards to the decision. It is then necessary to refer to the decision surface itself to find the relevant elements of description of a case. Besides, in a small neighborhood of P_3, the classes are separated by H_3. This hyperplane is the closest part of the decision surface to P_3. Similarly, H_2 and H_3 separates the classes in a neighborhood of P_2 (and P_1). In fact the point M on the intersection of H_2 and H_3 is the closest point of the decision surface to P_1 and P_2. The preceding examples suggest that sensitivity analysis can identify the tests that describe the situation in the neighborhood of a case.

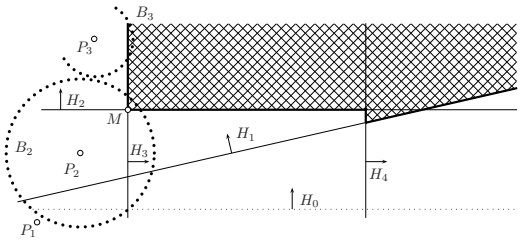

Figure 2. A partition associated to the linear trees in Figure 1. The decision surface is materialized with bold lines

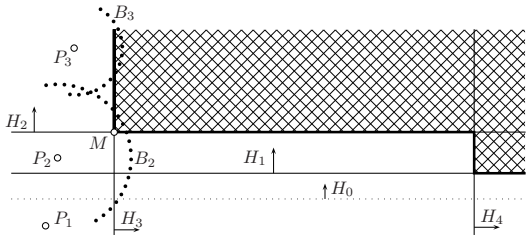

Figure 3. Axis-parallel partition corresponding to the trees in Figure 1

3 PROJECTING INPUT DATA ONTO THE DECISION SURFACE

3.1 Sensitive tests

In the case of LDT, the decision surface Γ is piecewise affine since it consists in pieces of hyperplanes. At each point y of Γ, the decision surface is defined by a list of hyperplanes, often reduced to a single hyperplane. For instance, in Figure 2 and 3, the decision surface is either on one of the hyperplanes H_i, $1 \leq i \leq 4$, either on the intersections of H_2 and H_3; H_2 and H_4; or H_4 and H_1.

Let $x \in E$ be a point of the input space. If E is a complete metric space, $d(x, \Gamma)$ is reached on at least one point $p(x)$ for which we have $d(x, p(x)) = d(x, \Gamma)$. At point $p(x)$ the decision surface is defined by a list of hyperplanes $L(x) = (H_i)_{i \in I}$ and we have: $p(x) \in \bigcap_{i \in I} H_i$.

Definition 1 *The sensitive tests of point x are the tests $h(x, H)$ verified by x, for the hyperplanes H such that $p(x) \in \bigcap_{L(x)}(H)$, with $d(x, p(x)) = d(x, \Gamma)$.*

Theorem 1 *A generic point has a unique list of sensitive tests for a given decision surface.*

Proof: Points with multiple lists of sensitive tests must have several projections onto Γ. These points belongs to the skeleton[3] of their connected part A (of the same class area). In the case of LDT, the skeleton S is the complement of a dense open set, so unicity of the projection is a generic property. S is a closed set because the number of hyperplanes is finite. So it is impossible for a sequence of points $(x_n) \in S$ with several projections to converge to a point x from the interior of A with a unique projection (otherwise $d(x, p(x)) = 0$, which is impossible). •

Theorem 2 *All the sensitive tests of a point have to be modified to change its class in a small neighborhood (when the decision surface separates only 2 classes at the projection point)*

We note $c(x)$ the label of the class of point x, and d the distance $d(x, p(x))$. Then we have:

$$\exists \epsilon > 0,\ \forall y \in B(x, d + \epsilon),\ y \notin \Gamma,$$
$$c(y) \neq c(x) \Leftrightarrow (\forall H \in L(x),\ h(y, H) \cdot h(x, H) < 0) \quad (1)$$

[3] The skeleton S of A is the set of the centers of all maximal open balls in A (see [12] for a study of its topological properties)

For instance, if point P_3 moves in B_3, it must cross H_3 in order to change its class. If P_2 moves in B_2, it must cross both H_2 and H_3.
Proof: It is possible to show that in a neighborhood of x, points y with a different class from x class are close to $p(x)$.

$$\exists \alpha,\ \beta < \epsilon\ (\beta \sim \sqrt{\alpha}),\ (d(x,y) < d + \alpha) \Rightarrow d(p(x),y) < \beta \quad (2)$$

So $p(x)$ belongs to the boundary of the leaf f that classifies y. Conversely, if relation 1 is not verified, it is possible to find outside Γ a sequence (y_n) converging to $p(x)$, on the opposite side of all $H \in L(x)$ as x, and yet of the same class as x. Since $p(x)$ belongs to the boundary of the different classes, there exists a sequence $(z_n) \notin \Gamma$, converging to $p(x)$, with $c(z_n) \neq c(x)$, and so also on the opposite side of all $H \in L(x)$ as x. We suppose that the sequences are classified by the same leaf respectively f and g. Obviously $f \neq g$, $p(x)$ belongs to the boundary of f and also of g, then f and g have a common boundary H at $p(x)$ that separates the sequences, which is contradictory to their definition. •

Theorems 1 and 2 base the relevance of sensitive tests to describe the situation of a case to the end-user (relatively to a given metric). Different trees inducing the same partition of the input space, for which the trace of a case is different, will lead to the same sensitive tests (This is the case for the points and the DT of Figure 1). These tests have to be modified to change the decision, and it is the smallest modification that changes the decision.

3.2 Robustness

Sensitive tests verify a property of robustness that make them not very sensitive to the uncertainty in the input data.

Theorem 3 *The sensitive tests are stable. For a generic point x, there exists $\epsilon > 0$, such that: $d(x,y) < \epsilon \Rightarrow L(y) = L(x)$*

Proof: The projection is a continuous operator, and Γ a piecewise affine surface, therefore outside the skeleton and outside a finite number of hyperplanes (where the changes occur), the list of sensitive tests remains constant. •

We consider a T-deformation Φ of the decision surface Γ, that is deformation such that $\Phi(\Gamma)$ is still a LDT decision surface. (Typically Φ adds and removes small pieces of hyperplanes). We note $p_\Phi(x)$ the projection point of x onto $\Phi(\Gamma)$, and $L_\Phi(x)$ the list of hyperplanes which define $\Phi(\Gamma)$ at $p_\Phi(x)$.

Theorem 4 *For generic points, a small local T-deformation Φ of the decision surface Γ implies a small change of the projection point and in the NDT case, small changes for the sensitive tests.*

$$(\exists \alpha < \epsilon, \forall z \in \Gamma, d(\Phi(z),z) < \alpha) \Rightarrow d(p_\Phi(x),p(x)) < \epsilon \quad (3)$$

$$d(p_\Phi(x),p(x)) < \epsilon \Rightarrow \forall H_\Phi \in L_\Phi(x),$$
$$\begin{cases} (\forall H \in L(x),\ H_\Phi \perp H) \Rightarrow d(x,H_\Phi) < \epsilon \\ (\exists H \in L(x), H \parallel H_\Phi) \Rightarrow |d(x,H) - d(x,H_\Phi)| < \epsilon \end{cases} \quad (4)$$

In the NDT case, a small global deformation Ψ that changes (of less than ϵ) the threshold value of the hyperplanes defining the tests of the tree implies small changes in the sensitive tests of generic points. In a space of dimension d, $d(p_\Psi(x),p(x)) < \epsilon\sqrt{d}$.

Proof: Relation (3) is verified for LDT, because of relation (2). Relation (3) implies Relation (4) in the NDT case since it is verified for each coordinate. In the NDT case, a change in the threshold value of less than ϵ translates the hyperplanes along their normal vector from less than ϵ. The points at which Γ is not differentiable moves from at worst ϵa_d, where $a_d = \sqrt{d}$ is the diagonal of the unit hypercube. •

These properties of robustness increase the legitimacy to the sensitive tests regarding their use to describe the situation of a case. Close cases (outside the skeleton and the hyperplanes defining the tree) have the same sensitive tests. In the NDT case, small perturbations of the decision surface or small changes of the test threshold values have small effects on the sensitive tests.

3.3 Computing and ordering sensitive tests in the NDT case

The case of DT with separators normal to axes (NDT) is interesting since each test operates on a single attribute, so the tests are supposed to be easier to understand. If the attributes used by the tree can be considered as orthogonal axes, it may be possible to define an inner product. The input space is E is then an Euclidean space.

Theorem 5 *In the NDT case, if the DT uses d attributes, a point has at most d sensitive tests*

Projection onto convex decision surface in the general case is a very classical problem and many algorithms are available (see [1] for a review). But in the NDT case a very simple and efficient algorithm *projectionOnto*Γ inspired from [2] computes the projection point $p(x)$ and the list of hyperplanes $L(x)$ that define the decision surface at the point of projection. It consists in projecting x onto all the leaves f such that class $c(f) \neq c(x)$. The nearest projection point is the projection of x onto the decision surface.

Algorithm 1 *projectionOnto*Γ*(x,DT)*
1. Gather the set F of leaves f which class $c(f) \neq c(x)$;
2. For each $f \in F$ do: {
3. compute $(p_f(x), L_f(x)) =$ projectionOntoLeaf(x,f) ;
4. compute and rank $d(x, p_f(x))$ }
5. Return $(p_n(x), L_n(x))$ with $n = argmin_{f \in F}(d(x, p_f(x)))$

The projection onto a leaf is straightforward since the area classified by the leaf, intersection of halfspaces $E(H)$, is a hyper-rectangle.

Algorithm 2 *projectionOntoLeaf(x,*$(E(H_i))_{i \in I}$
1. y = x; $L = (L_1, ..., L_d)$; $L_1 = ... = L_d = \{\}$
2. For i = 1 to size(I) do: {
3. if $y \notin E(H_i)$ then { *y_u = b ;*
$L_u = H_i$ } }
with u the coordinate corresponding to the attribute defining H_i
b the threshold value for H_i.
4. Return (y,flatten(L))

The complexity of the algorithm *projectionOntoLeaf* is proportional to the size of the leaf, it doesn't depend on the dimension d of the input space. Only step 4 of *projectionOnto*Γ depends on d, for the computation of the distance. The total complexity of the algorithm is in $O(Nd)$ where N is the number of tests of the tree and d the dimension of the input space.

Property 5 is particularly interesting when dealing with big trees. The size of the trace grows with the size of the tree, but it is not the case for the list of sensitive tests. Table 1 shows the comparative size of the trace and the number of sensitive tests, for the test data of the Pima Indian Diabetes (Pima), the Ionosphere and the Wine database of the UCI repository [3] (resp. 256, 151 and 60 cases). Weka j48 [19], a C4.5-based algorithm [16], was used to build the trees on the training data (2/3 of the database). Several trees were build on the Pima database with different pruning parameters.

Table 1. Comparative size of the trace and of the list of sensitive tests.

Feature	Trees (P1 to P4 from Pima)					
	P1	P2	P3	P4	Iono.	Wine
Number of leaves	28	25	19	11	11	5
Max. depth of the tree	10	9	8	7	8	4
Number of attributes	8	8	8	5	9	4
Size of the trace						
Mean	4.9	4.3	4.4	3.1	5.5	2.3
Median	5	5	5	5	6	2
Number of sensitive tests						
Mean	1.5	1.8	1.7	1.7	1.6	1.2
Median	1	2	1	1	1	1

When the dimension of the input space is high, the list of sensitive tests can be to big to be presented to the end-user, and it is necessary to prune the list. This is easy because the sensitive tests can be ordered according to the margin of the test (the distance between the point and the hyperplane that defines the test).

Definition 2 *Let $T(H_1)$ and $T(H_2)$ be two sensitive tests of point x. We say that $T(H_1)$ is more sensitive than $T(H_2)$ iff the margin of $T(H_1)$ is greater than the margin of $T(H_2)$*

$$T(H_1) \succeq T(H_2) \Leftrightarrow d(x, H_1) \geq d(x, H_2) \quad (5)$$

The relation $\succeq$ is a quasi-order on the list of sensitive tests of a case. Actually, greatest margins contribute the most to the distance of the case to the decision surface. The change of the test which has the greatest margin brings the case the closest to the decision surface. It is then interesting to present to the end-user the tests with the highest margin when it is necessary to prune the list of sensitive tests.

4 SENSITIVE TESTS AS SUPPORT OF EXPLANATION

4.1 Displaying the sensitive tests

We now develop an example from the Pima database. The attributes of the database are female patient information and medical test measurements linked to diabetes. The attributes are meaningful and numerical, and (theoretically) there is no missing value. Figure 5 shows a DT built from Pima training data with Weka j48.

An initial metric has to be defined on the input space in order to apply the sensitive tests algorithm. The objective here is to deal with data without unit (normalized) and on a comparable scale (homogenized). So we used simple metrics based on the basic information available on the data set, an estimate of the range of each attribute i or of its mean E_i and of its standard error s_i. The change of coordinate for the Min/Max metric is $y_i^{MM} = \frac{x_i - min_i}{max_i - min_i}$; for the standard metric it is $y_i^s = \frac{x_i - E_i}{s_i}$. The main difference between the two metrics is that the Min/Max metric assumes that the cost of a change is uniform on the range of the values.

We consider two cases of the test database (see Table 2). Both cases are classified by the same leaf (asterisked in Figure 5). The trace of the classification is shown in Figure 4. It is easy to verify in Figure 5 that the change of the tests of the trace doesn't systematically change the predicted class. Except for the first test, which implies a drastic change in the value of the *glucose concentration* for both cases, the modification of one or more tests doesn't change the class. Both cases happen to have the same list of sensitive tests (see Table 3), and the sensitive tests are the same for the two metrics.

Table 2. Cases of the Pima test database.

Attribute	Case 188	Case 63
Glucose concentration in plasma	90	93
Diastolic blood pressure (mmHg)	68	50
Body mass index (kg/m^2)	38.2	28.7
Diabetes pedigree function	0.503	0.356
Age (years)	27	23
Predicted class	0 (not diabetic)	0
Real class	1 (diabetic)	0

We can see in Figure 5 that the modification of all the sensitive tests changes the class of the points: they will be classified by leaf C_3. Besides it is possible to give information about the distance of the cases to the decision surface. Table 4 shows for both cases the distance to the decision surface compared to the theoretical maximum distance (distance to leaf C_1) and to the norm of the standard error vector. We can see that Case 188 is very close to the decision surface. On the contrary, the threshhold value of the attribute *diabetes predigree* sensitive test is far from its value for Case 63.

Table 3. Sensitive tests for the Pima cases.

Sensitive Tests
Age ≤ 28.5
Diabetes pedigree function (d) ≤ 0.546
Glucose concentration (g) ≤ 94.5

Table 4. Distance information.

Metric	Case 188	Case 63	Norm of SE	Max. distance
Min/Max	0.038	0.123	0.35	0.7
Standard	0.223	1.474	1	4.3

4.2 The metric as support of interaction

The change of the metric is performed relatively easily since the algorithm *projectionOntoLeaf* doesn't depend on the metric, as long as the change of the metric keeps invariant the hyperplanes with normal vectors parallel to axes. This is the case for a broad set of transformation, in particular the dilatation. Any change of coordinate of the type $y = a^T \cdot (x - b)$ where a^T is the transpose vector of strictly positive coefficient and b a point, keeps the projection of a point x onto the leaves invariant. So the change of the metric implies only the computation and the ranking of the new distances between x and its projections (step 4 of the *projectionOnto*Γ algorithm).

We assumed that some basic information was available in order to define the initial metrics. In fact it can be estimated from the training data used to build the tree. (With the metric Min/Max, the bounds for each attributes have to be verified for each new case). The initial metrics allow us to give a first description of the case to the end-user, with the sensitive tests and the position of the case in relation to the decision surface. In the case of the Pima database, both metrics give similar results. The percentage of cases for which the projection point is different varies from 1.95% to 3.13%, depending on the size of the database on which the mean and standard error of each attribute are

estimated. But it is not always the case (for the Wine database [3], it can be 20%, since the database is smaller).

From this initial description, the end-user can suggest some modification of the metric or of the input space. Tree types of modification seem interesting:

1. The freeze of an attribute: *diabetes pedigree function* is constant.
2. The freeze of an attribute direction: *age* cannot decrease.
3. To consider a weighted cost for the different direction of move: the initial metrics are modified to take into account the new dilatation coefficient.

For both Pima cases, a constraint on the direction of *age* doesn't change the sensitive tests. If the *diabetes pedigree function* remains constant, then Table 5 shows that the new list of sensitive tests is reduced to a test on the glucose concentration C_4. Table 5 also shows that a change of the weight in the definition of the metric cannot change the result, since the *glucose concentration* is part of the definition of each leaf. In a multi class problem, the same Table will

Table 5. Distance to the different leaves

Leaf	Hyperplanes defining Γ	Case 188	Case 63
C_1	g = 137.5	0.239	0.224
C_2	g = 43.2.5 age = 28.5	0.236	0.266
C_3	(see Table 3)	0.038	0.123
C_4	g = 127.5	0.188	0.143

show the most sensitive tests relative to each different class.

5 CONCLUSION

We have presented a method to search a linear decision tree for the most relevant tests, in order to help the end-user to understand the

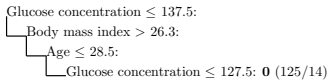

Figure 4. Trace of classification of cases from Table 2.

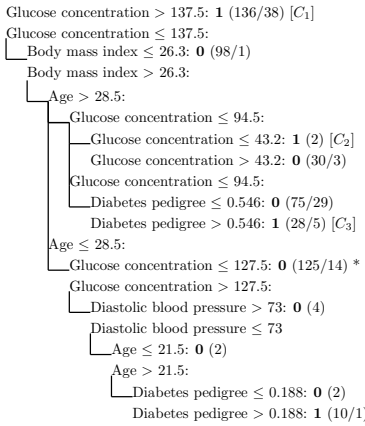

Figure 5. Pima DT. The class returned by a leaf is bold. (The number of cases/errors appears between parentheses). Leaves of class 1 are labelled.

prediction of the tree for a new case. In the case of axis-parallel decision trees, a very simple algorithm provides the list of sensitive tests for a case, and examples show that it can provide interesting information. More tests are necessary to verify that the efficiency of the algorithm is sufficient for big trees and for higher dimension, since in high dimension space the computation of the Euclidean distance can become costly. In the case of oblique tree, the algorithm of projection onto the decision surface is not as simple, and tests have to be done with recursive algorithm [1]. The main limit of the method is the difficulty of its extension to non numerical data, since the definition of a metric is necessary. But since optimality (regarding the classification task) is not requested, utility or cost functions could eventually be used. In the NTD case, the sensitive tests could be presented to the end-user with some information on the connected component of the case. The description of the situation would be better, and this would help the end-user to suggest appropriate metric change.

REFERENCES

[1] H. Bauschke and J.M. Borwein, 'On projection algorithms for solving convex feasibility problems', *Society for Industrial and Applied Mathematics Review*, **38**(3), 367–426, (1996).

[2] K. Bennett and J. Blue. Optimal decision trees, technical report no 214, rensselaer polytechnic institute, 1996.

[3] C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.

[4] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth, Belmont, 1984.

[5] B. Chandrasekaran, M.C. Tanner, and J.R. Josephson, 'Explaining control strategies in problem solving', *IEEE Expert*, **4**(1), 9–24, (1989).

[6] N.V. Chawla, K.W. Bowyer, L.O. Hall, and W.P. Kegelmeyer, 'Smote: Synthetic minority over-sampling technique', *Journal of Artificial Intelligence and Research*, **16**, 321–357, (2002).

[7] P. Domingos, 'Metacost: A general method for making classifiers cost-sensitive', in *Knowledge Discovery and Data Mining*, pp. 155–164, (1999).

[8] F. Esposito, D. Malerba, and G. Semeraro, 'A comparative analysis of methods for pruning decision trees', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**(5), 476–491, (1997).

[9] L. Karsenty and P. Brezillon, 'Cooperative problem solving and explanation', *International Journal of Expert Systems With Applications*, **8**(4), 445–462, (1995).

[10] M.J. Kearns and D. Ron, 'Algorithmic stability and sanity-check bounds for leave-one-out cross-validation', in *Computational Learing Theory*, pp. 152–162, (1997).

[11] R. Kohavi, 'A study of cross-validation and bootstrap for accuracy estimation and model selection', in *IJCAI*, pp. 1137–1145, (1995).

[12] G. Matheron, *Examples of topological properties of skeletons*, 217–257, Image Analysis and Mathematical Morphology, Academic Press, 1988.

[13] S.K. Murthy, 'Automatic construction of decision trees from data: A multi-disciplinary survey', *Data Mining and Knowledge Discovery*, **2**(4), 345–389, (1998).

[14] C. Paris, M.R. Wick, and W.B. Thompson, 'The line of reasoning versus the line of explanation', in *AAAI'88 Workshop on Explanation*, pp. 4–8, (1988).

[15] F.J. Provost and T. Fawcett, 'Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions', in *Knowledge Discovery and Data Mining*, pp. 43–48, (1997).

[16] J.R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, 1993.

[17] M. Umano, K. Okomato, I. Hatono, H. Tamura, F. Kawachi, S. Umezu, and J. Kinoshita, 'Fuzzy decision trees by fuzzy id3 algorithm and its application to diagnosis systems', in *3rd IEEE International Conference on Fuzzy Systems*, pp. 2113–2118, (1994).

[18] M.R. Wick and J.R. Thomson, 'Reconstructive expert system explanation', *Artificial Intelligence*, **54**(4), 33–70, (1992).

[19] I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementation*, Morgan Kaufmann, 2000.

[20] L. Yu and H. Liu, 'Feature selection for high-dimensional data: A fast correlation-based filter solution', in *ICML*, pp. 856–863, (2003).