Why Fuzzy Decision Trees are Good Rankers

Eyke Hüllermeier

Department of Mathematics and Computer Science

Marburg University

35032 Marburg, Germany

eyke@mathematik.uni-marburg.de

Stijn Vanderlooy

Department of Knowledge Engineering

Maastricht University

6200 Maastricht, The Netherlands

s.vanderlooy@micc.unimaas.nl

Draft of a paper to appear in IEEE Transactions on Fuzzy Systems

Abstract

Several fuzzy extensions of decision tree induction, an established machine learning method, have already been proposed in the literature. So far, however, fuzzy decision trees have almost exclusively been used for the performance task of classification. In this paper, we show that a fuzzy extension of decision trees is arguably more useful for another performance task, namely ranking. Roughly, the goal of ranking is to order a set of instances from most likely positive to most likely negative. The motivation for applying fuzzy decision trees to this problem originates from recent investigations of the ranking performance of conventional decision trees. These investigations will be continued and complemented in this paper. Our results reveal some properties which seem to be crucial for a good ranking performance—properties that are better and more naturally offered by fuzzy than by conventional decision trees. Most notably, a fuzzy decision tree produces scores in terms of membership degrees on a fine-granular scale. Using these membership degrees as a ranking criterion, a key problem of conventional decision trees is solved in an elegant way, namely the question of how to break ties between instances in the same leaf or, more generally, between equally-scored instances.

1 Introduction

Decision trees are one of the most extensively studied methods in machine learning and data mining. Several factors contribute to their popularity, notably the following. Decision trees are comprehensible and interpretable by the domain expert and can naturally handle different types of attributes (e.g., numerical and categorical). Moreover, they are intrinsic multi-class learners, scale comparatively well with the number of attributes and instances [24], and have an attribute selection mechanism built-in. Finally, decision trees have shown to obtain classification performance close to or even outperforming other state-of-the-art methods, especially when they are boosted or used in a different ensemble method [12]. The two most widely used implementations for decision trees are CART [4] and C4.5 [25].

Traditionally, decision trees are used in the common supervised classification setting where the goal is to find an accurate mapping from instance space to label space. However, in many applications it is not enough to predict the most likely class label for each test instance [6]. What is needed instead is, assuming a binary label space for simplicity, a total ordering of test instances from most likely positive to most likely negative. In this way, a *ranking* of test instances is obtained. For example, a credit-card fraud detection system may need to rank bank accounts according to the possibility of being compromised. The rank position of an instance depends on the score it receives. Obviously, the mapping from an instance to the probability of belonging to the positive class is a perfect *scoring function*, but so is any monotone transformation thereof. It follows that a good probability estimator is a good ranker, but not necessarily the other way around. Even more interestingly, improving accuracy does not necessarily decrease the number of ranking errors, and vice versa [11]. The standard performance metric for the supervised ranking setting is the area under the ROC curve, or AUC for short [3].

A decision tree, trained in the usual way as a classifier, can be used for ranking by scoring an instance in terms of the frequency of positive examples found in the leaf to which it is assigned. A few papers provide experiments showing that unpruned trees lead to better rankings (higher AUC values) than standard pruned trees [23, 10]. This is counterintuitive at first sight since it is well-known, at least for classification accuracy, that pruning prevents or alleviates the overfitting effect and strongly reduces variance. Several other enhancements to the trees have been proposed to improve their ranking performance, as we will review later.

Fuzzy decision trees are more advanced in the sense that they model uncertainty around the split values of the features, resulting in soft instead of hard splits. Moreover, they naturally produce scores in the form of membership degrees. Several papers have compared decision trees with their fuzzy variants, but always in terms of classification accuracy; see for example [19] and references therein. Yet, even though some gains have occasionally been reported, it is still unclear whether fuzzy decision trees can systematically and significantly outperform non-fuzzy trees in terms of classification accuracy.

In this paper, we argue that a fuzzification of decision trees is potentially more useful for the problem of ranking, that is, if performance is measured in terms of AUC instead of classification accuracy. The main aim of the paper is to corroborate this claim by offering convincing explanations for the good ranking performance of fuzzy decision trees. To this end, we first readdress the empirical observations concerning conventional (non-fuzzy trees) in a systematic way. Afterward, we present a formal analysis which, for the first time, shows a close connection between the AUC and the number of distinct scores assigned to test instances. All previous (and sometimes surprising) observations can be explained by our analysis. As a consequence of these findings, we conclude that fuzzy decision trees can be expected to produce good rankings, and we conduct an extensive set of new experiments on benchmark data sets to verify this conjecture. More specifically, we show that fuzzy trees can consistently outperform the best methods for AUC-optimizing trees while still maintaining the benefits of decision trees such as comprehensibility and interpretability.

The remainder of the paper is organized as follows. Background and notation is given in Section 2. In Section 3, we discuss related work from the machine learning literature. Afterward, in Section 4, we provide a first set of experiments that verify, correct, and extend earlier empirical results. Inspired by these results, we provide our formal analysis in Section 5. In Section 6, we show that fuzzy decision trees are very good rankers. Section 7 concludes the paper.

2 Background and Notation

In this section, we briefly explain the bipartite ranking problem, ROC curves, and the AUC. Moreover, we provide a short note on the Laplace correction. We assume that the reader is familiar with the basic concepts of decision tree learning.

2.1 Bipartite Ranking Problem

Consider an instance space $\mathcal{X}$ and let the sample space $\mathcal{X} \times \{-1, +1\}$ be endowed with a probability measure $\mathbb{P}$; thus, $\mathbb{P}(\boldsymbol{x}, c)$ denotes the probability to observe instance $\boldsymbol{x}$ with class label c. An instance $\boldsymbol{x}$ with class label $+1$ (-1) is called a positive (negative) instance. A decision tree f is considered as an $\mathcal{X} \to [0,1]$ mapping and $f(\boldsymbol{x})$ is interpreted as the probability or, more generally, as a degree of confidence that the class label of $\boldsymbol{x}$ is $+1$. Henceforth, we call $f(\boldsymbol{x})$ the *score* of instance $\boldsymbol{x}$ and we score an instance in terms of the frequency of positive examples found in the leaf to which it is assigned.. The goal is to use the scores in order to rank instances such that the positives are ranked higher than the negatives.

2.2 ROC Curves and the AUC

A natural performance metric for ranking is the probability of the event $f(\boldsymbol{x}) > f(\boldsymbol{y})$ given that $\boldsymbol{x}$ is a positive instance and $\boldsymbol{y}$ is a negative instance, both independently drawn from the sample space according to $\mathbb{P}(\cdot)$. Empirically, this probability has to be estimated from a sample $S = \{(\boldsymbol{x}_i, c_i)\}_{i=1}^n \subseteq (\mathcal{X} \times \{-1,+1\})^n$. An unbiased estimator is the Wilcoxon-Mann-Whitney statistic, which is given by the fraction of pairs $(\boldsymbol{x}_i, \boldsymbol{x}_j)$, with $\boldsymbol{x}_i$ a positive instance and $\boldsymbol{x}_j$ a negative instance such that $f(\boldsymbol{x}_i) > f(\boldsymbol{x}_j)$ [3]. So, we simply count the number of pairs of instances that are correctly ordered. In case of a tie in the scores, $f(\boldsymbol{x}_i) = f(\boldsymbol{x}_j)$, the instance pair is counted with 1/2 instead of 1.

Interestingly, the above statistic is equivalent to the area under the ROC curve, or AUC for short [3]. The output of a decision tree f has to be thresholded in order to produce a crisp classification (i.e., to decide whether the instance is a positive or a negative one). When the score is above some threshold $t \in [0,1]$, then we classify the instance as positive; otherwise as negative. A positive instance that is correctly classified is called a true positive. If it is classified as negative, then it is called a false negative. A true negative and a false positive are defined analogously. The total number of true positives, false positives, true negatives, and false negatives are denoted by TP, FP, TN, and FN, respectively. From these numbers we derive

$$\begin{aligned} tpr &= \frac{TP}{TP+FN}\ , \ tnr = \frac{TN}{TN+FP}\ , \\ fpr &= \frac{FP}{FP+TN}\ , \ fnr = \frac{FN}{TP+FN}\ , \end{aligned}$$

where true positive rate is denoted by tpr and true negative rate by tnr. We note that $fpr = 1-tnr$ is the false positive rate and $fnr = 1-tpr$ is the false negative rate. Now, an ROC curve visualizes these performances across all possible classification thresholds. So, if there are s different scores assigned to instances in the sample S, then there are $s+1$ thresholding ranges, each resulting in different fpr and tpr values. The connection of these points in the (fpr, tpr) plane results in a piecewise linear curve, which is called the ROC curve. Since a non-convex ROC curve indicates a suboptimal behavior that can easily be repaired, the convex hull of the ROC curve can be used as reference without loss of generality [8]. Fig. 1 gives an illustration.

For decision trees, it has been shown that the ROC convex hull can be generated efficiently [9]. Each leaf corresponds to a line segment of the curve, with slope equal to the class distribution in that leaf. Hence, ordering the leaves according to their class distributions (i.e., the scores) generates an ROC convex hull. The AUC is then equivalent to the sum of the area of the trapezoids defined by the line segments of the convex hull.

2.3 Laplace Correction

The so-called *Laplace correction* is often used in probability estimation. It simply adds a pseudo-count of 1 to the relative frequencies of all classes. For a decision tree, this means that, given p positive examples and n negative examples in a leaf, the estimated probability for the positive class is

$$\frac{p+1}{p+n+2}\ . \tag{1}$$

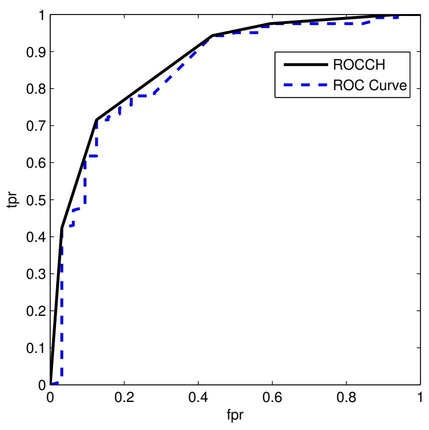


Figure 1: An empirical ROC curve and its ROC convex hull (ROCCH). Note that an ROC curve is always dominated by its convex hull.

Roughly speaking, Laplace correction yields a more cautious estimation, which is biased toward the middle (probability 1/2) and avoids extreme estimates; in fact, neither 0 nor 1 can be produced as an estimate (though convergence toward these values is still possible for large sample sizes). Even though the Laplace correction appears to be ad-hoc at first sight, (1) can be derived from a uniform prior on $[0, 1]$ as a Bayes estimate of the success parameter of a binomial distribution [2, p. 71-74].

3 AUC-Optimizing Decision Trees

Relative class frequencies in leaf nodes of a decision tree have been shown to produce poor scores [26, 15, 31]. Even though, as mentioned earlier, good probability estimation is only a sufficient but not a necessary precondition, this suggests that the learning strategy of decision trees is not suitable for learning good rankers. Several researchers have therefore tried to improve the ranking performance of decision trees; see for example [9, 10, 27, 32, 5]. In this paper, we restrict ourselves to improvements involving a single decision tree, and hence do not consider ensemble methods such as bagging, which produce incomprehensible results. Our special interest concerns the frequently cited paper [23], in which the authors propose two enhancements to improve the AUC of decision trees. They call the resulting method C4.4 and show empirically that it leads to higher test AUC values than the standard C4.5 learner, that was used as a baseline. The two proposed enhancements are the following:

- First, learn unpruned trees by turning off error-based pruning and node collapsing in C4.5. The stopping criterion for tree building is then that either the number of instances in the leaf is at most two, or all the features have been exhausted on the current path. The main idea is that the branches removed by the pruning step might still be useful for ranking. They allow for assigning different scores to instances contained in a hyper-rectangle that is considered good enough for classification purposes.[1] A drawback of this technique is that unpruned decision trees may become extremely large.

- Second, smooth the class frequencies in a leaf using Laplace correction. An unpruned tree will often produce leaves with low *cardinality* (i.e., the leaves comprise only a small number of training instances) and, hence, the tree will produce unreliable scores. The most compelling example is a pure leaf, that is, a leaf consisting of only positive or only negative examples. Such a leaf produces a score of either 0 or 1, indicating that it is most certain of its output. Laplace correction makes the scores from small leaves less extreme and, as conjectured by the authors, hereby more accurate and reliable. The win-loss-equal statistics that they obtained show that most of the improvement of C4.4 over C4.5 is due to the Laplace correction. Interestingly, a generalized version of the Laplace correction, called m-estimate, was found to not produce higher test AUC values [10].[2]

In a response to these empirical results, [16] mentions the following two possible disadvantages of C4.4. First, the unpruned tree will probably overfit, and this may

[1]A decision tree divides the input space by means of hyper-rectangles since, at each node, a feature test induces a decision boundary perpendicular to the corresponding axis.

[2]Note that Laplace correction does not change the accuracy of a decision tree, except when the score is 0.5 (which is in turn very unlikely due to the tree learning strategy).

cause unreliable class frequencies in the leaves. Second, the number of training examples in the leaves of an unpruned tree is often (very) small. It follows that these leaves are likely to give the same score to test instances which are very different from each other in the sense of following different branches in the tree. To overcome these possible disadvantages, the authors propose the following method to improve ranking performance. The tree building strategy is left unchanged, i.e., the learning algorithm C4.5 is used in its standard form. However, in the prediction phase, a test instance is propagated along all branches emerging from a node, with a smaller weight for the branches not satisfying the test predicate in that node. The weights are multiplied along the paths toward leaves and corresponding predictions are averaged. So, each leaf contributes to the overall score of a test instance and the weight of the contribution is determined by the number of feature tests that failed on the path from root to leaf. As a rationale of this approach, the authors explain that it is natural to expect a small chance that feature values are altered due to noise or errors in the training set. Hence, different paths in the tree should be explored. Experimental results show that this technique is able to outperform C4.4 when the pruning level and a second parameter controlling the weight of each branch are set appropriately. Yet, only six very small data sets are used in these experiments. The method has not been used afterward or further investigated, maybe also because a formal analysis of its effectiveness is missing (in fact, as we will see later, the rationale of the approach is not the true reason for its improved AUC). Nonetheless, at least in the prediction phase, some connections with fuzzy decision trees exist, since an instance can be propagated simultaneously along multiple paths.

4 Experimental Analysis of AUC-Optimizing Trees

In this section, we present a first set of experiments that elaborate on some details and (implicit) assumptions of the aforementioned two methods. More specifically, we investigate the effect of three factors that have been considered to be influential for the AUC of decision trees. We start by explaining the setup of the experiments.

4.1 Experimental Setup

We used twenty binary classification data sets from the UCI benchmark repository [1]. The data sets vary strongly in size, number and type of features, and in class distribution. Their most important characteristics are given in Table 1. The tree induction algorithm that we use is the implementation of C4.5 provided by the `WEKA` machine learning package [30]. This package is open-source and for free available (`http://www.cs.waikato.ac.nz/ml/weka/`).

The reported values of the AUC (or any other test statistic to our interest) are obtained as follows. We apply a 10-fold stratified cross validation procedure, and on each training fold we learn ten decision trees with different pruning levels. The pruning level is controlled by the *confidence factor*, a parameter of the learning algorithm. C4.5 allows for confidence factors between 0 and 0.5, where smaller values incur more pruning. The confidence factor of decision tree $\#i \in \{1, \ldots, 9\}$ is set equal to $i/20$, and the tenth decision tree is a completely unpruned tree (pruning turned off, confidence factor is irrelevant). Since the default value of the confidence factor in C4.5 is 0.25, a standard pruned tree (i.e., a conventional decision tree) in earlier experiments is our decision tree #5. For each pruning level, the values of the test statistic on the ten test sets are recorded and averaged. This procedure is repeated twenty times and each time the data set is randomly shuffled. We report the estimation of the expected value of the test statistic in terms of the mean, accompanied by the estimated standard deviation.

To analyze some of the results more thoroughly, we applied the Wilcoxon signed-ranks test as recommended in [7]. The test looks at the difference between the test statistic values of two methods on each data set. These differences are ranked according to their absolute values; average ranks are assigned in case of ties. The

Table 1: The twenty data sets: (1) reference number, (2) name, (3) number of instances, (4) number of nominal features, (5) number of numerical features, and (6) percentage of the majority class.

#	name	size	nom	num	% maj class
1	adult	48842	7	6	76.07
2	breast cancer	286	9	0	70.28
3	breast wisconsin	699	0	8	65.52
4	caravan	5822	85	0	65.52
5	credit rating	690	9	6	55.51
6	german credit	1000	13	7	69.40
7	heart statlog	270	7	6	59.50
8	horse colic	368	14	7	63.04
9	house votes	435	16	0	38.62
10	ionosphere	351	0	34	35.90
11	kr vs kp	3196	36	0	52.22
12	liver	345	1	5	42.03
13	monks1	556	6	0	50.00
14	monks2	604	6	0	65.72
15	pima	768	0	8	65.10
16	sick	3772	22	7	93.16
17	sonar	208	0	60	53.36
18	spambase	4601	0	57	61.00
19	spect	267	22	0	58.80
20	tic-tac-toe	958	9	0	65.34

test statistic is then given by the difference between the sum of ranks for the data sets on which the second method outperformed the first and the sum of ranks for the opposite case.

4.2 Dependence of AUC on Pruning Level

Unpruned trees with Laplace correction have been shown to consistently outperform standard pruned trees in terms of AUC [23]. However, the effect of decreasing the pruning level has not been tested and it was implicitly assumed to be monotone. Also, no tests have been performed comparing different pruning levels for a decision tree without Laplace correction. So, to complement earlier experiments, we show in Fig. 2(a) the typical behavior of the AUC obtained by the ten decision trees for four representative data sets.[3] The pruning level is decreasing from left to right, i.e., left is highly pruned and right is completely unpruned. Solid and dashed curves show the result for decision trees, respectively, with and without Laplace correction. We can make three interesting observations from these illustrations:

- First, comparing dashed and solid curves, it can be seen that Laplace correction applied in the leaf nodes never decreases the AUC, and hence, Laplace-corrected decision trees significantly outperform conventional trees. We therefore strongly advocate the use of Laplace correction.

- Second, in conjunction with Laplace correction, unpruned trees indeed produce better rankings than the trees pruned at the default level. The Wilcoxon signed-ranks test shows that we can reject the null hypothesis (equal ranking performance of unpruned trees and standard pruned trees) at the 1% significance level. However, we note that the value of the AUC is not always a monotone decreasing function of the pruning level, or equivalently, a monotone increasing function of the depth of the tree. Therefore, it seems to be a good idea to try out several pruning levels below the standard value used for classification purposes.

- Third, no clear trend can be identified when inspecting the dashed curves only (trees without Laplace correction), although in general the standard pruned trees have higher AUC values than unpruned trees. Nonetheless, when using the Wilcoxon signed-ranks test, we do not have a statistically significant difference at the 5% significance level due to some data sets where the unpruned tree outperforms with a large margin the standard pruned tree.

Table 2 summarizes these three findings for all data sets by reporting the AUC of decision trees #1, #5, and #10. A fourth column indicates where the highest

[3]These data sets are representative in the sense that the shape of the curves of the test statistics and the corresponding differences in absolute values represent the other data sets as well.

AUC occurs. The results verify that the best rankers are decision trees with Laplace correction and, except for a few exceptions, with less pruning than is commonly used for classification purposes.

4.3 The Effect of Laplace Correction

The above results show that Laplace correction has an important effect; in fact, it yields the largest improvement in AUC. In earlier work, this effect was attributed to an increased reliability, i.e., scores of small leaves are presumably less reliable estimates of the true probabilities of the positive class as are scores of large leaves. This distinction is especially considered to be important when extreme scores are produced by small leaves since corresponding instances are ranked at the very ends of the list of instances, and hereby, may incur large ranking errors.

More specifically, consider a leaf with p positive and n negative examples, a second one with corresponding values p' and n'. Moreover, suppose that $p > n$, and that both leaves have the same score $s_1 = p/(p+n) = p'/(p'+n') = s_2$. The corresponding Laplace-corrected scores are $s_1' = (p+1)/(p+n+2)$ and $s_2' = (p'+1)/(p'+n'+2)$, respectively, which means that $s_1' > s_2'$ if and only if $p+n > p'+n'$. In other words, Laplace correction breaks ties between leaves with the same score in favor of the larger one. For example, an instance falling in a leaf with 40 positive and 10 negative examples will be ranked ahead of an instance in a leaf with only 4 positives and 1 negative. From a statistical point of view, this is clearly reasonable, since the prediction of the positive class is arguably more reliable for the former than for the latter. Analogous corrections are made in the case where $p < n$, i.e., for presumably negative leaf nodes.

It is worth noting that, apart from breaking ties between equally scored instances, Laplance correction may also produce a reversal of scores. Thus, it may happen that $s_1 > s_2$ while $s_1' < s_2'$ for the corrected scores, a property which is perhaps less desirable. In particular, instances falling into pure leaf nodes with only positive (negative) instances should still be ranked above (below) the instances falling into non-pure leaves. This is however not guaranteed, especially in the case of small nodes (the score of a pure leaf with only 2 positive examples, for example, would be strongly reduced from 1 to 3/4). To elaborate on this, we tested whether Laplace correction changes the rank position of extremely scored instances with respect to instances that do not fall into pure leaves. In our experiments, we have found that this event is unlikely to occur, and when it still does, then the change in rank position is on average very small; see [13] for the details of the results. Thus, it seems that the main effect of Laplace correction is *local tie breaking*: The ranking of instances is refined by resolving ties between instances having the same score, i.e., by turning some equalities into strict order relations, but mostly without swapping the position of instances. We conjecture that this effect is the true benefit of Laplace correction.

4.4 The Number of Distinct Scores

As a last experiment, we investigate a conjecture of [16], namely that large trees are only able to produce a small number of different scores. Instances with equal score are potentially disadvantageous for AUC because they can at most contribute a count of 1/2. Hence, to obtain more diversity among the instances, a pruned tree should be preferred. The main observation here is that the cardinality of a leaf determines the number of scores it can produce, i.e., a leaf with m examples can only produce $m+1$ different (uncorrected) scores, namely $0, 1/m, 2/m, \ldots, 1$.

However, we conjecture that the net effect of pruning on the total number of scores produced by a tree is not clear. On the one hand, pruning indeed increases the cardinality of the leaves (and, hence, reduces the probability that two leaves have the same score). On the other hand, however, it also reduces the number of leaves. In any case, it is clear that Laplace correction increases the number of scores since two leaves produce the same score only when the absolute numbers of positive and negative instances are identical (and not only the relative frequencies).

Fig. 2(c) shows the typical behavior of the number of different scores for the four representative data sets, and Table 3 summarizes the findings for all data sets and decision trees #1, #5, and #10. We make the following two observations from these illustrations and statistics, depending on the type of features in the data set:

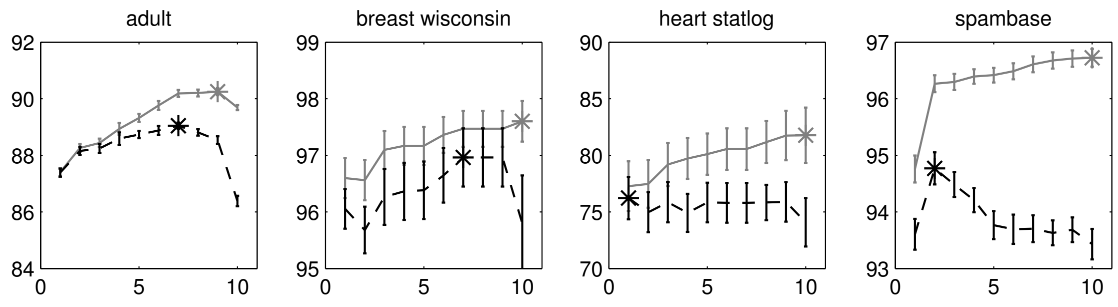
- First, for the data sets with only numerical features, the number of scores is almost always non-decreasing in the pruning level (since a numerical feature can be tested repeatedly on a path). When we have a decrease, then this decrease is very small, often limited to low pruning levels, and only happens in case of no Laplace correction. This shows that a few leaves can indeed start to produce the same scores, but Laplace correction breaks the ties.

- Second, even for data sets with only nominal features we can have an increase in scores along all pruning levels (e.g., `spect` and `house votes`). When this is not the case, we often see that the increase is up to a certain pruning level. After that level, trees with Laplace correction slightly decrease the number of scores (this happened only on four data sets). Without Laplace correction, the decrease is of course stronger.

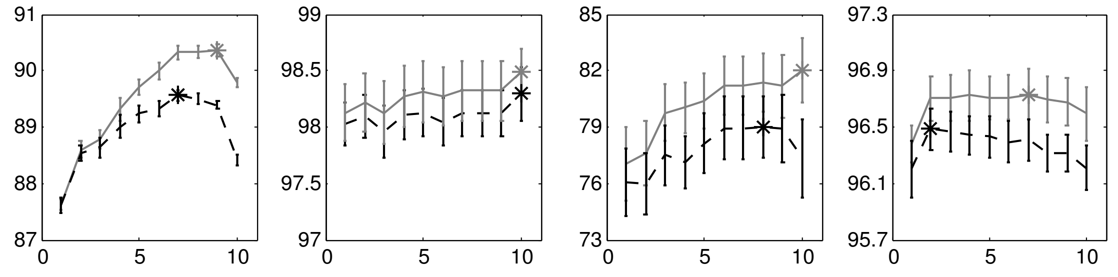
Independent of the aforementioned two observations, it is interesting to compare Figs. 2(a) and 2(c), suggesting that the decision tree that produces the best rankings is the one that assigns many different scores to the test instances.

4.5 Conclusions from the Experimental Results

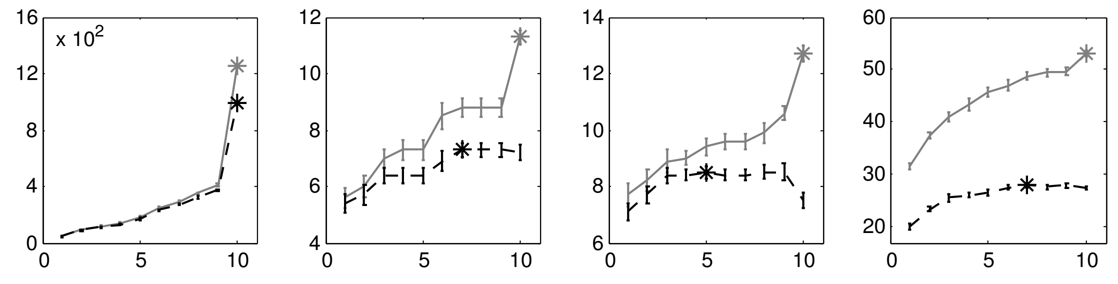
With this set of experiments we have verified earlier results in more detail and we tested some issues that have not been carefully considered before. Summarizing the results, we confirm that Laplace correction is always beneficial and that unpruned trees almost always have higher AUC than standard pruned trees (at least with Laplace correction). In addition, our results show that a slightly pruned tree can have higher AUC than an unpruned tree, at least when it produces more distinct



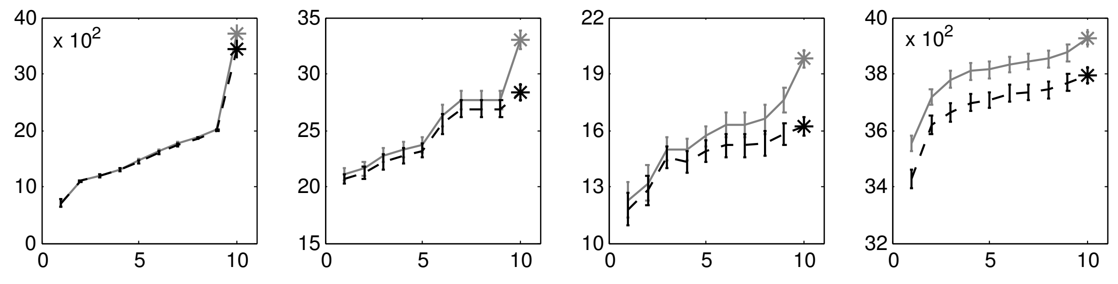
(a)



(b)



(c)



(d)

Figure 2: Test statistics of the ten decision trees. Solid and dashed curves represent trees, respectively, with and without Laplace correction. An asterisk indicates the first tree with the highest test statistic, which is the AUC of C4.5 and the perturbation method (a - b) and corresponding number of different scores (c - d).

Table 2: The average AUC and standard deviations of decision trees #1, #5, and #10. Results are shown in two parts: without and with Laplace correction. The fourth column of each part lists the decision trees that obtain the highest AUC.

#	without Laplace correction				with Laplace correction			
	1	5	10	max	1	5	10	max
1	.8739 ± .0014	.8874 ± .0013	.8638 ± .0019	7	.8741 ± .0015	.8932 ± .0014	.8969 ± .0009	9
2	.5691 ± .0176	.5942 ± .0285	.5830 ± .0246	2-3	.5691 ± .0176	.5933 ± .0278	.6127 ± .0210	2-3
3	.9605 ± .0035	.9638 ± .0051	.9580 ± .0084	7-9	.9660 ± .0035	.9717 ± .0034	.9760 ± .0036	10
4	.5000 ± .0000	.4997 ± .0004	.5777 ± .0091	10	.5000 ± .0000	.4990 ± .0005	.6997 ± .0093	10
5	.8915 ± .0057	.8940 ± .0063	.8530 ± .0121	5-6	.8929 ± .0060	.9079 ± .0055	.9022 ± .0034	8
6	.6261 ± .0260	.6467 ± .0148	.6253 ± .0142	2	.6323 ± .0261	.7100 ± .0084	.7090 ± .0108	2
7	.7622 ± .0188	.7583 ± .0176	.7408 ± .0215	1	.7728 ± .0219	.8011 ± .0182	.8178 ± .0245	10
8	.8517 ± .0192	.8517 ± .0192	.8414 ± .0167	1-5	.8517 ± .0192	.8517 ± .0192	.8617 ± .0158	10
9	.9670 ± .0082	.9797 ± .0073	.9762 ± .0076	2	.9728 ± .0061	.9852 ± .0041	.9875 ± .0043	10
10	.8663 ± .0170	.8660 ± .0150	.8822 ± .0141	10	.8998 ± .0132	.9083 ± .0137	.9126 ± .0144	10
11	.9948 ± .0013	.9967 ± .0012	.9969 ± .0012	10	.9967 ± .0005	.9988 ± .0003	.9991 ± .0002	10
12	.5753 ± .0152	.6251 ± .0150	.6423 ± .0144	10	.5723 ± .0145	.6406 ± .0128	.6537 ± .0139	10
13	.9233 ± .0145	.9823 ± .0054	.9804 ± .0051	9	.9233 ± .0145	.9823 ± .0054	.9858 ± .0037	9
14	.5000 ± .0000	.5356 ± .0168	.6532 ± .0103	8	.5000 ± .0000	.5388 ± .0179	.6742 ± .0099	8
15	.7412 ± .0145	.7558 ± .0084	.7546 ± .0102	4	.7533 ± .0134	.7805 ± .0098	.7870 ± .0104	10
16	.9468 ± .0088	.9523 ± .0075	.9512 ± .0089	8-9	.9441 ± .0101	.9704 ± .0076	.9917 ± .0027	8-10
17	.7344 ± .0197	.7439 ± .0222	.7416 ± .0213	2-3	.7841 ± .0160	.7814 ± .0145	.7851 ± .0156	2-3
18	.9360 ± .0027	.9377 ± .0025	.9343 ± .0027	2	.9476 ± .0024	.9642 ± .0013	.9672 ± .0016	10
19	.7112 ± .0215	.6626 ± .0268	.7033 ± .0210	1	.7191 ± .0206	.7121 ± .0188	.7325 ± .0210	7
20	.8352 ± .0136	.9117 ± .0049	.8966 ± .0084	9	.8314 ± .0133	.9229 ± .0042	.9374 ± .0035	10

Table 3: The average number of distinct scores and standard deviations of decision trees #1, #5, and #10. Results are shown in two parts: without and with Laplace correction. The fourth column of each part lists the decision trees that produce the most distinct scores.

#	without Laplace correction				with Laplace correction			
	1	5	10	max	1	5	10	max
1	46.3 ± 3.1	168.5 ± 6.9	989.7 ± 6.0	10	50.4 ± 2.9	182.5 ± 7.0	1251.8 ± 9.5	10
2	3.1 ± 0.3	4.2 ± 0.2	10.7 ± 0.4	10	3.1 ± 0.3	4.2 ± 0.2	15.3 ± 0.4	10
3	5.4 ± 0.3	6.4 ± 0.3	7.2 ± 0.3	7	5.6 ± 0.4	7.3 ± 0.4	11.3 ± 0.2	10
4	1.0 ± 0.0	1.5 ± 0.2	42.8 ± 1.2	10	1.0 ± 0.0	1.9 ± 0.4	78.2 ± 1.4	10
5	5.3 ± 0.4	9.2 ± 0.5	16.9 ± 0.5	10	5.4 ± 0.4	9.3 ± 0.5	25.5 ± 1.3	10
6	6.8 ± 1.2	20.2 ± 0.3	17.6 ± 0.5	9	6.7 ± 1.1	23.7 ± 0.3	27.9 ± 0.7	10
7	7.1 ± 0.3	8.5 ± 0.2	7.5 ± 0.3	5	7.7 ± 0.4	9.4 ± 0.3	12.7 ± 0.3	10
8	3.9 ± 0.1	4.2 ± 0.2	25.9 ± 0.9	10	3.9 ± 0.1	4.3 ± 0.2	28.0 ± 0.8	10
9	4.5 ± 0.3	5.5 ± 0.2	8.8 ± 0.3	10	4.5 ± 0.3	5.5 ± 0.2	9.1 ± 0.3	10
10	4.0 ± 0.1	4.3 ± 0.2	4.4 ± 0.2	8	6.2 ± 0.3	7.2 ± 0.3	7.6 ± 0.3	10
11	7.5 ± 0.2	6.3 ± 0.2	4.5 ± 0.3	1	20.0 ± 0.3	22.4 ± 0.5	24.8 ± 0.5	10
12	6.6 ± 1.1	11.5 ± 0.6	11.2 ± 0.5	3	6.6 ± 1.1	16.6 ± 0.4	17.3 ± 0.4	10
13	3.9 ± 0.3	3.0 ± 0.2	4.7 ± 0.4	10	12.2 ± 0.6	15.1 ± 0.5	20.6 ± 0.6	10
14	1.0 ± 0.0	4.6 ± 1.4	13.7 ± 0.5	8	1.0 ± 0.0	6.8 ± 2.2	21.2 ± 0.4	9
15	9.3 ± 0.6	11.1 ± 0.6	11.6 ± 0.4	8	10.4 ± 0.9	13.1 ± 0.9	14.8 ± 0.8	10
16	10.9 ± 0.7	15.1 ± 0.6	16.8 ± 0.5	8	14.2 ± 1.4	22.6 ± 1.2	28.3 ± 0.6	10
17	4.5 ± 0.1	4.3 ± 0.1	4.2 ± 0.1	2	7.9 ± 0.3	8.0 ± 0.3	8.1 ± 0.3	10
18	20.1 ± 0.5	26.6 ± 0.6	27.5 ± 0.4	7	31.7 ± 0.6	45.8 ± 0.9	53.1 ± 0.7	10
19	3.4 ± 0.2	6.1 ± 0.5	9.5 ± 0.3	10	3.4 ± 0.2	6.3 ± 0.5	13.4 ± 0.6	10
20	13.9 ± 0.6	14.4 ± 0.4	11.1 ± 0.3	7	25.5 ± 1.4	33.2 ± 0.5	32.9 ± 0.5	9

scores. We may conclude that there seems to be a strong positive correlation between number of scores and AUC. In the next section, we present a formal analysis that indeed shows that a higher number of possible scores (implying less ties among ranked instances) is likely to result in a higher AUC.

5 Formal Analysis of AUC-Optimizing Trees

We will mainly adopt a geometric perspective, as we think that it greatly facilitates the understanding. Given a set of instances S, the ROC curve of a decision tree is constructed as follows. Let the leaves be numbered in decreasing order according to their score and denote by L_i the i-th leaf or, depending on the context, the set of (training) examples in the i-th leaf ($i = 1, \ldots, m$). With p_i and n_i denoting, respectively, the number of positive and negative examples in L_i, the score of this leaf is $s_i = p_i/(p_i + n_i)$. Let $P = p_1 + \ldots + p_m$ be the total number of positive instances in S and, analogously, we define $N = n_1 + \ldots + n_m$ as the number of negative instances in S. Now, each leaf L_i contributes a segment S_i with slope $d_i = (p_i \cdot N)/(n_i \cdot P)$, and lengths $\Delta x_i = n_i/N$ and $\Delta y_i = p_i/P$ in the directions of the abscissa and ordinate, respectively. The concatenation of these ordered segments leads to a piecewise linear convex curve, which is the ROC convex hull. In general, we denote the concatenation of segments S_a and S_b by $S_a|S_b$.

Now, suppose that a leaf L_i is further split into two leaves L_{i1} and L_{i2}. Without loss of generality, we assume that $d_{i1} = p_{i1}/n_{i1} \geq p_{i2}/n_{i2} = d_{i2}$, where p_{i1} is the number of positive examples in L_{i1} and the other quantities are defined analogously. In the original ROC curve, the segment S_i is replaced by two segments S_{i1} and S_{i2} such that S_{i1} shares the starting point and S_{i2} the end point of S_i. Obviously, the curve thus obtained dominates the original curve, since S_i runs below $S_{i1}|S_{i2}$ while the rest of the curve remains unchanged. Therefore, the area under the modified curve is larger than the area under the original curve (or at least equal if $d_{i1} = d_{i2}$). The new area, however, may not correspond to the AUC of the modified tree since the segments $S_1, \ldots, S_{i-1}, S_{i1}, S_{i2}, S_{i+1}, \ldots, S_m$ are not necessarily well-ordered, which means that at least one of the conditions $d_{i-1} \geq d_{i1}$ and $d_{i2} \geq d_{i+1}$ might be violated. Hence, the question arises whether a locally beneficial modification has also a positive global effect on the AUC.

Lemma 5.1. *Consider a piecewise linear continuous curve consisting of m segments $S_1, \ldots, S_m$ and let $d_j \geq 0$ be the slope of S_j. If $d_{j-1} < d_j$, then swapping segments S_{j-1} and S_j can only increase the area under the curve.*

Proof. Since the rest of the curve remains unaffected, it suffices to consider the change of the area of the intersection between the area under the original curve and the rectangle defined by the diagonal that starts in the starting point of S_{j-1} and ends in the endpoint of S_j. Thus, subsequent to a suitable transformation, we can assume without loss of generality that S_{j-1} starts in $(0, 0)$ and S_j ends in $(1, 1)$. With (a, b) being the point where the two segments meet, we must have $b \leq a$ since the slope of S_{j-1} is smaller than the slope of S_j, which means that (a, b) must be located below the diagonal. After swapping the two segments, they will meet in the point $(1 - a, 1 - b)$, which is then located above the diagonal. Thus, $S_j|S_{j-1}$ runs above $S_{j-1}|S_j$ and, therefore, the area under the curve can only increase. □

Theorem 5.2. *Splitting a leaf can only increase the empirical AUC of a decision tree.*

Proof. We have seen that splitting a leaf L_i and replacing the corresponding segment S_i by S_{i1} and S_{i2} (in the correct order) leads to a curve that runs above the original ROC curve and, therefore, covers a larger area. The new AUC is given by the area under the curve that is obtained after re-ordering the segments in decreasing order according to their slopes. This re-ordering can be achieved by repeatedly swapping the segment S_{i1} with its left neighbour and, likewise, repeatedly swapping the segment S_{i2} with its right neighbour. The previous lemma has shown that, in each of these steps, the area under the curve can only increase. Thus, the theorem immediately follows by induction over the number of swaps that are needed to bring the segments into a proper ordering. □

As shown by the previous result, a local improvement of a single segment is also globally beneficial in terms of AUC. Restrictively, however, one has to consider that our result refers to the *empirical* AUC and not the true AUC. In fact, the score associated with a leaf as well as the slope of the corresponding segment are only estimations that may deviate from the real values. In the true ROC curve of a decision tree, the length of a leaf's segment corresponds to the *probability* that an instance falls into that leaf, which is estimated by the relative frequency in the training set. Likewise, the true slope δ_i of the segment is determined by the conditional probabilities of positive and negative instances in the leaf, and therefore may deviate from the estimation d_i derived from relative frequencies. As a result, the ordering of the segments according to the d_i may not coincide with the ordering according to the δ_i. In this case, the true ROC curve is non-convex and, therefore, the ranking performance in terms of AUC suboptimal. Formally, a non-convexity is caused by an *inversion*, that is, a pair of leaves (L_i, L_j) such that $d_i > d_j$ but $\delta_i < \delta_j$.

Nonetheless, even by looking at the problem from this point of view, there are two other convincing explanations for our finding that, in general, having more scores is better than having less:

- First, the ROC curve that can be obtained by splitting a segment, and hence increasing the number of scores, is at least *potentially* better than the original curve. Roughly speaking, the more small segments are available (i.e., the less ties among ranked instances), the more convex the curve can become. To illustrate, consider the two extreme cases: If there is only a single leaf, the ROC curve consists of a single segment, namely the diagonal, and the AUC is

0.5. On the other hand, if each instance is comprised by a single leaf, the AUC can become arbitrarily close to 1, given that the leaves are correctly ordered.

- Second, the main pitfall preventing from a high AUC in the case of many leaves is inversions due to wrongly estimated scores, which may turn potential convexities into actual concavities. However, given the assumption that the estimation errors are bounded, inversions are likely to have a more local effect, i.e., they concern only leaves that are "neighbored" in terms of their scores. Again, this means that having more scores is better, as it increases the probability that two leaves are well-ordered (there is a smaller probability that two randomly chosen leaves are neighbored and, therefore, potentially inverted).

To illustrate and further investigate these two points, we conduct a simple but conclusive experiment on synthetic data. The results will show that our formal analysis also applies to the true AUC with high probability, even when estimation errors are quite large. We generate a random binary list of 1000 elements where a 1 (0) indicates that the corresponding instance is positive (negative). We assume that these instances belong to the same leaf and start applying an iterative procedure that takes a random "leaf" and splits it in two. A fixed success parameter $u \in [0.5, 1]$ is used to arrange the instances in the new leaves such that the average fraction of positives in one leaf is u and the average fraction of negatives in the other leaf is also u. At each iteration we compute the true and empirical AUC. By definition, the true AUC is determined by the frequency of positives in all leaves and the empirical AUC is determined by adding noise to each of these frequencies. Noise is randomly drawn from an interval $[-h, +h]$, so if a true frequency is a, then its estimation lies in $[a - h, a + h]$. We stop the leaf splitting iteration when there is no leaf left comprising more than two instances. The complete process is repeated for twenty times and we report average test statistic values. We already note that the results do not depend on the size of the binary list.

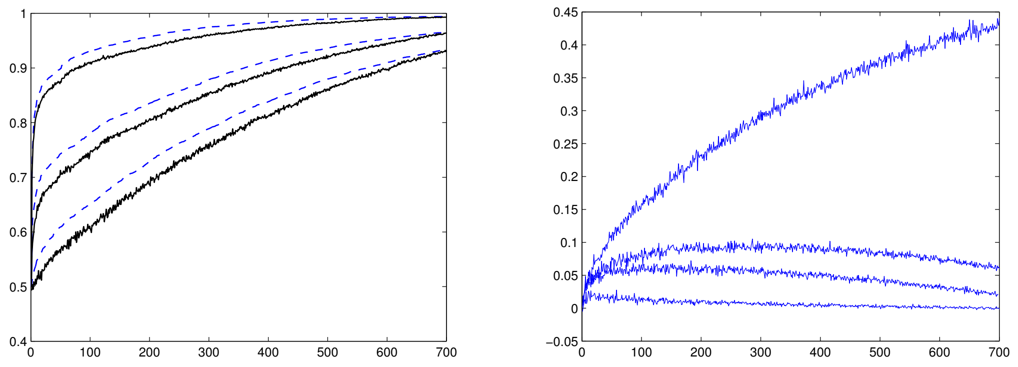


Figure 3: Results of the simulation study as a function of the number of leaves: (a) true AUC (dashed) and empirical AUC (solid) where curves more to the point (0, 1) correspond to higher values of u, and (b) difference between true AUC and empirical AUC where higher curves correspond to higher values of h.

Fig. 3(a) shows the average empirical AUC for a fixed $h = 0.2$ and, respectively, $u = 0.5, 0.6, 0.7$ as a function of the number of leaves. Clearly, splitting leaves (breaking ties or splitting segments) leads to higher AUC, and the better the tie breaking, the larger the gains in AUC. Corresponding standard deviations are very small, decreasing with the number of leaves, and for clarity have been omitted in the figure. The influence of the estimation errors is illustrated in Fig. 3(b). Here, we have fixed $u = 0.2$ and we show the average difference between true AUC and empirical AUC for, respectively, $h = 0.1, 0.2, 0.4, 1$ (with the latter implying completely random scores). Clearly, even for quite high values of h the difference between true and empirical AUC is small, and after an early point decreasing when the number of leaves increases (of course, except for completely random scores). In other words, leaf-splitting seems to be tolerant toward estimation errors of probabilities for the positive class in the sense that only the *ordering* of scores is important: As long as an estimation error does not cause an inversion of scores with respect to the true probabilities, there is no negative effect on the AUC.

5.1 Conclusions from the Theoretical Results

From our theoretical analysis above, we may conclude that the empirical AUC increases with the number of scores produced by a tree, at least when these scores are better than random approximations of the true conditional probabilities in the leaves. This result explains why unpruned or only slightly pruned trees in combination with Laplace correction in the leaves have been shown to be very good rankers. It also explains why other methods such as bagging lead to an improvement in AUC. The experiments with synthetic data support our conjecture that local tie breaking is beneficial for the AUC, and that it is robust toward estimation errors since we are only interested in a correct ordering of the scores.

6 Fuzzy Decision Trees

Our results, theoretical and empirical, lead us to conjecture that fuzzy decision trees should be good rankers. By softening the splits at feature threshold values in internal nodes, these trees produce a fuzzy partitioning of the instance space and, thereby, much more scores than conventional decision trees. Thus, while being at least competitive to conventional trees in terms of classification accuracy, they additionally offer a better representation of the confidence of a classification. Roughly speaking, boundary cases close to the split points are considered less certain than examples far away from these boundaries. And even if the scores produced by fuzzy decision trees are not necessarily good probability estimates, they still yield a reasonable ordering in terms of confidence, which is completely sufficient for good ranking performance.

Fuzzy variants of decision tree induction have been developed for quite a while (e.g [29, 14, 17, 18]) and seem to remain a topic of interest even today [20, 21, 22]. We recommend [19] as a one of the most sophisticated approaches, including a comprehensive overview of research in this field. All approaches provide a typical example for the "fuzzification" of standard machine learning methods. More specifically, fuzzification here primarily concerns the threshold values used for defining splitting predicates (constraints) for numeric attributes. In conventional decision trees, predicates such as `size` ≤ 180 can be modeled in terms of an indicator function, which in this example in given by the mapping $x \mapsto \mathbb{I}_{(0,180]}(x)$. In fuzzy trees, thresholds are defined in terms of fuzzy sets and, hence, an indicator function is replaced by a monotone increasing or decreasing function $\mu(\cdot)$ that maps to $[0, 1]$. An instance with attribute value x will then follow the left branch with degree $1 - \mu(x)$ and the right branch with degree $\mu(x)$. To classify a new instance by a fuzzy decision tree, different inference mechanisms have been devised. A common approach is to associate with each leaf node, which is labeled by a class, a degree of support which is given by the t-norm combination of the degrees to which the instance follows the edges along that path. The overall support of a class is then given by the t-conorm combination of all support degrees for that class. If a classification is requested, i.e., a definite decision in favor of a single class, then the class with the highest overall support is an obvious choice. In our context of ranking, the score of an instance is given by the support of the positive class.

To verify our conjecture, we analyzed the ranking performance of fuzzy decision trees in an experimental way. Decision tree learners are quite complex from an implementation point of view, and technical details of an implementation may have a significant influence on the performance. To guarantee maximal comparability, we therefore resorted to a very simple approach for constructing fuzzy decision trees: First, we build a conventional decision tree as usual, and afterward we fuzzify the split points of this tree. Thus, the type of splitting is the only structural difference between the fuzzy and non-fuzzy trees used in the experiments. Fuzzifying the split points is done by replacing the corresponding step function on the domain of the split attribute by a sigmoid membership function, namely the cumulative distribution function of a Gaussian density. Moreover, fuzzy inference is done using the product t-norm and the algebraic sum as t-conorm.

Fig. 2(b) shows the average AUC values obtained by the fuzzy decision tree. Table 4 summarizes the results for all data sets that contain numerical attributes by reporting the average improvement in AUC value. The number in the fourth column is in bold when the fuzzy tree is smaller and has higher AUC value than the best decision tree obtained in our first set of experiments (as described in Section 4). The experimental setup is kept identical except, of course, for the fuzzification of the trees. We estimated the bandwidths of the Gaussians as follows. First, we standardized the training data to have zero mean and unit variance (and test instances are transformed accordingly). Then, we only tried a single bandwidth $\lambda \in \{0.0001, 0.00015, 0.001, 0.005, 0.01, 0.1, 0.15, 0.2\}$. The reported results correspond to the best λ as measured on a validation set (20% of the training set). We make three observations from these results.

1. First, decision trees without Laplace correction benefit most from the post-fuzzification. This was to be expected, since these trees by themselves are not able to produce many distinct scores. Nonetheless, in general, it is still best to use Laplace correction in the leaves in order to distinguish between small and large leaves. More specifically, the null hypothesis that Laplace correction does not change the ranking performance of our fuzzy trees is rejected by the Wilcoxon signed-ranks at the 5% significance level.

2. Second, fuzzy decision trees have significantly larger AUC values along all pruning levels when compared to the standard trees, independent of whether Laplace correction is used. This result holds at the 1% significance level and hence is strong evidence in favor of our conjecture that fuzzy trees are very good rankers.

3. Third, to improve on the AUC value of the best tree in our first set of experiments, we can almost always use a fuzzy decision tree that is much smaller. For example, a fuzzy tree #7 is a better ranker than a standard unpruned

tree (#10) at the 5% significance level. This is important for practical purposes since the unpruned or slightly pruned trees that were used so far are not comprehensible, interpretable, and clearly have a very large variance.

As a final remark, we note that the number of distinct scores produced by a fuzzy decision tree is many times higher when compared to conventional trees (even when they use Laplace correction; see Fig. 2(d) and Table 5). So again, there is indeed a clear correlation between increase in number of distinct scores and increase in average AUC value, as was to be expected from the formal analysis.

A more advanced tuning of the membership functions, such as allowing for different shapes and parameters in each node, is very likely to improve further on the results obtained so far. We expect the same for the more advanced fuzzy decision tree learners, since their scores are very accurate with respect to classification accuracy and ties among ranked instances are not likely to occur [28].

7 Conclusion

In this paper, we have analyzed, both formally and experimentally, important and sometimes surprising observations from the recent machine learning literature concerning methods to improve decision trees for the bipartite ranking problem. More specifically, we can draw two main conclusions from our analysis. First, Laplace correction significantly increases performance in terms of AUC. In contrast to previous conjectures, however, the reason is not a better probability estimation, but instead a reasonable tie breaking effect that comes along with an increased number of scores. This also explains the previous finding that generalizations of the Laplace correction, like the m-estimate, essentially yield the same results (no further improvement is obtained). Second, unpruned trees almost always have higher AUC values than standard pruned trees (at least with Laplace correction). The reason is that unpruning increases the number of segments of the ROC curve, and thereby

Table 4: Improvement in AUC and its standard deviation (values separated by a /) of fuzzy decision trees #1, #5, and #10. Results are shown in two parts: without and with Laplace correction. The fourth column of each part lists the decision trees that obtain the highest AUC. A number is in bold when there is a fuzzy tree that is smaller and has higher AUC than the best decision tree so far.

#	without Laplace correction				with Laplace correction			
	1	5	10	max	1	5	10	max
1	.0022 / -.0050	.0050 / -.0045	.0203 / -.0074	**7**	.0022 / -.0058	.0039 / -.0050	.0011 / -.0033	**9**
3	.0207 / -.0132	.0193 / -.0200	.0269 / -.0357	**10**	.0143 / -.0139	.0096 / -.0130	.0069 / -.0137	**10**
5	.0051 / -.0200	.0096 / -.0229	.0470 / -.0455	**9**	.0053 / -.0211	.0062 / -.0200	.0115 / -.0113	**9**
6	.0004 / -.0913	.0316 / -.0508	.0393 / -.0514	2	.0004 / -.0919	.0082 / -.0298	.0034 / -.0400	**4**
7	-.0017 / -.0658	.0231 / -.0628	.0325 / -.0754	**8**	-.0025 / -.0786	.0028 / -.0667	.0025 / -.0921	**10**
8	-.0000 / -.0675	-.0000 / -.0675	.0003 / -.0587	1-5	-.0000 / -.0675	-.0000 / -.0675	.0006 / -.0557	10
10	.0338 / -.0618	.0412 / -.0525	.0319 / -.0502	**10**	.0167 / -.0461	.0111 / -.0477	.0102 / -.0506	**10**
12	.0563 / -.0497	.0461 / -.0554	.0401 / -.0526	**10**	.0595 / -.0463	.0453 / -.0442	.0342 / -.0477	**6**
15	.0496 / -.0529	.0556 / -.0283	.0543 / -.0370	**4**	.0440 / -.0497	.0373 / -.0341	.0323 / -.0371	**10**
16	.0139 / -.0322	.0211 / -.0271	.0356 / -.0363	**10**	.0039 / -.0360	.0080 / -.0264	.0043 / -.0116	10
17	.1046 / -.0733	.0905 / -.0843	.0946 / -.0799	1	.0537 / -.0561	.0555 / -.0501	.0537 / -.0547	**10**
18	.0260 / -.0102	.0266 / -.0096	.0278 / -.0104	**2**	.0162 / -.0094	.0029 / -.0042	-.0013 / -.0051	**7**

Table 5: Relative increase in the number of distinct scores gained by the fuzzy decision trees #1, #5, and #10 (e.g, a value of 10 means that the fuzzy tree yields ten times as many distinct scores). Results are shown in two parts: without and with Laplace correction. The fourth column of each part lists the trees that obtain the highest number of distinct scores.

#	without Laplace correction				with Laplace correction			
	1	5	10	max	1	5	10	max
1	15.2 ± 4.5	8.6 ± 0.9	3.5 ± 1.1	10	14.0 ± 4.8	7.9 ± 0.9	2.7 ± 0.6	10
3	3.8 ± 0.3	3.6 ± 0.5	3.9 ± 0.6	10	3.7 ± 0.3	3.2 ± 0.4	2.5 ± 0.7	10
5	2.0 ± 0.7	2.2 ± 1.1	2.2 ± 0.8	10	1.9 ± 0.6	2.2 ± 1.1	1.4 ± 0.3	10
6	1.4 ± 0.3	1.5 ± 0.8	2.0 ± 0.6	10	1.4 ± 0.3	1.3 ± 0.9	1.3 ± 0.3	10
7	1.7 ± 0.6	1.8 ± 0.6	2.2 ± 0.4	10	1.5 ± 0.5	1.6 ± 0.4	1.3 ± 0.4	10
8	1.0 ± 0.2	1.0 ± 0.2	1.0 ± 0.2	10	1.0 ± 0.2	1.0 ± 0.2	0.9 ± 0.2	10
10	5.3 ± 1.8	6.0 ± 1.3	5.9 ± 1.3	8	3.4 ± 1.1	3.6 ± 0.5	3.4 ± 0.5	10
12	3.8 ± 0.2	2.6 ± 0.3	2.7 ± 0.3	6	3.8 ± 0.2	1.8 ± 0.3	1.7 ± 0.2	6
15	5.5 ± 1.2	5.5 ± 0.9	5.7 ± 0.5	10	5.0 ± 0.7	4.7 ± 0.6	4.5 ± 0.3	10
16	5.0 ± 1.0	5.1 ± 1.0	5.0 ± 0.9	10	3.8 ± 0.6	3.4 ± 0.7	3.0 ± 0.7	10
17	3.9 ± 0.6	4.0 ± 0.7	4.1 ± 0.8	2	2.2 ± 0.2	2.2 ± 0.2	2.1 ± 0.2	5
18	17.0 ± 1.5	13.9 ± 1.1	13.8 ± 1.5	10	10.8 ± 1.0	8.1 ± 0.7	7.1 ± 0.8	10

leads to higher AUC values provided that the empirical scores are sufficiently good estimations of the true slopes of the segments. In fact, the relationship between AUC value and pruning level is not necessarily monotone, as it was previously assumed to be, especially when Laplace correction is turned off.

One obvious conclusion from these results is that fuzzy decision trees should be good rankers, a conjecture that we could verify in an experimental way. In fact, our experiments show that fuzzy trees are able to consistently outperform the methods proposed in the machine learning literature, while not loosing the advantages of decision trees such as comprehensability and interpretability. Strictly speaking, our claims were of course only verified for a specific decision tree learner and a fuzzy variant thereof. Yet, C4.5 is a state-of-the-art method, and the fuzzy variant we used is a rather simple one that could still be improved in various ways. Therefore, we believe that a generalization of our conclusions beyond these specific algorithms is warranted.

Roughly speaking, the distinguishing feature of fuzzy decision trees is their ability to represent the confidence of a classification in a reasonable way. Ordering the instances according to their scores, i.e., their degrees of membership in the positive class, consequently leads to good rankings in terms of the AUC metric.

Interestingly, this advantage is not exploited in the conventional classification setting. To make a classification, a fuzzy prediction has to be defuzzified, which essentially means predicting the class with the higher degree of membership. Thus, the advantage of discriminating scores is obviously lost, and indeed, a defuzzified fuzzy decision tree again produces hard decision boundaries in the instance space. We believe this to be the main reason for why a consistent advantage of using fuzzy trees for classification has not yet been shown (at least not in a resilient way across several independent studies). As we have seen, the situation is quite different in the ranking setting, where fuzzy decision trees significantly outperform their non-fuzzy counterparts.

Another potential advantage of fuzzy decision trees is a better trade-off between classification and ranking performance. In fact, our results have shown that, in the non-fuzzy case, pruned trees are good classifiers but poor rankers, while unpruned trees are good rankers but poor classifiers. In other words, a single decision tree cannot be both at the same time, a good classifier and a good ranker. Fuzzy decision trees may overcome this dilemma, since even small fuzzy trees can produce diverse scores. In future work, we plan to elaborate on this aspect more closely.

References

[1] Arthur Asuncion and David Newman. UCI machine learning repository, 2007.

[2] Christopher Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, NY, USA, 2007.

[3] Andrew Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, 1997.

[4] Leo Breiman, Jerome Friedman, Charles Stone, and Richard Olshen. *Classification and Regression Trees*. Chapmand and Hall/CRC Press, Boca Raton, FL, USA, 1984.

[5] Stéfan Clémençon and Nicolas Vayatis. Approximation of the optimal ROC curve and a tree-based ranking algorithm. In Yoav Freund, László Györfi, György Turán, and Thomas Zeugmann, editors, *Proceedings of the 19th International Conference on Algorithmic Learning Theory (ALT 2008)*, pages 22–37, Budapest, Hungary, October 13-16 2008. Springer.

[6] William Cohen, Rober Shapire, and Yoram Singer. Learning to order things. In Michael Jordan, Micheal Kearns, and Sara Solla, editors, *Advances in Neu-*

ral Information Processing Systems (NIPS 1997), pages 451–457, Denver, CO, USA, December 2-4 1997. MIT Press.

[7] Janes Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7(Jan):1–30, 2006.

[8] Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[9] César Ferri, Peter Flach, and José Hernández-Orallo. Learning decision trees using the area under the ROC curve. In Claude Sammut and Achim Hoffmann, editors, *Proceedings of the 19th International Conference on Machine Learning (ICML 2002)*, pages 139–146, Sydney, Australia, July 8-12 2002. Morgan Kaufmann.

[10] César Ferri, Peter Flach, and José Hernández-Orallo. Improving the AUC of probabilistic estimation trees. In Nada Lavrac, Dragan Gamberger, Ljupco Todorovski, and Hendrik Blockeel, editors, *Proceedings of the 14th European Conference on Machine Learning (ECML 2003)*, pages 121–132, Cavtat-Dubrovnik, Croatia, September 22-26 2003. Springer.

[11] Peter Flach and Edson Matsubara. A simple lexicographic ranker and probability estimator. In Joost Kok, Ramon López de Mántaras, Stan Matwin, Dunja Mladenic, and Andrzej Skowron, editors, *Proceedings of the 18th European Conference on Machine Learning (ECML 2007)*, pages 575–582, Warsaw, Poland, September 17-21 2007. Springer.

[12] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 36(1):3–42, 2006.

[13] Eyke Hüllermeier and Stijn Vanderlooy. An empirical and formal analysis of decision trees for ranking. Technical Report Computer Science Series 56, Philipps Universität Marburg, Germany, 2008.

[14] Cezary Janikow. Fuzzy decision trees: Issues and methods. *IEEE Transactions on Systems, Man, and Cybernetics*, 28(1):1–14, 1998.

[15] Ron Kohavi. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In Evangelos Simoudis, Jiawei Han, and Usama Fayyad, editors, *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD 1996)*, pages 202–207, Portland, OR, USA, August 24 1996. AAAI Press.

[16] Charles Ling and Robert Yan. Decision tree with better ranking. In Tom Fawcett and Nina Mishra, editors, *Proceedings of the 20th International Conference on Machine Learning (ICML 2003)*, pages 480–487, Washington, DC, USA, August 21-24 2003. AAAI Press.

[17] Christophe Marsala and Bernadette Bouchon-Meunier. Fuzzy partitioning using mathematical morphology in a learning scheme. In *Proceedings of the 5th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 1996)*, pages 1512–1517, New Orleans, LA, USA, September 8-11 1996.

[18] Christophe Marsala and Bernadette Bouchon-Meunier. An adaptable system to construct fuzzy decision trees. In *Proceedings of the 18th North American Fuzzy Information Processing Society Conference (NAFIPS 1999)*, pages 223–297, New York, NY, USA, June 10-12 1999.

[19] Cristina Olaru and Louis Wehenkel. A complete fuzzy decision tree technique. *Fuzzy Sets and Systems*, 138(2):221–254, 2003.

[20] Witold Pedrycz and Zenon Sosnowski. Designing decision trees with the use of fuzzy granulation. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, 30(2):151–159, 2000.

[21] Witold Pedrycz and Zenon Sosnowski. The design of decision trees in the framework of granular data and their application to software quality models. *Fuzzy Sets and Systems*, 123(3):271–290, 2001.

[22] Witold Pedrycz and Zenon Sosnowski. C-fuzzy decision trees. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 35(4):498–511, 2005.

[23] Foster Provost and Pedro Domingos. Tree induction for probability-based ranking. *Machine Learning*, 52(3):199–215, 2003.

[24] Foster Provost and Venkateswarlu Kolluri. A survey of methods for scaling up inductive algorithms. *Data Mining and Knowledge Discovery*, 3(2):131–169, 1999.

[25] Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco, CA, USA, 1993.

[26] Padhraic Smyth, Alexander Gray, and Usama Fayyad. Retrofitting decision tree classifiers using kernel density estimation. In Armand Prieditis and Stuart Russell, editors, *Proceedings of the 12th International Conference on Machine Learning (ICML 1995)*, pages 506–514, Tahoe City, CA, USA, July 9-12 1995. Morgan Kaufmann.

[27] Bin Wang and Harry Zhang. Improving the ranking performance of decision trees. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *Proceedings of the 17th European Conference on Machine Learning (ECML 2006)*, pages 461–472, Berlin, Germany, September 18-22 2006. Springer.

[28] Xizhao Wang, Bin Chen, Guoliang Qian, and Feng Ye. On the optimization of fuzzy decision trees. *Fuzzy Sets and Systems*, 112(1):117–125, 2000.

[29] Rosina Weber. Fuzzy-ID3: a class of methods for automatic knowledge acquisition. In *Proceedings of the 2nd International Conference on Fuzzy Logic*

and Neural Networks (IIZUKA 1992), pages 265–268, Iizuka, Japan, July 17-22 1992.

[30] Ian Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques.* Morgan Kaufmann, San Francisco, CA, USA, 2nd edition, 2005.

[31] Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers. In Carla Brodley and Andrea Pohoreckyj Danyluk, editors, *Proceedings of the 18th International Conference on Machine Learning (ICML 2001)*, pages 609–616, San Francisco, CA, USA, June 28 - July 1 2001. Morgan Kaufmann.

[32] Harry Zhang and Jiang Su. Learning probabilistic decision trees for AUC. *Pattern Recognition Letters*, 27(8):892–899, 2006.