Learning Decision Trees Using the Area Under the ROC Curve

Cèsar Ferri CFERRI@DSIC.UPV.ES
Departament de Sistemes Informàtics i Computació, Universitat Politècnica de València, SPAIN

Peter Flach PETER.FLACH@BRISTOL.AC.UK
Department of Computer Science, University of Bristol, UK

José Hernández-Orallo JORALLO@DSIC.UPV.ES
Departament de Sistemes Informàtics i Computació, Universitat Politècnica de València, SPAIN

Abstract

ROC analysis is increasingly being recognised as an important tool for evaluation and comparison of classifiers when the operating characteristics (i.e. class distribution and cost parameters) are not known at training time. Usually, each classifier is characterised by its estimated true and false positive rates and is represented by a single point in the ROC diagram. In this paper, we show how a single decision tree can represent a set of classifiers by choosing different labellings of its leaves, or equivalently, an ordering on the leaves. In this setting, rather than estimating the accuracy of a single tree, it makes more sense to use the area under the ROC curve (AUC) as a quality metric. We also propose a novel splitting criterion which chooses the split with the highest local AUC. To the best of our knowledge, this is the first probabilistic splitting criterion that is not based on weighted average impurity. We present experiments suggesting that the AUC splitting criterion leads to trees with equal or better AUC value, without sacrificing accuracy if a single labelling is chosen.

1. Introduction

Traditionally, classification accuracy (or error), i.e., the percentage of instances that are correctly classified (respectively incorrectly classified) has been used as a measure of the quality of classifiers. However, in many situations, not every misclassification has the same consequences, and problem-dependent misclassification costs have to be taken into account. If the cost parameters are not known at training time, Receiver Operating Characteristic (ROC) analysis can be applied (Provost & Fawcett 1997; Swets, Dawes & Monahan 2000). ROC analysis provides tools to distinguish classifiers that are optimal under some class and cost distributions from classifiers that are always sub-optimal, and to select the optimal classifier once the cost parameters are known.

ROC analysis for two classes is based on plotting the true-positive rate (TPR) on the *y*-axis and the false-positive rate (FPR) on the *x*-axis. This gives a point for each classifier. A curve is obtained because, given two classifiers, we can obtain as many derived classifiers as we want along the segment that connects them, just by voting them with different weights. Consequently, any point "below" that segment will have greater cost for any class distribution and cost matrix, because it has lower TPR and/or higher FPR. According to that property, given several classifiers, one can discard the classifiers that fall under the convex hull formed by the points representing the classifiers and the points (0,0) and (1,1), which represent the default classifiers always predicting negative and positive, respectively.

This paper is concerned with taking costs into account when learning decision trees. If costs are known at training time, the training algorithm could be made cost-sensitive, e.g. by incorporating costs in the splitting criterion. However, it has been shown that such cost-sensitive techniques do not lead to trees with lower costs (Drummond and Holte 2000; Elkan 2001) and that cost-sensitive class labelling is more effective (Bradford et al. 1998; Ferri, Flach & Hernandez 2002). In this paper we assume that costs are unknown at training time. Clearly, each of the 2^n possible labellings of the n leaves of a given decision tree establishes a classifier, and we can use ROC analysis to determine the optimal labellings among them. However, this set of classifiers has special properties (e.g., for any classifier there is another one making opposite predictions) which allows a more direct computation of the optimal labellings. We prove that there are n+1 of these, which are determined by a simple ordering on the leaves of the tree.

Thus, from a cost-sensitive perspective it makes sense to view a decision tree as an unlabelled tree with an ordering

on the leaves. Furthermore, this suggests to use the area under the ROC curve (AUC), obtained by plotting the n+1 optimal labellings in ROC space, to evaluate the quality of a decision tree (or any other partitioning of instance space). A natural question is then whether existing decision tree algorithms – which aim at optimising the accuracy of a single labelling – also lead to good AUC values, or whether we can do better by adapting the algorithm. We show that a simple AUC-based splitting criterion leads to trees with better AUC, without sacrificing accuracy if a single labelling is chosen. To the best of our knowledge, this is the first probabilistic splitting criterion that is not based on weighted average impurity.

The paper is organised as follows. Section 2 poses the problem of finding all labellings of the tree on the ROC convex hull, and shows how to effectively obtain this subset of labellings. In section 3, we discuss the AUC metric and propose the AUC-based splitting criterion. In section 4 we experimentally compare AUCsplit with several well-known impurity-based splitting criteria with respect to accuracy and AUC. Finally, section 5 closes the paper with a discussion of the main conclusions and some plans for future work.

2. Finding Optimal Labellings of a Decision Tree

A decision tree classifier can be represented by a point in the ROC space. However, if we change the class assignment of one leaf, we obtain a different classifier and hence a different point in the ROC space. Note that this change can be made a posteriori, after the tree was learnt or pruned. By changing in many different ways the assignments of each leaf of the tree we can obtain different trees. In what follows we will call *labelling* a set of assignments to each tree leaf.

The idea is to view the ROC curve of a decision tree not as the three-point curve given by a single labelling together with the two default classifiers, but as the convex hull defined by all the possible labellings. The problem is that given n leaves and c classes, there are c^n possible labellings. Although this value alone can make this intractable for many trees even for two classes, the problem gets worse if we consider that we would need to compute the convex hull of these c^n points. Note that the cost of computing the convex hull of N points in a d-dimensional space is in $O(N \log N + N^{d/2})$ (Boissonat & Yvinex 1998). Consequently, one relevant question is whether there is a way to restrict these c^n combinations and obtain the same ROC curve.

2.1 Preliminaries

Given a set of tree leaves l_k ($1 \le k \le n$) and a training set S with possible classes c_i ($1 \le i \le c$), we denote by E_k the number of examples of S that fall under leaf l_k, and we denote by E_k^i the number of examples of S that fall under leaf l_k of class i. The k subscript in E_k^i can be dropped when the leaf is clear from context. A *labelling* is defined as a set of pairs of the form (k, i), where k represents the leaf l_k and i represents the class assigned to that leaf. The set of all possible labellings is denoted by Λ. Clearly, the cardinality of Λ is c^n.

In what follows, we study for 2-class problems how we can restrict the 2^n labellings but still obtain the points on the convex hull. We will denote the two classes: + and –. We also assume the following properties:

$$E_k^+ \ge 0, E_k^- \ge 0, E_k^+ + E_k^- > 0, \forall\, 1 \le k \le n \ ,$$

$$\sum_{1 \le k \le n} E_k^- > 0 \text{, and } \sum_{1 \le k \le n} E_k^+ > 0$$

That means that there are no empty leaves and that there exists at least one example of each class.

We use the following notation for cost matrices $C_{i,j}$:

		ACTUAL +	ACTUAL –
PREDICTED	+	C_{++}	C_{+-}
	–	C_{-+}	C_{--}

where all the costs are greater or equal than 0. Additionally, $C_{-+} > C_{++}$, $C_{+-} > C_{--}$. Given a leaf l_k we define $Cost_k^i$ as the cost of the examples under that leaf if class i would be assigned:

$$Cost_k^i = \sum_j E_k^j \cdot C_{i,j}$$

The best assignment for a leaf l_k is then defined as:

$$Best_k = \arg\min_i Cost_k^i$$

The optimal labelling S_{opt} for a given cost matrix C is then given by:

$$S_{opt} = \{(k, Best_k)\}_{1 \le k \le n}$$

which means that each leaf is assigned the class that minimises the cost for the cost matrix C.

2.2 Subset of Labellings Forming the Convex Hull

In this section we determine the subset of decision tree labellings on the convex hull.

Lemma 1. Given a leaf of a decision tree for a 2-class problem with the distribution E_+ and E_-, and given a cost matrix C, the cost is minimised if the leaf is assigned class + when

$$\frac{E_+}{E_- + E_+} \ge \frac{(C_{+-} - C_{--})}{(C_{-+} - C_{++}) + (C_{+-} - C_{--})}$$

and assigned class – otherwise.

Proof: The cost of this leaf will be assigned to + iff $Cost^+ \le Cost^-$, i.e.

$$E^+ \cdot C_{++} + E^- \cdot C_{+-} \le E^+ \cdot C_{-+} + E^- \cdot C_{--}$$

$$\frac{E_+}{E_- + E_+} \ge \frac{(C_{+-} - C_{--})}{(C_{-+} - C_{++}) + (C_{+-} - C_{--})}$$

The value on the left hand side is defined as the *local positive accuracy* of a leaf l_k, and is denoted by r_k. This result has also been used elsewhere to assign classes (see e.g. Elkan 2001), but we will use it to *order* the leaves. The value on the right hand side of the equation is called the *cost ratio* (CR). In particular, when r_k = CR either class can be assigned arbitrarily.

The main definition of this section is the following.

Definition 2 (Optimal labellings). Given a decision tree for a problem with 2 classes formed by n leaves $\{l_1, l_2, \ldots, l_n\}$ *ordered* by local positive accuracy, i.e, $r_1 \geq r_2, \ldots, r_{n-1} \geq r_n$, we define the set of optimal labellings $\Gamma = \{S_0, S_1, \ldots, S_n\}$ where each labelling S_i $(0 \leq i \leq n)$ is defined as: $S_i = \{A^1_i, A^2_i, \ldots, A^n_i\}$ where $A^j_i = (j,+)$ if $j \leq i$ and $A^j_i = (j,-)$ if $j > i$.

The following three lemmas are needed to establish the main result of this section. The reader in a hurry may wish to skip the technical details and proceed directly to Theorem 6 and the subsequent example.

Lemma 3. Given a decision tree for a problem with 2 classes with n leaves, the labelling that minimises the cost according to the training set and an arbitrary cost matrix belongs to the set of optimal labellings Γ.

Proof: The cost matrix has one degree of freedom expressed with the CR. Imagine that the CR is 1, then all the leaves will be set to –, according to Lemma 1 (in the case r_k = CR we also select –). This labelling is in Γ. This solution minimises the cost of any matrix until $r_1 \leq \text{CR} \leq r_2$. Then, leaf l_1 will change its assignment to + according to Lemma 1; this labelling also belongs to Γ. We can repeat this argument until CR=0, where all the leaves will be set to +. Thus, there are n+1 states that correspond to the labellings in Γ.

Blockeel and Struyf (2001) used the same set of assignments. However, no theoretical properties were discussed. Lemma 3 shows that the set of optimal labellings is sufficient for calculation of the convex hull. We now proceed to show that these points are also necessary.

Lemma 4. Given three labellings from the set of optimal labellings Γ: S_{i-1}, S_i, S_{i+1} $(1 \leq i \leq n-1)$, the point in the ROC space corresponding to classifier S_i is above the convex hull formed by (0,0), (1,1), and the points in the ROC space corresponding to classifiers S_{i-1}, S_{i+1}, if and only if

$$\frac{E_i^+}{E_i^- + E_i^+} > \frac{E_{i+1}^+}{E_{i+1}^- + E_{i+1}^+}.$$

Proof: The three points in the ROC space corresponding to S_{i-1}, S_i, S_{i+1} are:

$$P_{i-1} = \left(\frac{x_{i-1}}{x_t}, \frac{y_{i-1}}{y_t} \right),$$

$$P_i = \left(\frac{x_{i-1} + E_i^-}{x_t}, \frac{y_{i-1} + E_i^+}{y_t} \right),$$

$$P_{i+1} = \left(\frac{x_{i-1} + E_i^- + E_{i+1}^-}{x_t}, \frac{y_{i-1} + E_i^+ + E_{i+1}^+}{y_t} \right)$$

$$\text{where } x_t = \sum_{1 \leq j \leq n} E_j^-, \; y_t = \sum_{1 \leq j \leq n} E_j^+,$$

$$x_{i-1} = \sum_{1 \leq j \leq i-1} E_j^- \text{ and } y_{i-1} = \sum_{1 \leq j \leq i-1} E_j^+.$$

Obviously, $\frac{x_{i-1}}{x_t} \leq \frac{x_{i-1} + E_i^-}{x_t} \leq \frac{x_{i-1} + E_i^- + E_{i+1}^-}{x_t}$.

Thus, according to the definition of ROC curve, we only want to know when P_i is *above* the straight line that joins P_{i-1} and P_{i+1}, focusing on the y coordinate.

The formula of a straight line that joins two points $P_1=(X_1,Y_1)$ and $P_2=(X_2,Y_2)$ is:

$$y = \frac{Y_2 - Y_1}{X_2 - X_1}(x - X_1) + Y_1$$

Substituting $P_1 = P_{i-1}$ and $P_2 = P_{i+1}$, the y coordinate of P_i will be above iff:

$$\frac{y_{i-1} + E_i^+}{y_t} > \frac{\frac{y_{i-1} + E_i^+ + E_{i+1}^+}{y_t} - \frac{y_{i-1}}{y_t}}{\frac{x_{i-1} + E_i^- + E_{i+1}^-}{x_t} - \frac{x_{i-1}}{x_t}} \cdot \left(\frac{x_{i-1} + E_i^-}{x_t} - \frac{x_{i-1}}{x_t} \right) + \frac{y_{i-1}}{y_t}$$

$$\text{iff} \quad \frac{y_{i-1} + E_i^+}{y_t} > \frac{\frac{E_i^+ + E_{i+1}^+}{y_t}}{\frac{E_i^- + E_{i+1}^-}{x_t}} \cdot \left(\frac{E_i^-}{x_t} \right) + \frac{y_{i-1}}{y_t}$$

$$\text{iff} \quad E_i^+ \cdot E_{i+1}^- > E_i^- \cdot E_{i+1}^+,$$

$$\text{iff} \quad \frac{E_i^+}{E_i^- + E_i^+} > \frac{E_{i+1}^+}{E_{i+1}^- + E_{i+1}^+}$$

We have shown this result for three consecutive classifiers of the set of optimal labellings; however, it also holds for three non-consecutive classifiers.

Lemma 5. Given three labellings from the set of optimal labellings Γ: S_{i-1}, S_i, S_{i+1} $(1 \leq i \leq n-1)$ such that $r_i = r_{i+1}$, it is not necessary to consider the point in the ROC space corresponding to S_i, because it will not affect the convex hull.

Proof: If $r_i = r_{i+1}$ then

$$\frac{E_i^+}{E_i^- + E_i^+} = \frac{E_{i+1}^+}{E_{i+1}^- + E_{i+1}^+},$$

which, according to Lemma 4, means that the point in the ROC space corresponding to S_i is placed just on the straight line between the points in the ROC space corresponding to S_{i-1} and S_{i+1}.

We can now formulate the main result of this section.

Theorem 6. Given a decision tree for a problem of 2 classes with n leaves, the convex hull of the 2^n possible labellings is formed by exactly those ROC points

corresponding to the set of optimal labellings Γ, removing repeated leaves with the same local positive accuracy.

Proof: From Lemma 3 we can easily derive that all the ROC points that are on the convex hull from the 2^n possible labellings belong to the ROC points generated from the set of optimal classifiers. We only have to show that all the ROC points from the set of optimal labellings are on the convex hull. Suppose we have three consecutive labellings S_{i-1}, S_i, S_{i+1}, where S_{i-1} and S_{i+1} are on the convex hull. Lemma 4 has shown that S_i will be above the convex hull iff $r_i \geq r_{i+1}$, which is the case since the set of optimal labellings is ordered by local positive accuracy. In the case that $r_i = r_{i+1}$ we have, from Lemma 5, that we can remove one of them.

The relevance of Theorem 6 is that computation of the convex hull of the 2^n possible labellings of the n leaves of a decision tree is equivalent to ordering the leaves by local positive accuracy.

2.3 Example

Suppose we have a decision tree with three leaves and the following training set distribution:

	+	–
LEAF 1	3	5
LEAF 2	5	1
LEAF 3	4	2

There are 2^3=8 possible classifiers corresponding to each labelling in Λ. Figure 1 represents the ROC points of these classifiers. As can be seen in the figure, the points are mirrored through the point (0.5, 0.5), because for each labelling there is another labelling assigning the opposite class to each leaf.

We first order the leaves by the local positive accuracy and then we generate the set of n+1= 4 optimal labellings:

	+	–	S_0	S_1	S_2	S_3
LEAF 1	5	1	-	+	+	+
LEAF 2	4	2	-	-	+	+
LEAF 3	3	5	-	-	-	+

If we plot the ROC points of these 4 combinations, these are the points corresponding to the convex hull of Λ, which are shown in Figure 1 as squares.

A final question is what to do with empty leaves, a case that we have not considered in the previous results because we excluded this case in the assumptions. Empty leaves can be generated when there are splits with more than two leaves, some of which may not cover any example, i.e., E^+=0 and E^-=0. One easy solution to this problem is to use some kind of smoothing (such as Laplace or m-estimate) for E^+ and E^-. Another option is to assign a local positive accuracy 0.5 and work with the leaf without cardinality, not affecting the ROC curve.

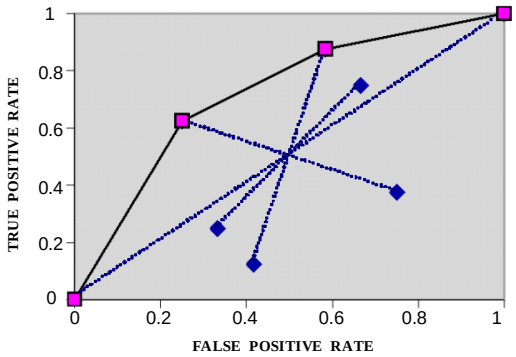


Figure 1. ROC points and convex hull of set Λ.

3. AUC-based Decision Tree Evaluation and Construction

The previous analysis supports the interpretation of a decision tree as having several possible labellings of the leaves, or alternatively, an ordering on the leaves. We propose to use the *area under the ROC curve* (AUC) obtained from these labellings to evaluate the quality of the tree. Notice that if the accuracy of the tree is 100%, all leaves are pure and the ROC curve covers the whole space. If the tree consists of a single unsplit leaf, the two labellings of this leaf correspond to the two default classifiers and the area under the curve is 0.5. Also notice that we can even calculate the AUC of a single labelling, i.e. the area under the curve (0,0)–(FPR,TPR)–(1,1), which is (TPR-FPR+1)/2, i.e. the average of positive and negative accuracies.

3.1 The AUC Metric for Decision Tree Evaluation

In order to compute the area under the ROC curve we employ the leaf ordering from the previous section to compute the areas of each trapezoid. Specifically, it is easy to compute the area between two consecutive points P_{i-1} and P_i in the ROC curve given by the set Γ:

$$A(P_{i-1}, P_i) = \frac{E_i^-}{x_t} \cdot \frac{2y_{i-1} + E_i^+}{2y_t}$$

where y_{i-1}, x_t and y_t are as defined in Lemma 4. Since the first point is P_0=(0,0), we can define AUC as follows.

Definition 7 (AUC). Let Γ be the set of optimal labellings of a decision tree with n leaves, then the AUC metric is defined as

$$\mathrm{AUC}(\Gamma) = \sum_{i=1..n} A(P_{i-1}, P_i) = \sum_{i=1..n} \frac{E_i^-}{x_t} \cdot \frac{2y_{i-1} + E_i^+}{2y_t} = \frac{1}{2x_t y_t} \sum_{i=1..n} E_i^- \left[\left(\sum_{j=1..i-1} 2E_j^+ \right) + E_i^+ \right]$$

(See Lemma 4 for the meaning of the symbols.)

AUC is like any other machine learning metric in that it is a population statistic which needs to be estimated from a

sample. We can use the standard techniques of using a test set or cross-validation to obtain such an estimate. In the case of a test set, note that the leaf ordering is obtained during training, while the leaves' positive and negative coverage is determined on the test set. Consequently, the ROC curve on the test set may not be convex (it is, however, monotonically non-decreasing by construction). Definition 7 is a general geometric construction which does not assume convexity of the curve.

3.2 The AUCsplit Splitting Criterion for Decision Tree Construction

In the previous section we have argued that AUC is a better metric than accuracy for evaluating decision trees when class and cost distributions are unknown at training time. However, the existing methods for growing decision trees typically use splitting criteria based on error/accuracy or discrimination. In this subsection we propose an AUC-based splitting criterion.

Without the results introduced in section 2, computing the AUC corresponding to a set of n leaves could be computationally expensive, especially if splits have more than two children. Using the optimal labelling set Γ, AUC of the leaves under a split can be computed efficiently. In particular, given several possible splits for growing the tree, where each split consists of a set of new leaves, we can compute the ordering of these leaves and calculate the corresponding ROC curve. The area under this curve could be compared to the areas of other splits in order to select the best split. More precisely, we can use the previous formula for AUC(Γ). This yields a new splitting criterion.

Definition 8 (AUCsplit). Given several splits s_j, each one formed by n_j leaves $\{l^j_1, l^j_2, ..., l^j_{nj}\}$, then the best split is the one that maximises:

$$AUCsplit(s_j) = \sum_{i=1..n_j} A(P^j_{i-1}, P^j_i)$$

where the points P_i^j are obtained in the usual way (sorting the leaves of each split by local positive accuracy).

The first question that arises with a new splitting criterion is how it differs from other criteria previously proposed. To answer this question, let us review the general formula of other well-known splitting criteria, such as Gini (Breiman et al. 1984), Gain, Gain Ratio and C4.5 criterion (Quinlan 1993) and DKM (Kearns & Mansour 1996). These splitting criteria find the split with the lowest $I(s^j)$, where $I(s_j)$ is defined as:

$$I(s) = \sum_{j=1..n_j} p_j \cdot f(p^+_j, p^-_j)$$

where p_j is the probability of being sorted into that node in the split (cardinality of child node divided by the cardinality of parent node). Using this general formula, each splitting criterion implements a different function f, as shown in the following table:

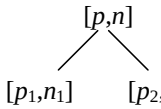
CRITERION	$f(a,b)$
ACCURACY (EERROR)	$\min(a,b)$
GINI (CART)	$2ab$
ENTROPY (GAIN)	$a\cdot\log(a)+b\cdot\log(b)$
DKM	$2(a\cdot b)^{1/2}$

These functions $f(a,b)$ are impurity functions, and the function $I(s)$ calculates a weighted average of the impurity of the children in a split. In general, we need to compare this weighted average impurity of the children with the impurity of the parent, if we are comparing different splits of different nodes.

Consider for instance the following two splits:

The children have the same weighted average impurity in both cases. In order to see that the left is a better split than the right (assuming $a \neq b$), we need to take the impurity of the parent into account. In contrast, AUCsplit evaluates the quality of the whole split (parent + children) and cannot be reduced to a difference in impurity between parent and children. The left split has AUCsplit=$a/(a+b)$ (assuming $a>b$), while the right split has AUCsplit=0.5, indicating that nothing has been gained in ROC space with respect to the default diagonal from (0,0) to (1,1).

An interesting relationship can be established with the Gini index. Consider the following binary split:



If the left child has higher local positive accuracy, then we have:

$$AUCsplit = \frac{1}{2}\left(\frac{p_1}{p} - \frac{n_1}{n} + 1\right) = \frac{p_1 n - p n_1 + p n}{2pn} = \frac{p_1 n + p n_2}{2pn}$$

It is interesting to note that the denominator of this expression is the Gini index of the parent, and the enumerator could be called a mutual Gini index of the children given the parent.

Finally, we have to consider the computational complexity of calculating the AUCsplit with respect to other well-known splitting criteria. Let n denote the maximum number of children in all the splits. Then, if we have k partitions, the selection of the best split by using any of the information measures $I(s_j)$ requires, for each partition, n computations of the entropy formula, that can be considered in O(1). Consequently, the cost would be in O($k\cdot n$). On the other hand, the selection of the best split by using the AUCsplit(s_i) requires $n\cdot\log n$ for sorting the n nodes, and n computations of the A($\cdot,\cdot$) formula that can be considered in O(1). Consequently, the cost would be $k\cdot(n\cdot\log n + n)$ which is in O($k\cdot n\cdot\log n$). This difference in

log *n* is negligible especially if we realise that the number of children of a partition is 2 for numerical attributes and very small for nominal attributes.

4. Experimental Evaluation

We evaluate the previous methods by using 25 datasets extracted from the UCI repository (Blake and Merz 1998). All of them have two classes, either originally or by selecting one of the classes and joining all the other classes. Table 1 shows the dataset (and the class selected in case of more than two classes), the size in number of examples, the nominal and numerical attributes and the percentage of examples of the minority class.

Table 1. Datasets used for the experiments.

#	Dataset	Size	Attributes		%min
			nom	num	class
1	Monks1	566	6	0	50
2	Monks2	601	6	0	34.28
3	Monks3	554	6	0	48.01
4	Tic-tac	958	8	0	34.66
5	House-votes	435	16	0	38.62
6	Agaricus	8124	22	0	48.2
7	Breast-wdbc	569	0	30	37.26
8	Breast-wpbc	194	0	33	23.71
9	Ionosphere	351	0	34	35.9
10	Liver	345	0	6	42.03
11	Pima	768	0	8	34.9
12	Chess-kr-vs-kp	3196	36	0	47.78
13	Sonar	208	0	60	46.63
14	Breast-Cancer	683	0	9	34.99
15	Hepatitis	83	14	5	18.07
16	Thyroid-hypo	2012	19	6	6.06
17	Thyroid-sick-eu	2012	19	6	11.83
18	tae [{0}]	151	2	3	32.45
19	cars [{unacc}]	1728	6	0	29.98
20	nursery [{nr}]	12960	8	0	33.33
21	pendigits [{0}]	10992	0	16	10.4
22	page-blocks [{0}]	5473	0	10	10.23
23	yeast [{ERL}]	1484	0	8	31.2
24	letter [{a}]	20000	0	16	3.95
25	optdigits [{0}]	5620	0	64	9.86

The first thing to be considered is the behaviour of classical splitting criteria with the AUC evaluation measure. We compare the most commonly used splitting criteria: Gain Ratio (only considering splits with at least average gain as is done in C4.5), Gini (as used in CART), DKM and Expected Error. All the experiments have been done within the SMILES system (Ferri et al. 2002) that includes all of these criteria, the labelling method and AUC computation. The use of the same system for all the methods makes the criteria comparison more impartial because all the other things remain equal.

The experiments were performed with and without pruning, although we only show the methods with pruning because the results are better in general (both in accuracy and AUC) for all the splitting criteria. The post-pruning method used is the "Pessimistic Error Pruning" introduced by (Quinlan 1987). According to the study in (Esposito, Malerba & Semeraro 1997), this is the best method that does not modify the tree structure (unlike C4.5 pruning). Although it has a tendency to underprune, we think that it is a quite simple and effective method that allows a fairer comparison. We have also used frequency smoothing (Laplace correction) for the nodes in each split, because it is favourable for the AUC measure for all methods, especially Gini and DKM. Accuracy of Gain Ratio results are slightly worse when smoothing is used, although AUC values are still better. Table 2 shows AUC results obtained by 10-fold cross-validation.

Table 2. AUC values for different splitting criteria.

Set	Gain Ratio	Gini	DKM	EErr
1	81.5 ± 14.0	79.8 ± 11.9	79.8 ± 11.9	82.2 ± 5.3
2	60.6 ± 10.4	57.7 ± 8.4	55.5 ± 7.9	69.8 ± 4.1
3	98.8 ± 1.6	98.7 ± 1.7	98.7 ± 1.7	95.4 ± 2.6
4	81.3 ± 8.0	80.6 ± 7.5	79.8 ± 8.1	76.4 ± 5.6
5	96.9 ± 2.5	96.9 ± 2.5	96.9 ± 2.5	96.9 ± 2.5
6	1 ± 0	99.9 ± 0.2	1 ± 0	1 ± 0.1
7	91.1 ± 6.6	90.9 ± 5.8	95.7 ± 5.3	93.6 ± 3.7
8	58.1 ± 24.4	66.4 ± 18.3	54.9 ± 18.6	51.2 ± 3.5
9	88.8 ± 10.2	56.1 ± 13.6	90.8 ± 5.0	59.0 ± 15.1
10	65.1 ± 6.7	63.4 ± 8.2	65.6 ± 8.4	59.9 ± 9.4
11	78.0 ± 5.2	27.8 ± 3.5	69.3 ± 25.7	30.5 ± 39.8
12	99.7 ± 0.4	99.3 ± 0.4	99.7 ± 0.3	98.3 ± 0.8
13	60.6 ± 10.2	69.7 ± 10.4	72.7 ± 6.8	68.1 ± 12.8
14	95.5 ± 2.5	95.2 ± 2.7	96.8 ± 2.1	94.8 ± 2.9
15	92.9 ± 12.4	65.4 ± 24.4	72.9 ± 26.3	65 ± 24.2
16	83.2 ± 16.5	48.6 ± 51.2	96.9 ± 5.7	34.8 ± 41.1
17	93.6 ± 3.2	49.7 ± 46.1	65.8 ± 45.5	3.7 ± 11.3
18	50.5 ± 25.9	48.9 ± 27.1	52.5 ± 24.5	21.5 ± 21.4
19	98.1 ± 0.7	98.2 ± 0.8	98.1 ± 0.8	97.8 ± 1.1
20	1 ± 0	1 ± 0	1 ± 0	1 ± 0
21	99.7 ± 0.6	98.2 ± 0.7	99.7 ± 0.3	96.3 ± 2.1
22	93.7 ± 3.7	81.7 ± 4.9	66.6 ± 21.6	50 ± 0
23	73.7 ± 3.1	66.6 ± 9.9	73.5 ± 4.3	51.0 ± 4.0
24	98.7 ± 1.0	95.9 ± 2.4	99.4 ± 0.5	85.7 ± 0.5
25	98.1 ± 2.3	95.9 ± 3.3	98.0 ± 2.6	96.0 ± 3.3
M	85.53	77.26	83.19	71.12

Although all methods behave very similarly in terms of accuracy (as has been shown in the machine learning literature and by our own experiments not listed here), the differences in AUC are very noticeable, especially in datasets 9, 11, 15, 16, 17, 22, 23. There is no apparent relationship with any dataset characteristic except the minority class proportion, which will be analysed at the end of this section.

The worst methods according to the AUC measure are clearly Gini and Expected Error. Better and more similar results are given by GainRatio and DKM. If we select

Gain Ratio as the best classical method, we can compare its results with AUCsplit results. In order to make comparisons significant, we have repeated 10-fold cross validation 10 times, making a total of 100 learning runs for each pair of dataset and method. These new results are shown in Table 3.

Table 3. Accuracy and AUC for Gain Ratio and AUCsplit.

	GAIN RATIO		AUCSPLIT		BETTER?	
SET	ACC.	AUC	ACC.	AUC	ACC.	AUC
1	90.7±6.6	83.6±11.8	96.5±3.9	94.3±6.7	✓	✓
2	57.7±6.5	61.1±7.9	56.0±6.2	56.7±8.0	x	x
3	97.6±7.8	97.4±8.5	99.1±1.1	99.1±1.4	✓	✓
4	78.9±4.6	79.8±7.2	77.6±4.7	76.9±6.5	x	x
5	95.8±2.6	95.2±3.1	95.8±2.6	95.2±3.1		
6	1±0	1±0	1±0	1±0		
7	92.5±4.1	91.5±6.1	92.9±3.7	94.7±4.6		✓
8	72.1±10.2	61.3±16.9	69.5±10.6	59.3±16.2	x	
9	92.0±4.7	90.4±7.0	89.6±5.0	89.7±6.7	x	
10	62.6±8.8	64.2±10.6	64.0±9.0	65.8±10.1		
11	73.3±5.7	76.6±6.9	72.5±5.1	76.7±6.0		
12	99.1±2.3	99.5±1.6	99.2±0.6	99.5±0.6		
13	68.2±10.2	67.4±11.9	71.0±10.4	73.6±11.0	✓	✓
14	95.4±2.5	96.3±2.5	96.2±2.5	97.6±2.1	✓	✓
15	86.4±14.2	85.1±17.9	83.4±14.0	63.5±22.3		x
16	98.0±10.9	84.6±13.1	98.6±0.8	94.8±5.6	✓	✓
17	95.2±1.4	92.6±3.5	96.7±1.2	95.1±3.1	✓	✓
18	71.4±12.4	61.5±20.8	68.9±11.6	59.8±21.3		
19	95.0±1.8	98.2±0.9	94.8±1.9	98.1±1.0		
20	1±0	1±0	1±0	1±0		
21	99.6±0.3	99.6±0.5	99.6±0.2	99.4±0.6		
22	96.8±0.9	93.3±4.7	96.8±0.2	95.1±6.9		✓
23	70.4±3.9	72.2±4.9	71.1±3.6	73.3±4.0		✓
24	99.5±0.2	98.9±1.4	99.5±0.1	99.3±0.7	✓	✓
25	98.9±1.8	94.2±19.4	99.5±0.3	98.5±1.8	✓	✓
M.	87.49	85.78	87.55	86.24		

Table 3 lists the accuracy of the chosen labelling and the AUC values of the whole set of optimal labellings. The first thing that can be observed is that the differences in accuracy are smaller than in AUC. In some cases it happens that Gain Ratio is better than AUCsplit in terms of accuracy, but not significantly in terms of AUC.

Since means of different datasets are illustrative but not reliable we compare dataset by dataset if one method is better than the other. The 'Better?' column represents if AUCsplit behaves better (✓) or worse (x) than Gain Ratio. These marks are only shown when the differences are significant according to the *t*-test with level of confidence 0.1. This gives 8 wins, 13 ties and 4 loses for accuracies and 11 wins, 11 ties and 3 loses for AUC.

In order to study the applicability of the AUCsplit for unbalanced datasets, we have selected the datasets with a percentage of the minority class less than 15%. Table 4 shows the accuracies of both methods (GainRatio and AUCsplit) with several test set distributions under the same experimental methodology as those shown in Table 3. The first two columns of Table 4 show the accuracy preserving the original class distribution for the test set. The new information appears in the next columns of Table 4. These show the accuracies if we modify the test set distribution to be 50% for both classes. Finally, we show the accuracies for the swapped class distributions (e.g. 10%-90% train distribution is swapped to 90%-10% test distribution).

Table 4. Accuracy results for unbalanced datasets.

#	ORIGINAL DIST.		50%-50%		SWAPPED DIST.		%MIN
	GR	AUCS.	GR	AUCS.	GR	AUCS.	CLASS
16	98.0	98.6	88.3	93.5	78.6	88.3	6.06
17	95.2	96.7	88.6	92.6	81.9	88.4	11.83
21	99.6	99.6	99.0	98.7	98.4	97.8	10.4
22	96.8	96.8	89.8	89.7	82.9	82.7	10.23
24	99.5	99.5	96.0	96.6	92.5	93.6	3.95
25	98.9	99.5	95.8	98.4	92.7	97.3	9.86
M.	98.0	98.5	92.9	94.9	87.8	91.4	

As we can see in Table 4, the difference in accuracy is small when train and test distributions are the same. In general, if a model learned with an unbalanced dataset is to be used with a distribution different from the train distribution, the accuracy decreases. However, the AUCsplit splitting criterion yields models whose accuracy decreases less than those obtained by GainRatio splitting criterion in these cases.

5. Conclusions and Future Work

We have reassessed the construction and evaluation of decision trees based on a very practical and direct way to compute the convex hull of the ROC curve of all the possible labellings of a decision tree. The cost of this operation is just $O(n \cdot \log n)$, for ordering n leaves of a tree according to their local positive accuracy. This gives a different perspective on decision tree learning, where just clustering trees are learned, and classes are assigned at application time.

Our approach to using only $n+1$ points is closely related to the ordering of decision tree leaves already presented in (Blockeel & Struyf 2001) and the ranking of predictions and its use for computing the AUC measure presented in (Hand & Till 2001). In comparison with Hand and Till's approach, their AUC measure is almost equivalent to ours (their area is step-like) but our node-based way of computing the AUC gives more insight and allows a direct implementation as splitting criterion. This leads to the first successful splitting criterion based on estimated probabilities we are aware of that is not a weighted average of the impurities of the children, and gives better results for the AUC measure and comparable results in terms of accuracy.

As future work, we plan to extend AUCsplit to more than 2 classes. For this, a simplified 1-point ROC curve could be used, or the generalised *M* function introduced by (Hand & Till 2001). This would only be feasible by using our node sorting technique, incurring a cost in $O(c^2 \cdot n \cdot \log n)$ where c is the number of classes and n the number of nodes.

Some other issues to be explored are the development of pre-pruning and post-pruning methods based on AUC, because accuracy-based pruning methods may counteract some of the AUCsplit benefits for the AUC measure. From a more general point of view, other subsets of the set Γ of optimal labellings or even Λ could be considered, or several smoothing methods could be applied to compute the AUC measure. The use of a validation set for estimating AUCsplit could also be examined.

A more ambitious approach would be the development of a global AUC search heuristic, which would compute the optimality of a split taking into account the leaves in the split but also all the other opened leaves of the tree. We think that a monotonic AUC-based heuristic could be derived, in order to implement an optimal AO* search.

Finally, we would like to point out that, while we have focused on decision trees in this paper, the results can be equally used with other learning methods that partition the instance space, such as CN2 or many ILP systems.

Acknowledgements

This work has been partially supported by CICYT under grant TIC2001-2705-C03-01, by Generalitat Valenciana under grant GV00-092-14, and by the EU project *Data Mining and Decision Support for Business Competitiveness: Solomon Virtual Enterprise* (IST-1999-11495). Two of the authors enjoyed two research stays in the Department of Computer Science of the University of Bristol, where this work was initiated during the last quarter of 2001. These stays were funded by Universitat Politècnica de València and by Generalitat Valenciana.

References

Blake, C., & Merz, C. (1998). UCI repository of machine learning databases (http://www.ics.uci.edu/~mlearn/MLRepository.html). University of California, Dept of Computer Science.

Blockeel, H., & Struyf, J. (2001). Frankenstein classifiers: Some experiments on the Sisyphus dataset, *in* C. Giraud-Carrier, N. Lavrac, and S. Moyle (eds.), *Integrating Aspects of Data Mining, Decision Support and Meta-Learning*, pp 1-12, ECML/PKDD'01 workshop notes.

Boissonat, J.D., & Yvinec, M. (1998). *Algorithmic Geometry*. Cambridge University Press.

Bradford, J., Kunz, C., Kohavi, R., Brunk, C., & Brodley, C. (1998). Pruning decision trees with misclassification costs, *in* H.Prade (ed.) *Proceedings of the European Conference on Machine Learning*, pp. 131-136.

Breiman, L., Friedman, J.H., Olshen, R.A., & Stone, C.J. (1984). *Classification and regression trees*, Belmont, CA, Wadsworth.

Drummond, C., & Holte, R.C. (2000). Exploiting the cost (in)sensitivity of decision tree splitting criteria, *in* Langley (ed.) *Proceedings of the Seventeenth International Conference on Machine Learning*, Morgan Kaufmann, pp. 239-246.

Elkan, C. (2001). The Foundations of Cost-Sensitive Learning, *in* B. Nebel (Ed.) *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, Seattle, Morgan Kaufmann, pp. 973-978.

Esposito, F., Malerba, D., & Semeraro, G. (1997). A Comparative Analysis of Methods for Pruning Decision Trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 5, pp. 476-491.

Ferri, C., Flach, P., & Hernández, J. (2002). *Rocking the ROC Analysis within Decision Trees* (Technical Report). Department of Computer Science, Bristol, UK.

Ferri, C., Hernández, J., & Ramírez, M.J. (2002). *SMILES: A Multi-purpose Learning System*. (Technical Report), Dep. Sistemes Informàtics i Computació, Univ. Pol. València. (http://www.dsic.upv.es/~flip/smiles/).

Hand, D.J., & Till, R.J. (2001). A Simple Generalisation of the Area Under the ROC Curve for Multiple Class Classification Problems, *Machine Learning*, 45, pp. 171-186.

Kearns, M., & Mansour, Y. (1996) On the boosting ability of top-down decision tree learning algorithms. *Journal of Computer and Systems Sciences*, 58(1), 1999, pp 109-128. *Also in Proceedings ACM Symposium on the Theory of Computing*, 1996, ACM Press, pp.459-468.

Provost, F., & Fawcett, T. (1997). Analysis and visualization of classifier performance: Comparison under imprecise class and cost distribution, *in* D. Heckerman, H. Mannila, D. Pregibon (eds.). *Proceedings of the 3rd Intl. Conf, on Knowledge Discovery and Data Mining* (KDD-97), Menlo Park, CA: AAAI Press, pp. 43-48.

Quinlan, J.R. (1987). Simplifying Decision Trees. *International Journal Man-Machine Studies*, vol. 27, pp. 221-234.

Quinlan, J.R. (1993) *C4.5. Programs for Machine Learning*, San Francisco, Morgan Kaufmann.

Swets, J., Dawes, R., & Monahan, J. (2000). Better decisions through science. *Scientific American*, October 2000, pp. 82-87.