A Comparative Analysis of Methods for Pruning Decision Trees

Floriana Esposito, *Member, IEEE,*
Donato Malerba, *Member, IEEE, and*
Giovanni Semeraro, *Member, IEEE*

Abstract—In this paper, we address the problem of retrospectively pruning decision trees induced from data, according to a top-down approach. This problem has received considerable attention in the areas of pattern recognition and machine learning, and many distinct methods have been proposed in literature. We make a comparative study of six well-known pruning methods with the aim of understanding their theoretical foundations, their computational complexity, and the strengths and weaknesses of their formulation. Comments on the characteristics of each method are empirically supported. In particular, a wide experimentation performed on several data sets leads us to opposite conclusions on the predictive accuracy of simplified trees from some drawn in the literature. We attribute this divergence to differences in experimental designs. Finally, we prove and make use of a property of the reduced error pruning method to obtain an objective evaluation of the tendency to overprune/underprune observed in each method.

Index Terms—Decision trees, top-down induction of decision trees, simplification of decision trees, pruning and grafting operators, optimal pruning, comparative studies.

1 Introduction

Decision tree induction has been studied in detail both in the area of pattern recognition and in the area of machine learning. In the vast literature concerning decision trees, also known as *classification trees* or *hierarchical classifiers*, at least two seminal works must be mentioned, those by Breiman et al. [2] and Quinlan [24]. The former originated in the field of statistical pattern recognition and describes a system, named CART (Classification And Regression Trees), which has mainly been applied to medical diagnosis and mass spectra classification. The latter synthesizes the experience gained by people working in the area of machine learning and describes a computer program, called ID3, which has evolved into a new system, named C4.5 [26].

Various heuristic methods have been proposed for designing a decision tree [29], the best known being the *top-down* method. There are three main problems in top-down induction of decision trees (TDIDT), the first of which concerns how to classify new observations, given a decision tree. The most common approach associates each leaf with a single class and then assigns that class to all new observations which reach that leaf. Typically, the associated class is the one with the largest number of examples reaching the leaf (*majority class* criterion). A second problem is determining the test to associate with each node of the tree. It can be broken down into a definition of the kind of tests allowed and selection of the best one. Many TDIDT systems adopt the following *univariate relational* test scheme:

(attribute # value)

where # denotes one of the following relational operators: (=, ≤, >). The type of test depends on the attribute domain: equality for *non-ordered* attributes and greater-than/less-than for *ordered* attributes. Since there are many different instances of a test scheme, one for each possible attribute-value pair, it is necessary to define a selection measure in order to choose the best. Mingers [19] reviewed some selection measures based on statistics and information theory. Other proposals can be found in [10], [34].

A third problem, which Breiman et al. [2] deem the most important, concerns the determination of the leaves. There are two different ways to cope with this: either by prospectively deciding when to stop the growth of a tree (*pre-pruning*) or by retrospectively reducing the size of a fully expanded tree, T_{max}, by pruning some branches (*post-pruning*) [5]. *Pre-pruning* methods establish stopping rules for preventing the growth of those branches that do not seem to improve the predictive accuracy of the tree. Some rules are:

1) All observations reaching a node belong to the same class.
2) All observations reaching a node have the same feature vector (but do not necessarily belong to the same class).
3) The number of observations in a node is less than a certain threshold.
4) There is no rejection for chi-square tests on the independence between a feature X_j and the class attribute C [24].
5) The merit attributed to all possible tests which partition the set of observations in the node is too low.

Actually, if the decision process adopted for a tree is based on the *majority class* criterion, then the stopping Rules 1 and 2 are reasonable, and indeed they are universally accepted. In particular, the second rule deals with the case of contradictory examples (*clashes*), which can be ob-

- *The authors are with Dipartimento di Informatica, Università degli Studi di Bari, via Orabona 4, 70126 Bari, Italy.*
  E-mail: {esposito, malerba, semeraro}@lacam.uniba.it.

Manuscript received 15 Dec. 1994; revised 2 Jan. 1996. Recommended for acceptance by R. Duin.
For information on obtaining reprints of this article, please send e-mail to: transpami@computer.org, and reference IEEECS Log Number P97006.

0162-8828/97/$10.00 © 1997 IEEE

served when either the class probability density functions overlap for some points in the feature space or the collection of examples is erroneously measured. Rule 3 is based on the idea that small disjuncts can be eliminated since they are error-prone, but an immediate objection is that in this way we cannot deal with exceptions [13]. Analogous considerations can be made for the fourth rule, since nodes with few cases do not generally pass significance tests, especially when approximate tests, such as chi-square tests, are used. Objections to the fifth stopping rule are more subtle, since they depend on the test scheme and selection measures adopted.

Indeed, when the selection measure belongs to the families of impurity measures [2] or C-SEP [10], stopping Rule 5 may fire, although some tests could be useful combined with others. For instance, given the following training set:

$$\{<2, 2, +>, <3, 3, +>, <-2, -2, +>, <-3, -3, +>,$$
$$<-2, 2, ->, <-3, 3, ->, <2, -2, ->, <3, -3, ->\}$$

where each triplet $<X_1, X_2, C>$ represents an example (see Fig. 1a), then every univariate test that leads to non-trivial partitioning, such as:

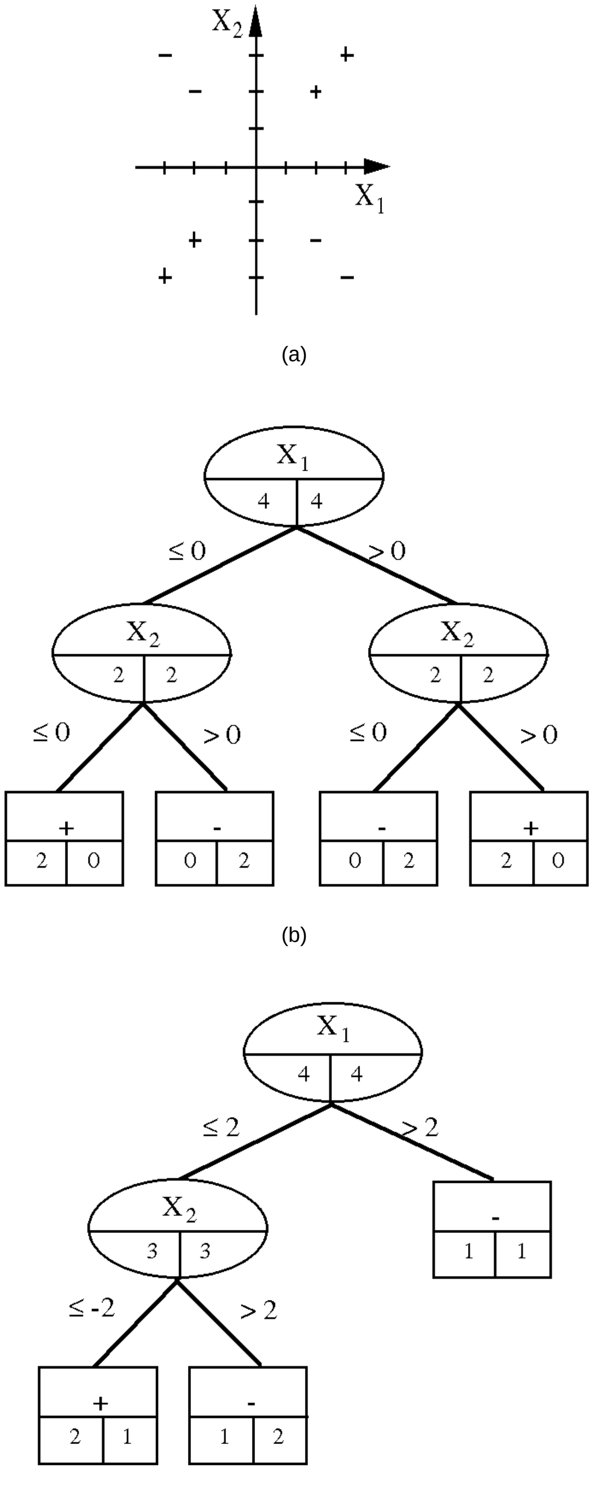
$$X_i \leq -3, X_i \leq -2, X_i \leq 2, i = 1, 2$$

maintains unchanged the distribution of training examples per class. Consequently, stopping Rule 5 will prevent expansion of the tree in Fig. 1b, which can correctly classify all the training instances.

One way around this short-sightedness is that of adopting a post-pruning method. In this case, a tree T_{max} is grown even when it seems worthless and is then retrospectively pruned of those branches that seem superfluous with respect to predictive accuracy [22]. For instance, for the same data set reported above, a learning system could generate the tree in Fig. 1c, in which case it is preferable to prune up to the root since no leaf captures regularities in the data. The final effect is that the intelligibility of the decision tree is improved, without really affecting its predictive accuracy.

In general, pruning methods aim to simplify those decision trees that overfitted the data. Their beneficial effects have attracted the attention of many researchers, who have proposed a number of methods. Unfortunately, their variety does not encourage the comprehension of both the common and the individual aspects. For instance, while some methods proceed from the root towards the leaves of T_{max} when they examine the branches to prune (*top-down approach*), other methods follow the opposite direction, starting the analysis from the leaves and finishing it with the root node (*bottom-up approach*). Furthermore, while some methods only use the *training set* to evaluate the accuracy of a decision tree, others exploit an additional *pruning set*, sometimes improperly called test set, which provides less biased estimates of the predictive accuracy of a pruned tree.

The main objective of this paper is to make a comparative study of some of the well known post-pruning methods (henceforth, denoted as pruning methods for the sake of brevity) with the aim of understanding their theoretical foundations, their computational complexity, and the strengths and weaknesses of their formulation. The next section is devoted to a critical review of six methods which have achieved wide-spread popularity, while Section 3 provides empirical support for some comments. In par-



(a)

(b)

(c)

Fig. 1. (a) Graphic representation of a training set. (b) Optimal decision tree for the training set. In each node, the class distribution, i.e., the number of training instances belonging to each class (+ or –), is reported. (c) A nonoptimal decision tree that can be subjected to pruning.

ticular, the experimental design and the characteristics of the databases considered are described. For each set of data, a preliminary empirical study is accomplished in order to gain additional information on the particular domain. The results of the significance tests reported in the paper concern both the performance of each single method and comparative analysis of pairs of methods on the various data sets. They are an extension of preliminary results presented by Malerba et al. [17]. Two main conclusions are made:

- setting aside some data for the pruning set is not generally a good strategy;
- some methods have a marked unsuspected tendency to overprune / underprune.

Finally, in Section 4, we discuss other methods not considered in the paper and some conclusions drawn in previous related works.

2 A Critical Review of Pruning Methods

Henceforth, $\mathfrak{I}_T$ will denote the set of *internal* (non-terminal) nodes of a decision tree T, $\mathfrak{L}_T$ will denote the set of *leaves* in T, and $\mathfrak{N}_T$ the set of nodes of T. Thus, $\mathfrak{N}_T = \mathfrak{I}_T \cup \mathfrak{L}_T$. The *branch* of T containing a node *t* and all its descendants will be indicated as T_t. The number of leaves of T will be denoted by the cardinality of $\mathfrak{L}_T$, $|\mathfrak{L}_T|$. The number of training examples of class *i* reaching a node *t* will be denoted by $n_i(t)$, the total number of examples in *t*, by *n(t)*, and the number of examples not belonging to the majority class, by *e(t)*.

In the following subsections, the descriptions of six pruning methods according to their original formulation are reported, together with some critical comments on the theoretical foundations, and the strengths and weaknesses of each method. A brief survey can be found in [17].

2.1 Reduced Error Pruning (REP)

2.1.1 *Description*

This method, proposed by Quinlan [25], is conceptually the simplest and uses the pruning set to evaluate the efficacy of a subtree of T_{max}. It starts with the complete tree T_{max} and, for each internal node *t* of T_{max}, it compares the number of classification errors made on the pruning set when the subtree T_t is kept, with the number of classification errors made when *t* is turned into a leaf and associated with the *best* class. Sometimes, the simplified tree has a better performance than the original one. In this case, it is advisable to prune T_t. This branch pruning operation is repeated on the simplified tree until further pruning increases the misclassification rate.

Quinlan restricts the pruning condition given above with another constraint: T_t can be pruned only if it contains no subtree that results in a lower error rate than T_t itself. This means that nodes to be pruned are examined according to a bottom-up traversal strategy.

THEOREM. *REP finds the smallest version of the most accurate subtree with respect to the pruning set.*[1]

PROOF. Let T^* denote the optimally pruned tree with respect to the pruning set, and t_0 its root node. Then, either T^* is the root tree $\{t_0\}$ associated with the most prevalent class, or, if $t_1, t_2, \ldots, t_s$, are the *s* sons of t_0, T^* is the tree rooted in t_0 with subtrees $T_{t_1}^*, \ldots, T_{t_s}^*$. The first part of the claim is obvious, while the second part is based on the additive property of the error rate for decision trees, according to which a local optimization on each branch T_{t_i} leads to a global optimization on T. These considerations immediately suggest a bottom-up algorithm that matches the REP procedure. □

As we will see later, this property can be effectively exploited in experimental comparisons to find the smallest optimally pruned tree with respect to the test set.

2.1.2 *Comments*

Another positive property of this method is its linear computational complexity, since each node is visited only once to evaluate the opportunity of pruning it. On the other hand, a problem with REP is its bias towards overpruning. This is due to the fact that all evidence encapsulated in the training set and exploited to build T_{max} is neglected during the pruning process. This problem is particularly noticeable when the pruning set is much smaller than the training set, but becomes less relevant as the percentage of cases in the pruning set increases.

2.2 Pessimistic Error Pruning (PEP)

2.2.1 *Description*

This pruning method, proposed by Quinlan [25], like the previous one, is characterized by the fact that the same training set is used for both growing and pruning a tree. Obviously, the *apparent* error rate, that is the error rate on the training set, is optimistically biased and cannot be used to choose the best pruned tree. For this reason, Quinlan introduces the continuity correction for the binomial distribution that might provide “a more realistic error rate.” More precisely, let

$$r(t) = e(t) / n(t)$$

be the apparent error rate in a single node *t* when the node is pruned, and

$$r(T_t) = \frac{\sum_{s \in \mathfrak{L}_{T_t}} e(s)}{\sum_{s \in \mathfrak{L}_{T_t}} n(s)}$$

be the apparent error rate for the whole subtree T_t. Then, the continuity correction for the binomial distribution gives:

$$r'(t) = [e(t) + 1/2] / n(t)$$

By extending the application of the continuity correction to the estimation of the error rate of T_t, we have:

[1] It can be proved that the optimality property is no longer guaranteed in Mingers' version of REP [20], in particular when among all the internal nodes, we prune the node *t* that shows the largest difference between the number of errors when the subtree T_t is kept and the number of errors when T is pruned in *t*.

$$r'(T_t) = \frac{\sum_{s\in\mathfrak{L}_{T_t}} \left[e(s)+1/2\right]}{\sum_{s\in\mathfrak{L}_{T_t}} n(s)} = \frac{\sum_{s\in\mathfrak{L}_{T_t}} e(s) + \frac{\left|\mathfrak{L}_{T_t}\right|}{2}}{\sum_{s\in\mathfrak{L}_{T_t}} n(s)}$$

For simplicity, henceforth we will refer to the number of errors rather than to the error rate, that is:

$$e'(t) = [e(t) + 1/2]$$

for a node *t*, and:

$$e'(T_t) = \sum_{s\in\mathfrak{L}_{T_t}} e(s) + \frac{\left|\mathfrak{L}_{T_t}\right|}{2}$$

for a subtree T_t.

It should be observed that, when a tree goes on developing until none of its leaves make errors on the training set, then *e(s) = 0* if *s* is a leaf. As a consequence, e'(T) only represents a measure of tree complexity that associates each leaf with a cost equal to 1/2. This is no longer true for partially pruned trees or when clashes (equal observations belonging to distinct classes) occur in the training set.

As expected, the subtree T_t makes less errors on the training set than the node *t* when *t* becomes a leaf, but sometimes it may happen that $n'(t) \le n'(T_t)$ due to the continuity correction, in which case the node *t* is pruned. Nonetheless, this rarely occurs, since the estimate $n'(T_t)$ of the number of misclassifications made by the subtree is still quite optimistic. For this reason, Quinlan weakens the condition, requiring that:

$$e'(t) \le e'(T_t) + SE(e'(T_t))$$

where

$$SE(e'(T_t)) = [e'(T_t) \cdot (n(t) - e'(T_t)) / n(t)]^{1/2}$$

is the standard error for the subtree T_t, computed as if the distribution of errors were binomial, even if the independence property of events does not hold any longer because T_{max} was built to fit the training data. The algorithm evaluates each node starting from the root of the tree and, if a branch T_t is pruned then the descendants of *t* are not examined. This top-down approach gives the pruning technique a high run speed.

2.2.2 Comments

The introduction of the continuity correction in the estimation of the error rate has no theoretical justification. In statistics, it is used to approximate a binomial distribution with a normal one, but it was never applied to correct over-optimistic estimates of error rates. Actually, the continuity correction is useful only to introduce a tree complexity factor. Nonetheless, such a factor is improperly compared to an error rate, and this may lead to either underpruning or overpruning. Indeed, if T_{max} correctly classifies all training examples, then:

$$e'(T_t) + SE(e'(T_t)) \approx \frac{1}{2}\left(\left|\mathfrak{L}_{T_t}\right| + \sqrt{\left|\mathfrak{L}_{T_t}\right|}\right)$$

and, since $e'(t) \approx e(t)$, then the method will prune if:

$$\left|\mathfrak{L}_{T_t}\right| + \sqrt{\left|\mathfrak{L}_{T_t}\right|} \ge 2e(t)$$

that is, pruning occurs if T_t has a sufficiently high number of leaves with respect to the number of errors it helps to recover. The constant 1/2 simply indicates the contribution of a leaf to the complexity of the tree. Obviously, such a constant is suitable in some problems but not others.

Lastly, we notice that even this method has a linear complexity in the number of internal nodes. Indeed, in the worst case, when the tree does not need pruning at all, each node will be visited once.

2.3 Minimum Error Pruning (MEP)

2.3.1 Description

Niblett and Bratko [23] proposed a *bottom-up* approach seeking for a single tree that minimizes "the expected error rate on an independent data set." This does not mean that a pruning set is used, but simply that the authors intend to estimate the error rate for unseen cases. Indeed, both the original version and the improved one reported in [6] exploit only information in the training set. However, implementation of the improved version requires an independent pruning set for the reasons explained later.

For a *k*-class problem, the expected probability that an observation reaching the node *t* belongs to the *i*th class is the following:

$$p_i(t) = \frac{n_i(t) + p_{ai} \cdot m}{n(t) + m}$$

where p_{ai} is the a priori probability of the *i*th class, and *m* is a parameter that determines the impact of the a priori probability on the estimation of the a posteriori probability $p_i(t)$.

For simplicity, *m* is assumed to be equal for all the classes. Cestnik and Bratko name $p_i(t)$ as *m-probability estimate*. When a new observation reaching *t* is classified, the expected error rate is given by:

$$\begin{aligned} EER(t) &= \min_i \left\{ 1 - p_i(t) \right\} \\ &= \min_i \left\{ \left[n(t) - n_i(t) + \left(1 - p_{ai}\right)\cdot m\right] / \left[n(t) + m\right] \right\} \end{aligned}$$

This formula is a generalization of the expected error rate computed by Niblett and Bratko [23]. Indeed, when m = k and $p_{ai} = 1/k$, i = 1, 2, …, k, i.e., the a priori probability distribution is uniform and equal for all classes, we get:

$$\begin{aligned} EER(t) &= \min_i \left\{ \left[n(t) - n_i(t) + k - 1\right] / \left[n(t) + k\right] \right\} \\ &= \left[\ e(t) + k - 1\ \right] / \left[n(t) + k\right] \end{aligned}$$

In the minimum error pruning method, the expected error rate for each internal node $t \in \mathfrak{T}_T$ is computed. This is called *static error*, STE(t). Then, the expected error rate of T_t, called *dynamic* (or *backed-up*) *error*, DYE(t), is computed as a weighted sum of the expected error rates of *t*'s children, where each weight p_s is the probability that an observation in *t* will reach the corresponding child *s*. In the original method proposed by Niblett and Bratko, the weights p_s were estimated by the proportion of training examples reaching the *s*th child. Later, Cestnik and Bratko [6] suggested an *m*-probability estimate with m = 2 for p_s, although they admitted having chosen *m* arbitrarily. In the following, we will consider the original proposal, which corresponds to a 0-probability estimate for p_s.

2.3.2 Comments

Generally, the higher the *m*, the more severe the pruning. In fact, when *m* is infinity, $p_i(t) = p_{ai}$, and since p_{ai} is estimated as the percentage of examples of the *i*th class in the training set, the tree reduced to a single leaf has the lowest expected error rate. However, for $m' > m$ the algorithm may not return a smaller tree than that obtained for a value *m*. This non-monotonicity property has a severe consequence on computational complexity: for increasing values of *m*, the pruning process must always start from T_{max}.

Obviously, the choice of *m* is critical. Cestnik and Bratko suggest the intervention of a domain expert who can choose the right value for *m* according to the level of noise in the data or even study the selection of trees produced. Since no expert was available, we decided to adopt a two-phase approach. First, we evaluate the classification accuracy of the pruned trees, produced for different *m* values, on an independent pruning set. Then, we select the smallest tree with the lowest empirical error rate.

Finally, we observe that the most recent version of minimum error pruning seems to have overcome two problems that affected the original proposal by Niblett and Bratko: optimistic bias [33] and dependence of the expected error rate on the number of classes [20].

2.4 Critical Value Pruning (CVP)

2.4.1 Description

This post-pruning method, proposed by Mingers [18], is very similar to a pre-pruning technique. Indeed, a threshold, named *critical value*, is set for the node selection measure. Then, an internal node of the tree is pruned if the value obtained by the selection measure for each test associated to edges coming out of that node does not exceed the critical value. Nevertheless, it may happen that the pruning condition is met by a node *t* but not by all its children. In this case, the branch T_t is kept because it contains relevant nodes. This further check is typical of a bottom-up method and represents the main difference from those pre-pruning methods that prevent a tree from growing even if subsequent tests might turn out to be meaningful.

The degree of pruning clearly changes with the critical value: the choice of a higher critical value results in a more drastic pruning. The method proposed by Mingers consists of two main steps:

1) Prune T_{max} for increasing critical values.
2) Choose the best tree among the sequence of pruned trees, by measuring the significance of the tree as a whole and its predictive ability.

2.4.2 Comments

In our experiments, we applied the CVP method to trees grown by using the gain-ratio selection measure [24], but we observed an unsuspected problem. In some cases, the gain ratio of a test equals the maximum value 1.0, so that we must prune the whole tree if we want to remove that test. A typical example is a binary test that separates all examples of one class from examples of the other classes. Since such tests are more likely to appear in the deepest levels of the tree, the result is that the series of pruned trees is actually reduced to only two trees, T_{max} and the root tree, with T_{max} generally being the most accurate on the test set. The final effect is that CVP does not prune at all.

As to the choice of the best tree in the sequence, one of the alternatives suggested by Mingers consists of estimating the error rate on an independent pruning set [20]. Nevertheless, the sequence detected in the first step of this method might not contain the best tree on the pruning set. This is a drawback with respect to the REP method, which is guaranteed to find the smallest optimally pruned subtree.

2.5 Cost-Complexity Pruning (CCP)

2.5.1 Description

This method is also known as the CART pruning algorithm [2]. It consists of two steps:

1) Selection of a parametric family of subtrees of T_{max}, $\{T_0, T_1, T_2, \ldots, T_L\}$, according to some heuristics.
2) Choice of the best tree T_i according to an estimate of the true error rates of the trees in the parametric family.

As regards the first step, the basic idea is that T_{i+1} is obtained from T_i by pruning those branches that show the lowest increase in apparent error rate per pruned leaf. Indeed, when a tree T is pruned in a node *t*, its apparent error rate increases by the amount $r(t) - r(T_t)$, while its number of leaves decreases by $\left|\mathcal{L}_{T_t}\right| - 1$ units. Thus, the following ratio

$$\alpha = \left(r(t) - r\left(T_t\right)\right) / \left(\left|\mathcal{L}_{T_t}\right| - 1\right)$$

measures the increase in apparent error rate per pruned leaf. Then, T_{i+1} in the parametric family is obtained by pruning all nodes in T_i with the lowest value of α. The first tree T_0 is obtained by pruning T_{max} of those branches whose α value is 0, while the last tree T_L is the root tree. It is possible to prove that each tree T_i is characterized by a distinct value α_i, such that $\alpha_i < \alpha_{i+1}$. Therefore, the set $\{T_0, T_1, T_2, \ldots, T_L\}$ is actually a *parametric family* of trees that we will denote as $T_{max}(\alpha)$. The parametric family can be built in a time that is quadratic in the number of internal nodes.

In the second phase, the best tree in $T_{max}(\alpha)$ with respect to predictive accuracy is chosen. The authors propose two distinct ways of estimating the true error rate of each tree in the family, one based on cross-validation sets, and the other on an independent pruning set.

2.5.2 Comments

In this former proposal, the training set $\mathfrak{E}$ used to build T_{max} is partitioned into v subsets $\mathfrak{E}^1, \mathfrak{E}^2, \ldots, \mathfrak{E}^v$ and then v auxiliary decision trees $T^1, T^2, \ldots, T^v$ are induced from the training sets $\mathfrak{E} - \mathfrak{E}^1, \mathfrak{E} - \mathfrak{E}^2, \ldots, \mathfrak{E} - \mathfrak{E}^v$, respectively. Following the same approach as before, it is possible to define v distinct parametric families $T^1(\alpha), T^2(\alpha), \ldots, T^v(\alpha)$, which can help to define the accuracy of the decision trees in $T_{max}(\alpha)$. More precisely, the error rate of $T_{max}(\alpha_i)$ is estimated as the average of the error rates of the trees $T^1((\alpha_i\alpha_{i+1})^{1/2}), \ldots, T^v((\alpha_i\alpha_{i+1})^{1/2})$. The assumption under this estimate is that trees $T^1((\alpha_i\alpha_{i+1})^{1/2}), \ldots, T^v((\alpha_i\alpha_{i+1})^{1/2})$ have the same true error rate as $T_{max}(\alpha_i)$. Nevertheless, there is no theoretical reason to support this. In fact, while it is reasonable to assume that $T, T^1, \ldots, T^v$, have the same error rate under conditions of stability of the TDIDT algorithm with respect to smaller data sets, the extension of such an assumption to pruned subtrees $T^j((\alpha_i\alpha_{i+1})^{1/2})$ cannot be justified. This means that cross-validation may provide us with an error rate estimate whose amount of bias is unpredictable [16]. It should be noted that the problem is the validity of the assumption, and not the estimate of the error rate of $T^j((\alpha_i\alpha_{i+1})^{1/2})$, $j = 1, 2, \ldots, v$, which is unbiased when the error rate is computed by counting the number of misclassifications on the jth cross-validation set.

When an independent pruning set is used, the CCP method is at a disadvantage with respect to the REP method because it can only choose a tree in the set $\{T_0, T_1, T_2, \ldots, T_L\}$ instead of the set of all possible subtrees of T_{max}. Consequently, if the most accurate subtree with respect to the pruning set is not in $\{T_0, T_1, T_2, \ldots, T_L\}$, it cannot be selected [11].

Another aspect of the CART pruning strategy that deserves attention is the 1SE rule. Kittler and Devijver [14] have shown that the standard deviation of the empirical error count estimator e_C, used with independent sets, is given by

$$\sigma(e_c) = (e(1-e) / N)^{1/2}$$

where:

- e is the true expected error rate of the classifier (in this case T_{max}),
- N is the size of the independent set used for computing the error rate estimate, e_C.

In order to reduce the instability of the size of the most accurate subtree of T_{max} when different training sets are sampled, Breiman et al. propose choosing the smallest tree in the parametric family $T_{max}(\alpha) = \{T_0, T_1, T_2, \ldots, T_L\}$ such that its error rate is not greater than $\sigma(e_C)$ with respect to the lowest error observed for trees T_i. Obviously, since e is not known, the authors resort to an estimate which is e_C itself. Nevertheless, if such an approximation is dubious in the case of independent pruning set, it is even more difficult to justify its extension to cross-validation since the errors performed on the v cross-validation sets are by no means independent. As we will see later, the effect of such a rule of thumb, called 1SE, is a tendency to overprune.

2.6 Error-Based Pruning (EBP)

2.6.1 Description

This is the pruning method implemented in C4.5 [26], the learning system that we employed in our experiments for building the trees. It is considered an improvement on the PEP method, since it is based on a far more pessimistic estimate of the expected error rate. Both use information in the training set for building and simplifying trees.

Unlike PEP, EBP visits the nodes of T_{max} according to a bottom-up post-order traversal strategy instead of a top-down strategy. The true novelty is that EBP simplifies a decision tree T by *grafting* a branch T_t onto the place of the parent of t itself, in addition to pruning nodes (see Fig. 2).

Taking the set of examples covered by a leaf t as a statistical sample, it is possible to estimate a confidence interval

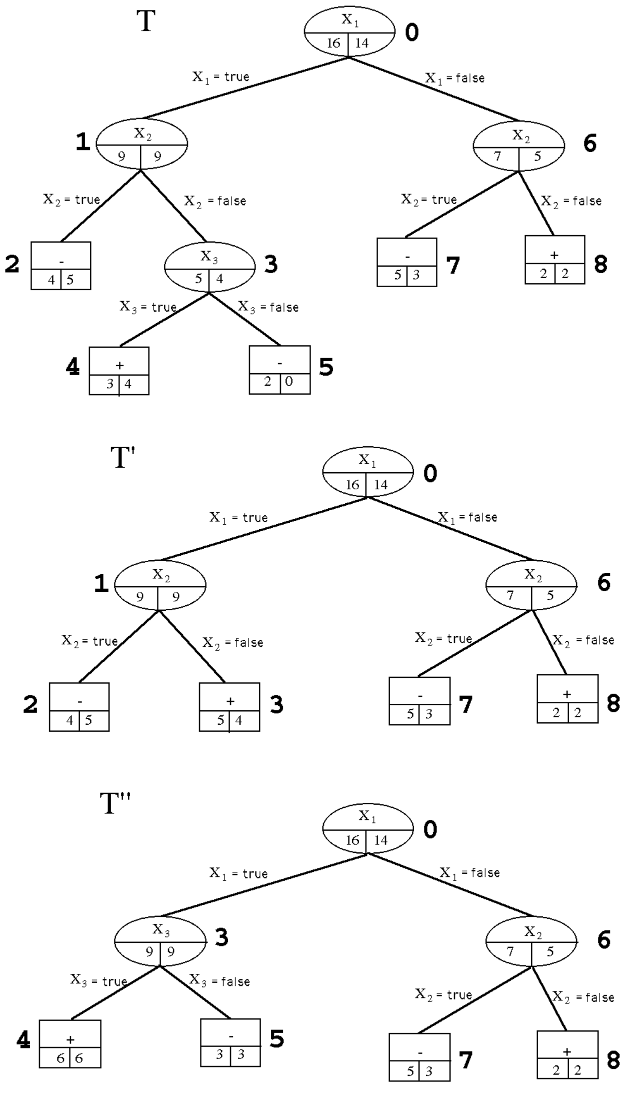


Fig. 2. The decision tree T′ is obtained by pruning T in node 1, while T″ is obtained by grafting the subtree rooted in node 3 onto the place of node 1.

TABLE 1
Main Characteristics of the Databases
Used for the Experimentation

database	No. Classes	No. Attributes	Real	Multi	Null Values	% Base Error	Noise Level	Uniform Distrib.
Iris	3	4	4	0	no	66.67	low	yes
Glass	7	9	9	0	no	64.49	low	no
Led	10	7	0	0	no	90	10%	yes
Hypo	4	29	7	1	yes	7.7	none	no
P.-gene	2	57	0	57	no	50	none	yes
Hepatitis	2	19	6	0	yes	20.65	none	no
Cleveland	2	14	5	5	yes	45.21	low	approx.
Hungary	2	14	5	5	yes	36.05	low	no
Switzerland	2	14	5	5	yes	6.5	low	no
Long Beach	2	14	5	5	yes	25.5	low	no
Heart	2	14	5	5	yes	44.67	low	approx.
Blocks	5	10	10	0	no	10.2	low	no
Pima	2	8	8	0	no	34.9	?	no
Australian	2	14	6	4	yes	44.5	?	approx.

$[L_{CF}(t), U_{CF}(t)]$ for the (posterior) probability of misclassification of *t*. The upper limit of the interval is of particular interest for a worst case analysis, and is defined as the real value, such that $P(e(t)/n(t) \leq U_{CF}) = CF$, where CF is the confidence level. Under the further assumption that errors in the training set are binomially distributed with probability *p* in n(t) trials, it is possible to compute the exact value of U_{CF} as the value of *p* for which a binomially distributed random variable X shows e(t) successes in n(t) trials with probability CF, that is $P(X \leq e(t)) = CF$.

In other words, if X has a binomial distribution with parameters $(U_{CF}, n(t))$, then the equality above must hold. Obviously, the value of U_{CF} depends on both e(t) and n(t). Having found the upper limit, the error estimates for leaves and subtrees are computed assuming that they are used to classify a set of unseen cases of the same size as the training set. Thus, the predicted error rate for *t* will be $n(t) \cdot U_{CF}$.

The sum of the predicted error rates of all the leaves in a branch T_t is considered to be an estimate of the error rate of the branch itself. Thus, by comparing the predicted error rate for *t* with that of the branch T_t and of the largest sub-branch $T_{t'}$ rooted in a child t' of *t*, we can decide whether it is convenient to prune T_t, to graft $T_{t'}$ onto the place of *t* or to keep T_t.

2.6.2 Comments

This method presents the advantage, with respect to the others, of allowing a subtree to be replaced by one of its branches. In this way, it is possible to remove "intermediate" tests which appear useless. Nevertheless, the algorithm implemented in C4.5 considers substituting a branch $T_{t'}$ onto the place of *t* even when $T_{t'}$ is reduced to a single leaf. The effect of this action is twofold: Pruning T in *t* and exchanging the class associated with *t* for that associated with t'. This latter effect, however, is undesirable since the class for *t* had already been chosen according to an optimality criterion (majority class). Hence, single node branches $T_{t'}$ are never grafted onto the place of T_t. The algorithm can be improved by simply checking that t' is an internal node, in which case the grafting operation should be considered.

Another point concerns two strong assumptions underlying this pruning method. It is hard to accept the training examples covered by a node *t* of T_{max} as a *statistical sample*, since T_{max} is not a generic tree randomly selected from a (potentially infinite) family of decision trees, but has been built in order to fit the data as well as possible. The assumption that errors in the sample have a *binomial distribution* is even more questionable.

Finally, the author maintains that this method employs a far more pessimistic estimate of errors than that adopted in pessimistic error pruning. As we will show in the next Section, our experimental results lead us to the very opposite conclusion. This can be explained by noting that U_{CF} is a pessimistic estimate of the error rate of both leaves and internal nodes. The effect of the pessimistic bias is therefore counter-balanced when we estimate the error rates in a node *t* and its branch T_t.

3 Empirical Comparison

3.1 The Design of the Experiment

In this section, we present the results of an empirical comparison of the methods presented above. The main characteristics of the data sets considered in our experiments are reported in Table 1. All databases are available in the UCI Machine Learning Repository[2] [21], and some of them have even been used to compare different pruning methods [25], [20], [3]. The database Heart is actually the union of four data sets on heart diseases, with the same number of attributes but collected in four distinct places (Hungary, Switzerland, Cleveland, and Long Beach).[3] Of the 76 original attributes, only 14 have been selected, since they are the only ones deemed useful. Moreover, examples have been assigned to two distinct classes: *no presence* (value 0 of the target attribute) and *presence* of heart diseases (values 1, 2, 3, 4).

[2] Data can be obtained electronically from
http://www.ics.uci.edu/~mlearn/MLRepository.html.
Furthermore, for each data file, relevant information on the characteristics of data is provided.

[3] The principal investigators responsible for these four databases are:
a. Hungarian Institute of Cardiology. Budapest: Andras Janosi, M.D.
b. University Hospital, Zurich, Switzerland: William Steinbrunn, M.D.
c. University Hospital, Basel, Switzerland: Matthias Pfisterer, M.D.
d. V.A. Medical Center, Long Beach and Cleveland Clinical Foundation: Robert Detrano, MD, PhD.

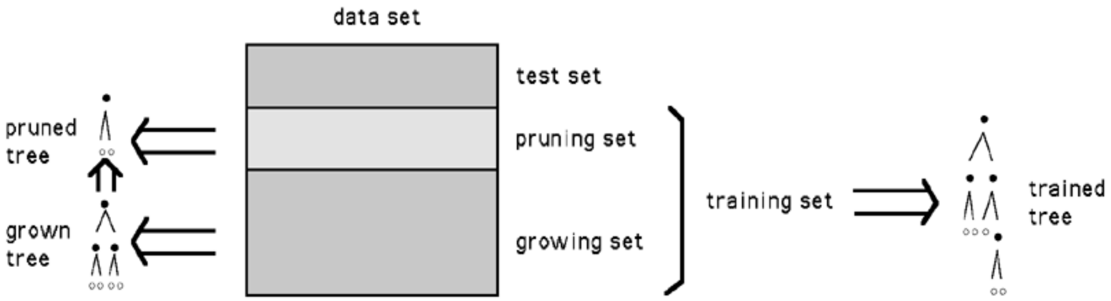


Fig. 3. The original data set is split into three subsets: The *growing* set (49 percent), the *pruning* set (21 percent), and the *test* set (30 percent). The union of the growing and pruning set is called the *training* set, and amounts to 70 percent of the size of the whole data set. The growing set contains 70 percent of the cases of the training set, and the pruning set contains the remaining 30 percent. Trees learned from the growing/training set are called *grown/trained* trees, respectively. Pruning trees can be obtained by pruning either grown trees or trained trees. In the former case, a pruning set is used.

TABLE 2
DISTRIBUTION OF CASES IN THE GROWING, PRUNING, TRAINING, AND TEST SET FOR EACH DATABASE

set	Iris	Glass	Led 1000	Led 200	Hypo	P.-gene	Hepatit.	Clevelnd	Hungary	Switzer	Long Beach	Heart	Blocks	Pima	Statlog Austral.
data	150	214	1000	-	3772	106	155	303	294	123	200	920	5473	768	690
train	105	150	700	200	2640	74	108	212	206	86	140	644	3831	538	483
grow	74	105	490	140	1848	52	76	148	144	60	98	451	2682	377	338
prune	31	45	210	60	792	22	32	64	62	26	42	193	1149	161	145
test	45	64	300	5000	1132	32	47	91	88	37	60	276	1642	230	207

In Table 1, the columns headed “real” and “multi” concern the number of attributes that are treated as real-value and multi-value discrete attributes, respectively. All other attributes are binary. In the column “null values,” we simply report the presence of null values in at least one attribute of any observation, since the system C4.5, used in our experiments, can manage null values [26]. In some cases, like in the Australian database, the missing values of categorical attributes had already been replaced by the mode of the attribute, while the missing values of continuous attributes had been replaced by the mean value. The column on base error refers to the percentage error obtained if the most frequent class is always predicted. We expect good decision trees to show a lower error rate than the base error. The last column states whether the distribution of examples per class is uniform or not.

Each data set has been randomly split into three subsets (see Fig. 3): *growing* set (49 percent), *pruning* set (21 percent), and *test* set (30 percent). The union of the growing and pruning set is called *training* set. The growing set and the training set are used to learn two decision trees, which are called *grown* tree and *trained* tree, respectively. The former is used by those methods that require an independent set in order to prune a decision tree, namely REP, MEP, CVP, as well as the *cost-complexity pruning*, based on an independent pruning set which adopts two distinct selection rules (0SE and 1SE). Conversely, the trained tree is used by those methods that exploit the training set alone, such as PEP, EBP, as well as the cost-complexity pruning, based on 10 cross-validation sets, that adopt either the 0SE rule (CV-0SE) or the 1SE rule (CV-1SE). The evaluation of the error rate is always made on the independent test set, using the empirical error count [14], which is an unbiased estimator.

The distribution of cases in the growing, pruning, training, and test sets for each database is reported in Table 2. In the case of Led-1000, we automatically generated a sample of 1,000 examples, and we performed a holdout resampling of training and test sets 25 times, as explained below. On the contrary, in the case of Led-200, we randomly generated all training sets of 200 samples and we tested the resultant trees on an independent test set of 5,000 instances. Since this is the procedure followed by Breiman et al. [2], our results can be compared with theirs.

With the exception of Led-200, each database has been randomly split into growing, pruning, and test set 25 times. For each run, two statistics are recorded: the number of leaves (*size*) of the resultant tree, and the error rate (*e.r.*) of the tree on the test set. This applies to pruned, grown, and trained trees.

Our experimental design, mostly based on holdout resampling, has been used in many other empirical studies, such as those performed by Mingers [20], Buntine and Niblett [4], and Holte [12]. The prediction errors are averaged over all trials in order to compute the mean prediction error and its corresponding variance or standard error. Mean prediction error would be an unbiased estimation if the prediction errors observed on the successive test sets were independent. However, this is not true since the test sets may overlap because of random resampling. Consequently, when a test is used to evaluate the significance of difference in prediction error, the results should be carefully interpreted. In particular, a statistical significance should be read as *very probable to hold for some expectation*

TABLE 3
AVERAGE SIZE AND ERROR RATE OF THE (OPTIMALLY PRUNED) GROWN/TRAINED TREES FOR EACH DATABASE USED IN THE EXPERIMENTS

	GROWN		**OPGT**		**TRAINED**		**OPTT**		**sign. test**
	size	**e.r.**	**size**	**e.r.**	**size**	**e.r.**	**size**	**e.r.**	
Iris	5.4 ± .265	5.866 ± .512	3.64 ± .19	4.888 ± .513	6.84 ± .33	5.598 ± .516	4.0 ± .173	4.442 ± .667	0
Glass	27.92 ± .707	36.87 ± 1.36	12.6 ± .849	31.62 ± 1.1	36.32 ± .457	35.38 ± 1.29	15.08 ± .954	28.31 ± 1.16	0
Led-1000	41.32 ± .787	27.96 ± .283	21.44 ± .651	25.64 ± .316	44.68 ± .72	27.48 ± .399	22.04 ± .552	25.31 ± .402	>
Led-200	24.4 ± .658	32.60 ± .417	17.92 ± .597	30.93 ± .262	29.76 ± .623	31.26 ± .369	20.16 ± .663	29.84 ± .295	>
Hypo	20.24 ± .708	.622 ± .052	9.16 ± .457	.399 ± .036	27.24 ± .91	.604 ± .041	9.36 ± .483	.352 ± .033	0
P-gene	19.0 ± .671	27.12 ± 1.7	7.84 ± .637	18.50 ± 1.22	25.6 ± .812	23.5 ± 1.17	10.0 ± .52	16.5 ± 1.044	>
Hepatitis	11.24 ± .441	22.81 ± 1.34	3.4 ± .539	15.83 ± .758	16.76 ± .53	21.87 ± 1.24	4.36 ± .622	16.34 ± .839	0
Cleveland	33.76 ± .981	30.07 ± .97	12.2 ± 1.03	22.99 ± .761	50.0 ± .802	29.1 ± .909	16.36 ± 1.03	20.84 ± .655	0
Hungary	32.56 ± .858	25.5 ± .866	10.16 ± .967	16.91 ± .708	47.88 ± 1.35	25.5 ± .812	9.64 ± 1.063	17.27 ± .745	0
Switzerland	9.64 ± .583	13.84 ± 1.04	1.32 ± .222	4.622 ± .731	11.76 ± .384	13.3 ± 1.40	1.16 ± .16	5.731 ± .721	0
Long Beach	31.64 ± .922	33.27 ± 1.29	4.12 ± .667	23.6 ± 1.04	43.48 ± 1.49	33.73 ± .894	4.92 ± .922	23.13 ± .899	0
Heart	115.7 ± 2.94	24.72 ± .523	36.16 ± 2.76	18.77 ± .387	164.6 ± 4.12	23.82 ± .636	45.96 ± 2.91	17.71 ± .351	0
Blocks	82.92 ± 1.41	3.65 ± .098	25.84 ± 1.03	2.609 ± .077	111.9 ± 1.83	3.57 ± .085	30.44 ± 1.1	2.354 ± .08	0
Pima	73.44 ± 1.29	30.24 ± .423	20.4 ± 1.53	21.95 ± .448	105.7 ± 1.03	31.43 ± .454	22.68 ± 1.63	21.71 ± .486	<
Australian	65.96 ± 2.46	18.74 ± .369	15.64 ± 2.06	13.08 ± .369	93.92 ± 2.21	17.60 ± .6	24.76 ± 1.86	12.04 ± .308	>

over the given data, and not as *very probable to hold for future data*. This observation, which should be regarded as true for this section, can actually be extended to any resampling method, including nonparametric bootstrap methods [8], "exact" permutation analysis [7] and repeated cross-validation [15]. Cross-validation, which does not suffer from this problem, has been used in a further study to confirm most of the conclusions we have reported below.

To study the effect of pruning on the predictive accuracy of decision trees, we compare the error rates of the pruned trees with those of the corresponding trained trees. In order to verify whether tree simplification techniques are beneficial or not, we compare two induction strategies: A *sophisticated* strategy that, in one way or another, prunes the tree T_{max}, and a *naive* strategy that simply returns T_{max}.

Unlike Mingers' previous empirical comparison of pruning methods [20], we will not rely on the Analysis of Variance (ANOVA) to detect statistically significant differences between pruning methods, since the ANOVA test is based on the assumption that the standard deviation is constant for all the experiments, whereas this is not so in our case since we compare the algorithms on different data sets, each of which has its own standard deviation. As proposed by Buntine and Niblett [4], a two-tailed paired t-test for each experiment is preferable.

Another interesting characteristic of pruning methods is their tendency to overprune decision trees. To study this problem, we produced two decision trees for each trial, called *optimally pruned grown tree* (OPGT) and *optimally pruned trained tree* (OPTT), respectively. The former is a grown tree that has been simplified by applying the REP method to the test set and is therefore, as proven in Section 2.1, the best pruned tree we could produce from the grown tree. Similarly, the OPTT is the best tree we could obtain by pruning some branches of the trained tree. OPGTs and OPTTs define an upper bound on the improvement in accuracy that pruning techniques can produce, as well as a lower bound on the complexity of pruned trees. Obviously, such an estimate is rather optimistic: The error rates of these optimal trees can be even lower than the corresponding Bayes optimal errors. However, optimally pruned trees are useful tools for investigating some properties of the data sets. For instance, by comparing the accuracy of the grown/trained trees with the accuracy of the corresponding OPGTs/OPTTs, it is possible to evaluate the maximum improvement produced by an ideal pruning algorithm. The magnitude of differences in accuracy of OPGTs and OPTTs can help to understand if the ideal goal of those simplification methods that require a pruning set is similar to the ideal goal for the other methods. On the contrary, a comparison of the accuracy of the corresponding grown and pruned trees provides us with an indication of the initial advantage that some methods may have over others. Moreover, the size of optimally pruned trees can be exploited to point out a bias of the simplification methods towards either overpruning or underpruning. In this case, we should compare the size of an OPGT with that of the corresponding tree produced by those methods that do use an independent pruning set, while the size of an OPTT should be related to the result of the other methods. As a matter of fact, optimally pruned trees were already exploited by Holte [12], but in that case only decision trees with a depth limited to one were considered.

Some experimental results, which are independent of the particular pruning method, are shown in Table 3. They are given in the form "5.4 ± 0.265," where the first figure is the average value for the 25 trials, while the second figure refers to the corresponding standard error.

As expected, the average size of the grown (trained) trees is always higher than that of the OPGT (OPTT). The ratio *(grown tree size/OPGT size)* ranges from 1.5 for the Iris data through 7.3 for the Switzerland data, up to 7.7 for the

TABLE 4
RESULTS OF THE TESTS ON ERROR RATES

database	REP	MEP	CVP	OSE	1SE	PEP	CV 0SE	CV 1SE	EBP	Total +/-
Iris	0	0	0	0	0	0	0	-	0	0/1
Glass	-	0	0	0	-	0	0	-	0	0/3
Led-1000	-	-	-	0	-	0	0	-	0	0/5
Led-200	-	-	-	-	-	0	0	+	0	1/5
Hypo	+	+	0	+	0	+	+	0	+	6/0
P. gene	0	0	0	0	0	0	0	0	0	0/0
Hepatitis	0	0	0	0	0	0	0	0	0	0/0
Cleveland	0	0	0	0	0	0	0	0	0	0/0
Hungary	+	+	0	+	+	+	+	+	+	8/0
Switzerland	+	0	+	+	+	+	+	+	+	8/0
Long Beach	+	+	+	+	+	+	+	+	+	9/0
Heart	0	0	0	0	0	+	0	0	+	2/1
Blocks	+	+	0	+	+	+	+	0	+	7/0
Pima	+	+	+	+	+	+	+	+	+	9/0
Australian	+	+	0	+	+	+	+	+	+	8/0
Total +/-	7/3	6/2	3/2	7/1	6/3	8/0	7/0	6/4	8/0	

Significance Level: 0.10.

database Long Beach, while the ratio *(trained tree size / OPTT size)* is even greater than 10 for the Switzerland data. Such great differences in size between some grown/pruned trees and their corresponding optimally pruned trees can be explained by looking at the “base error” column in Table 1. Indeed, for the “incriminated” data set, there are only two classes, one of which contains only 6.5 percent of cases. Since the learning system fails to find an adequate hypothesis for those cases, the pruning method will tend to prune the tree up to the root. Actually, there are three databases, namely Hepatitis, Switzerland, and Long Beach, in which the trained trees have an even greater error rate than the base error. They are typical examples of overfitting, for which pruning techniques should generally be beneficial. It is also worthwhile observing that the average size/error reported in the column “trained” of Table 3 for the Led-200 data are concordant with that observed by Schaffer [30, Table 1] under the same conditions, although in that case the trees were built using the Gini index [2] instead of the gain-ratio.

By comparing the error rate of the grown and trained trees, we can conclude that trained trees are generally more accurate than their corresponding grown trees (the result of a t-test at significance level 0.1 is shown in the last column of Table 3). This means that methods requiring a pruning set labor under a disadvantage. Nonetheless, the misclassification rate of the OPTT is not always lower than the error rate of the OPGT. Hence, in some data sets, like Hepatitis, Hungary, and Switzerland above, grown trees can be better starting points for pruning processes than trained trees.

Finally, the standard error reported in Table 3 confirms the large difference in standard deviation among the various databases, and thus use of the ANOVA significance testing is inappropriate.

3.2 Experimental Results

In this section, the results of 3,375 different experiments of pruning methods on various data sets are summarized and discussed.

The first factor that we analyze in this section is the error rate of the pruned trees. As stated above, we aim to discover whether and when a *sophisticated* strategy, which post-prunes a decision tree induced from data, is better than a *naive* strategy, which does not prune at all. Both strategies can access the same data, namely the training set, but the sophisticated strategy can either use some data for growing the tree and the rest for pruning it, or exploit all the data at once for building and pruning the decision tree. For this reason, we tested the significance of differences in error rate between the pruned decision trees and the trained trees.

Table 4 reports the outcomes of the tests for a confidence level equal to 0.10. A “+” in the table means that, on average, the application of the pruning method actually improves the predictive accuracy of the decision tree, while a “–“ indicates a significant decrease in predictive accuracy. When the effect of pruning is neither good nor bad, a 0 is reported.

At a glance, we can say that *pruning does not generally decrease the predictive accuracy*. More precisely, it is possible to partition the databases into three main categories: Those *prone to pruning*, *insensible to pruning*, and *refractory to pruning*.

The most representative of this latter category is cer-

TABLE 5
AVERAGE ERROR RATES OF TREES OBTAINED WITH DIFFERENT PRUNING METHODS

database	REP	MEP	CVP	0SE	1SE	PEP	CV-0SE	CV-1SE	EBP	trained
Iris	5.689 ± .617	6.222 ± .616	5.866 ± .512	5.778 ± .602	7.645 ± 1.54	5.332 ± .574	5.778 ± .574	***11.667 ± 2.62***	**5.065 ± .626**	5.598 ± .516
Glass	38.5 ± 1.21	38.19 ± 1.37	36.87 ± 1.36	38.0 ± 1.03	40.69 ± 1.29	**35.31 ± 1.35**	37.07 ± 1.74	***41.81 ± 1.86***	35.88 ± 1.29	35.38 ± 1.29
Led-1000	28.15 ± .455	28.24 ± .423	27.99 ± .357	28.16 ± .388	***29.52 ± .487***	27.91 ± .379	27.76 ± .349	29.09 ± .521	**27.36 ± .381**	27.48 ± .399
Led-200	33.64 ± .645	32.33 ± .369	32.60 ± .417	32.44 ± .518	***33.93 ± .61***	31.51 ± .299	31.47 ± .418	**29.84 ± .295**	31.33 ± .418	31.26 ± .369
Hypo	.515 ± .047	.51 ± .043	.541 ± .046	.508 ± .036	***.576 ± .041***	.505 ± .043	.512 ± .036	.576 ± .052	**.447 ± .039**	.604 ± .041
P. gene	23.0 ± 1.18	24.37 ± 1.33	***25.62 ± 2.25***	24.0 ± 1.21	25.62 ± 2.38	23.38 ± 1.25	23.88 ± 1.81	23.88 ± 1.03	**21.75 ± 1.40**	23.5 ± 1.17
Hepatitis	20.34 ± .903	21.28 ± 1.38	20.6 ± 1.03	**20.17 ± .936**	20.42 ± 1.15	21.1 ± 1.12	21.28 ± 1.2	21.11 ± 1.4	***21.36 ± 1.32***	21.87 ± 1.24
Cleveland	**27.65 ± .887**	28.88 ± .938	***30.07 ± 1.08***	29.10 ± .884	29.89 ± .989	29.01 ± .849	29.58 ± .735	29.76 ± .856	28.88 ± .855	29.1 ± .909
Hungary	21.68 ± .798	22.14 ± .907	***27.41 ± 1.25***	21.73 ± .782	21.77 ± 1.03	21.73 ± .876	21.86 ± .971	**20.82 ± .839**	22.36 ± .753	25.5 ± .812
Switzerland	6.272 ± .791	***12.65 ± 1.08***	**2.393 ± .762**	6.164 ± .707	5.839 ± .727	6.163 ± .689	6.055 ± .72	5.839 ± .727	6.163 ± .876	13.3 ± 1.40
Long Beach	26.27 ± 1.024	28.53 ± .983	26.93 ± 1.125	27.47 ± .977	25.67 ± 1.101	27.47 ± .929	26.4 ± 1.21	**25.4 ± 1.04**	***29.33 ± .953***	33.73 ± .894
Heart	23.54 ± .55	23.67 ± .436	24.13 ± .502	23.81 ± .445	***24.29 ± .458***	22.94 ± .505	23.754 ± .459	24.188 ± .521	**22.72 ± .51**	23.82 ± .636
Blocks	3.22 ± .107	3.34 ± .096	3.602 ± .098	3.19 ± .11	3.35 ± .096	**2.98 ± .099**	3.172 ± .118	***3.745 ± .155***	3.05 ± .106	3.57 ± .085
Pima	**25.88 ± .438**	27.22 ± .379	***30.0 ± .433***	26.65 ± .473	26.86 ± .455	28.85 ± .483	27.04 ± .622	27.18± .624	28.84 ± .472	31.43 ± .454
Australian	15.13 ± .445	15.07 ± .385	***18.47 ± .393***	15.0 ± .444	14.88 ± .39	**14.59 ± .344**	14.78 ± .4	14.84 ± .416	15.42 ± .368	17.60 ± .6

tainly the Led domain, for which almost all pruning methods that use an independent pruning set produce significantly less accurate trees. As shown in Table 3, data are weak relative to the complexity of the underlying model, hence reducing the number of samples for decision tree building also affects the success of the pruning process. Methods that operate on the trained trees seem to be more appropriate for refractory domains, especially if they are conservative. We will show later that the application of the 1SE rule leads to overpruning, which explains the negative result of the method CV-1SE for the Led-1000 database. Actually, the positive result for the analogous problem Led-200 is quite surprising, even though it agrees with figures reported in the book by Breiman et al. As proven by Schaffer [30], this result is actually an exception to the general behavior shown by cost complexity pruning in the digit recognition problem.

The sensibility to the choice of the simplification method seems to affect artificial domains rather than real data. Indeed, in our study, almost all databases are either prone or insensible to any kind of pruning. For instance, the Iris, Glass, Promoter-gene, Hepatitis, and Cleveland data, can be simplified by means of almost all methods considered, without significantly affecting the predictive accuracy of the tree. The most relevant exception is represented by the application of the 1SE rule with both an independent pruning set and cross-validation sets. Such heuristics increase the variance of the estimated error rate as shown in Table 5, which reports the average error rates together with the standard errors for each database and method. The lowest error rate is reported in bold type while the highest values are in bold italics.

It is worthwhile noting, that almost all databases not prone to pruning have the highest base error (from 45.21 percent for Cleveland to 90 percent for Led), while those databases with a relatively low base error, such as Hungary, Blocks, Pima, and Hypothyroid, benefit from any pruning strategy. There seems to be an indication that the simplification techniques may only improve the understandability of the trees, but cannot increase the predictive accuracy if no class dominates over the others. The database Hepatitis is a counterexample, which can be explained by considering the slight difference between the base error and the average error of the grown/trained trees. Indeed, pruning and grafting operators can do no more than remove superfluous or even harmful branches: If no attribute is relevant, and the decision tree has a low predictive accuracy, the best result we could expect is a single node tree with the base error as estimated error rate. Thus, if the difference between the base error and the average error rate of grown/trained trees is not significant, there is no way will we observe a "+."

This point is essential to explain the different results we obtained for five databases on heart disease, despite their equal structure. In fact, almost all pruning methods improved the accuracy on the Hungary, Switzerland, and Long Beach data. In the latter two databases, the misclassification rate of the grown/trained trees was much greater than the base error, so even those methods that returned single node trees significantly improved the accuracy.

The databases Heart and Australian provide another example showing that the common structure of databases is not sufficient to explain the effect of pruning. The number of classes and attributes, as well as the distribution of cases per class, is the same in both domains. Furthermore, the percentage of real-value and multi-value attributes is virtually the same. Neither can the different sizes of the databases justify the disparity in the results. The reasons should be sought in the relevance of the attributes. In the heart disease domain, 10 out of 14 attributes are moderately correlated with the presence/absence of heart disease, but taken together they do not explain much variance in the data. On the contrary, some attributes of the Australian data are more strongly correlated with the class, so that the sparseness of data is more contained. As Schaffer [31] has already shown, the benefits in terms of accuracy of pruning are inversely proportional to the degree of sparseness, hence the worse performance of some methods on the Heart data.

Other general conclusions on the pruning methods can be drawn from Table 4 by comparing columns rather than rows. The number of databases in which each method reported a "+" or a "–" can give an idea of the appropriateness of the bias of each pruning method for the various domains considered. If we think about the cases in which we observed a certain decrease in accuracy, we should con-

TABLE 6
RESULTS OF THE TESTS ON ERROR RATES: EPB VS. OTHER METHODS

database	REP	MEP	CVP	0SE	1SE	PEP	CV-0SE	CV-1SE
Iris	**+ (.0892)**	**+ (.0092)**	+ (.2137)	**+ (.0427)**	**+ (.0250)**	**+ (.0830)**	**+ (.0026)**	**+ (.0205)**
Glass	+ (.1147)	+ (.1813)	+ (.5665)	+ (.2165)	**+ (.0041)**	- (.6630)	+ (.5195)	**+ (.0088)**
Led-1000	**+ (.0320)**	**+ (.0337)**	**+ (.0271)**	**+ (.0413)**	**+ (.0001)**	**+ (.0330)**	**+ (.0849)**	**+ (.0001)**
Led-200	**+ (.0056)**	**+ (.0804)**	**+ (.0167)**	**+ (.0825)**	**+ (.0010)**	+ (.6960)	+ (.6518)	**- (.0001)**
Hypo	+ (.1269)	**+ (.0848)**	**+ (.0189)**	+ (.1208)	**+ (.0212)**	**+ (.0557)**	+ (.1027)	**+ (.0082)**
P. gene	+ (.4373)	+ (.1052)	**+ (.0731)**	+ (.1220)	+ (.1277)	**+ (.0092)**	**+ (.0235)**	**+ (.0441)**
Hepatitis	- (.2859)	- (.9517)	- (.4162)	- (.2150)	- (.3871)	- (.6814)	- (.9103)	- (.7795)
Cleveland	- (.2360)	- (.9987)	+ (.3286)	- (.8088)	+ (.3588)	+ (.8550)	+ (.4816)	+ (.4467)
Hungary	- (.3953)	- (.8301)	**+ (.0047)**	- (.4592)	- (.6056)	- (.2916)	- (.4329)	**- (.0209)**
Switzerland	+ (.8011)	**+ (.0001)**	+ (.7031)	+ (.9992)	- (.3273)	(.0)	+ (.6786)	- (.3273)
Long Beach	**- (.0012)**	- (.4171)	**- (.0079)**	**- (.0489)**	**- (.0011)**	**- (.0162)**	**- (.0060)**	**- (.0011)**
Heart	+ (.1618)	**+ (.0763)**	**+ (.0125)**	**+ (.0403)**	**+ (.0189)**	+ (.5280)	**+(.0062)**	**+ (.0357)**
Blocks	**+ (.0811)**	**+ (.0002)**	**+ (.0001)**	+ (.1721)	**+ (.0036)**	- (.1366)	+ (.2146)	**+ (.0001)**
Pima	**- (.0001)**	**- (.0087)**	+ (.1049)	**- (.0003)**	**- (.0022)**	+ (.9647)	**- (.0182)**	**- (.0221)**
Australian	- (.5299)	- (.4859)	**+ (.0001)**	- (.4254)	- (.2154)	**- (.0022)**	**- (.0762)**	- (.2647)

clude that CV-1SE is the method with the worst performance, immediately followed by 1SE and REP. In this latter case, however, the number of "+" is high as well. As expected, a static behavior was observed for CVP, which improved the accuracy in only three domains and performed badly in two. At least four methods, namely 0SE, CV-0SE, EBP, and PEP, performed equally well. Interestingly, PEP and EBP produced significantly better trees for the same data sets, so that it is possible to postulate, on empirical grounds, the equivalence of the two methods despite the difference in their formulation. By summarizing these results, we can conclude that *there is no indication that methods exploiting an independent pruning set definitely perform better than the others*. This claim is at variance with the conclusions reported by Mingers [20]; this discrepancy should be attributed to the different design of the experiments.

To complete the analysis of error rates, it would be interesting to investigate the significance of differences among the methods that lead to an improvement. Since it is not possible to report all the possible comparisons, we decided to compare one of the methods that seems to be more stable, namely EBP, to the others. The sign of the t-values and the corresponding significance levels are shown in Table 6. A positive (negative) sign at the intersection of the *ith* row and *jth* column indicates that the EBP performed better (worse) than the method in the *jth* column for the database in the *ith* row. Henceforth, we will pay attention to those entries with a significance level lower than 0.10 (in bold).

From a quick look at the table, we can conclude that EBP does not always beat the other methods, especially in the prone-to-pruning domain. In four cases, CV-1SE produces more accurate decision trees, while any other method worked better than EBP in the Long Beach database. Thus, summarizing these observations with those made above, we can state that EBP performs well on average and shows a certain stability on different domains, but its bias toward underpruning presents some drawbacks in those domains where the average size of OPTTs is significantly lower than that of trained trees. Analogous results have been observed by setting up an experimental procedure based on cross-validation.

As for the error rate, again for the tree size we tested the significance of the differences by means of two-tailed paired t-tests. Table 7 summarizes the results when the confidence level of the test is 0.10. It should be borne in mind that the comparison involves OPGTs for those methods that operate on the pruning set and OPTTs for the others. Here, "u" stands for significant underpruning, "o" for significant overpruning, while "–" indicates no statistically relevant difference. The tests confirm that MEP, CVP, and EBP tend to underprune, while REP, 1SE, and CV-1SE have a propensity for overpruning.

To be fair, we must also point out that the results reported in the EBP column of Table 7 should be taken with a grain of salt, since OPTTs are optimal when pruning is the only operator used in a tree simplification problem. This is no longer true when the additional operator of grafting is considered, as in EBP. As future work, we plan to find an algorithm for optimally pruning and grafting branches of a decision tree, in order to make a fairer comparison.

Detailed information on the average size of (optimally) pruned trees is reported in Table 8. The table can be virtually split into two subtables, one to be used in a comparison of pruning methods that operate on the grown trees, and the other concerning those methods that prune trained trees. Thus, in the first subtable we find a confirmation that REP tends to overprune, since, in nine of the 11 databases considered, it produces trees with a smaller average size than that obtained by the optimal pruning. As pointed out in Section 2.1, the explanation of this bias should be attributed to the fact that the decision to prune a branch is based on the evidence in the pruning set alone. In a separate

TABLE 7
RESULTS OF THE TESTS ON TREE SIZE

database	REP	MEP	CVP	OSE	1SE	PEP	CV 0SE	CV 1SE	EBP
Iris	-	-	u	-	o	-	-	o	u
Glass	-	u	u	-	o	u	u	o	u
Led-1000	-	u	u	u	o	o	u	o	u
Led-200	o	-	u	-	o	o	-	-	-
Hypo	o	u	u	-	o	-	-	o	u
P-gene	o	u	u	-	o	o	-	o	u
Hepatitis	-	u	-	-	o	-	-	o	u
Cleveland	-	-	u	-	o	u	-	o	u
Hungary	o	-	u	o	o	-	-	o	u
Switzerland	-	u	-	-	-	-	-	-	-
Long Beach	-	u	-	-	o	-	-	o	u
Heart	-	-	u	-	o	o	o	o	-
Blocks	-	u	u	-	o	u	o	o	u
Pima	-	u	u	o	o	u	o	o	u
Australian	o	o	u	o	o	o	o	o	u

Significance level: 0.10.

study on the REP, we have also observed that, in most of the domains considered in this paper, the optimal size of pruning set is about 70 percent of the training set. This excludes the possibility that REP might have been under a disadvantage in our experiments.

Table 8 also confirms the bias of MEP, CVP, and EBP towards underpruning; in particular, CVP does not prune at all in two domains, namely Iris and Glass. The reasons for such a behavior have been presented in Section 2. No clear indication is given for CV-0SE and PEP: In this latter case, the cost attributed to each leaf (1/2) seems more appropriate for some problems, but less for others.

We would be tempted to conclude that the predictive accuracy is improved each time a pruning method produces trees with no significant difference in size from the corresponding optimally pruned trees. However, this is not true for two reasons. Firstly, it is not always true that an optimally pruned tree is more accurate than the corresponding grown/trained tree. Indeed, pruning may help to simplify trees without improving their predictive accuracy. Secondly, tree size is a global feature that can provide us with an idea of what is happening, but it is not detailed enough to guarantee that only overpruning or underpruning occurred. For instance, if a method overprunes a branch with two leaves but underprunes another with the same number of leaves, then it is actually increasing the error rate with respect to the optimal tree, but not necessarily the size. This problem can be observed with the Glass data and the REP method. In this case, indeed, there is a significant decrease in accuracy whereas the size of pruned trees is close to the optimal value.

By ideally superimposing Table 4 and Table 7, it is also possible to draw some other interesting conclusions. For instance, in some databases, such as Hungary and Heart, overpruning produces better trees than underpruning. This latter result agrees with Holte's observation that even simple rules perform well on most commonly used data sets in the machine learning community (1993). It is also a confirmation that the problem of *overfitting* the data affects TDIDT systems.

The theoretical explanation of such a phenomenon has to be sought in the partitioning strategy adopted by such systems. Indeed, a decision tree can be regarded as a recursive partitioning of the feature space. Each internal node is associated with a partition that is, in turn, split further into several regions assigned to only one of the internal node's children. Generally, when the feature space is partitioned into few large regions, the decision tree is not even able to explain the training data. Statisticians call this lack of fitting to the data *bias*. By progressively splitting each region, the bias decreases, and, together with it, the number of training examples falling in each single unsplit region. Unfortunately, this means that the estimates of the a posteriori probabilities exploited by the decision rules are less reliable, hence the probability of labelling that region with a different class from the Bayes optimal one increases. This latter probability has been called *variance*, thus we can say that the true problem in tree pruning is a trade-off between bias and variance. Generally, neither bias nor variance can be known precisely, and we have to rely on two surrogate measures, such as the number of examples in each region and the number of leaves. In the light of such considerations, it is not difficult to accept Schaffer's claim [32] that tree pruning is a form of *bias* (here intended as a set of factors that influence hypothesis selection) rather than a statistical improvement of the classifier.

TABLE 8
AVERAGE SIZE OF TREES OBTAINED WITH DIFFERENT PRUNING METHODS

database	REP	MEP	CVP	0SE	1SE	OPGT	PEP	CV-0SE	CV-1SE	EBP	OPTT
Iris	3.4 ± .141	4 ± .231	5.4 ± .265	3.44 ± .201	3.16 ± .149	3.64 ± .19	3.76 ± .133	4.04 ± .261	3.125 ± .151	4.88 ± .247	4.0 ± .173
Glass	11.04 ± .861	18.52 ± 1.088	27.92 ± .707	14.12 ± 1.774	7.04 ± .799	12.6 ± .849	21.12 ± .498	20.28 ± 2.755	7.52 ± 1.551	28.72 ± .576	15.08 ± .954
Led-1000	20.52 ± .542	25.28 ± 1.666	36 ± . 806	25.64 ± 1.924	12.32 ± .66	21.44 ± .651	18.32 ± .325	31.48 ± 2.171	15.68 ± 1.395	30.6 ± .58	22.04 ± .552
Led-200	13.4 ± .545	18.76 ± 1.077	24.4 ± .658	17.04 ± 1.239	12.68 ± .862	17.92 ± .597	13.6 ± .451	22.52 ± 1.596	20.16 ± .663	20.68 ± .725	20.16 ± .663
Hypo	8.2 ± .404	15.76 ± .623	14 ± .643	8.64 ± .472	7.2 ± .408	9.16 ± .457	9.32 ± .222	9.6 ± .566	7.32 ± .269	13.68 ± .382	9.36 ± .483
P. gene	6.4 ± .424	10 ± .714	13.36 ±.969	7.24 ± 1.081	4.36 ± .469	7.84 ± .637	8.44 ± .578	8.92 ± 1.147	5.68 ± .602	15.28 ± .853	10.0 ± .52
Hepatitis	2.64 ± .391	8.36 ± .412	3.6 ±.728	2.56 ± .469	1.36 ± .151	3.4 ± .539	5.44 ± .469	3.8 ± .619	1.8 ± .30	9.08 ± .479	4.36 ± .622
Cleveland	10.92 ± .885	15 ± 1.371	30.76 ± 1.028	9.88 ± 1.803	4.12 ± .533	12.2 ± 1.03	19 ± .833	15.52 ± 3.278	4.84 ± .875	29.28 ± 1.075	16.36 ± 1.03
Hungary	7.32 ± .95	10.44 ± 1.522	21 ± 2.292	5.24 ± 1.455	2.28 ± .178	10.16 ± .967	10.2 ± 1.153	8.92 ± 2.398	2.68 ± .427	17.12 ± 1.431	9.64 ± 1.063
Switzerland	1.4 ± .224	8.92 ± .58	1.68 ± .479	1.56 ± .388	1 ± .0	1.32 ± .222	1.16 ± .16	1.36 ± .251	1 ± .0	1.08 ± .08	1.16 ± .16
Long Beach	6.16 ± 1.271	17.16 ± 1.778	7.28 ± 2.068	6.92 ± 2.164	1.4 ± .277	4.12 ± .667	4.96 ± 1.029	4.04 ± 1.73	1.04 ± .04	10.88 ± 1.288	4.92 ± .922
Heart	33.52 ± 2.199	31.92 ± 3.548	58.2 ± 4.833	27.52 ± 4.921	7.2 ± .978	36.16 ± 2.76	21.92 ± 1.31	29.4 ± 6.793	7.2 ± 1.147	46.04 ± 1.459	45.96 ± 2.91
Blocks	24.68 ± .979	65.04 ± 1.277	78.48 ± 1.277	28.12 ± 2.964	13.12 ± 1.721	25.84 ± 1.03	37.24 ± .735	17.56 ± 2.867	8.08 ± .443	50.92 ± 1.29	30.44 ± 1.1
Pima	18.48 ± 1.501	32.4 ± 3.744	70.88 ± 1.159	12.68 ± 3.051	4.28 ± .639	20.4 ± 1.53	55.72 ± 1.243	9.2 ± 2.598	2.84 ± .411	65.84 ± 1.497	22.68 ± 1.63
Australian	11 ± 1.492	7.12 ± 1.39	52.72 ± 2.162	6.96 ± 1.71	3.32 ± .602	15.64 ± 2.06	20.28 ± 1.356	4.48 ± .755	2.52 ± .289	33.88 ± 2.186	24.76 ± 1.86

4 RELATED WORK

Other empirical comparisons of pruning methods have already appeared in the machine learning literature. The first of them was made by Quinlan [25]. In his paper, four methods for simplifying decision trees are considered, three of which are the known 1SE, REP, and PEP, while the fourth is based on a reformulation of a decision tree as a set of production rules. Among the various data sets considered, we selected two in our empirical study, namely LED and Hypothyroid. Nevertheless, the differences in the experimental setup frustrate any attempt to compare our results with his. Quinlan, indeed, partitions data sets as follows: training set (approximately 66 percent of all available data), first test set (17 percent), and second test set (17 percent). Sampling is stratified by class, so that the proportion of cases belonging to each class is made as even as possible across the three sets. Moreover, only REP and 1SE exploit data of the first test set for pruning, while PEP does not. Therefore, the experimental procedure favors those methods that exploit an additional pruning set.

The same problem affects Mingers' empirical comparison as well [20], and justifies the discrepancy with some of our findings. On the other hand, Mingers' study involves four selection measures, and investigates possible interactions with the pruning method, while our analysis is limited to only one of those measures, namely the gain-ratio [24].

Another work that considers several selection measures is Buntine's thesis [3]. In this case, six pruning methods are considered, namely 10-fold CV-0SE, 10-fold CV-1SE, 0SE, 1SE, PEP, and REP. The major trends that emerge are:

- A marginal superiority of CV-0SE over the other methods.
- A superiority of PEP in those domains with a good deal of structure, such as LED and Glass, and the appropriateness of the 1SE rule for those data sets with little apparent structure.

Our findings confirm the second trend but not the first, although our experimental design largely follows his. The explanation of such a divergence should be sought in the different databases as well as in the different selection measures that Buntine considered (information gain, information gain with Marshall correction, and Gini index of diversity). Furthermore, the method that performed better in our study, EBP, was not considered in his experiments. In his study, no attention was paid to the optimality property of the reduced error pruning, and in general, to the optimality of pruned trees. The critical assumptions of the cost-complexity pruning were not discussed, and as "parsimony of the trees is not relevant" in his study, no consideration was made on the size of the final trees.

Another related empirical comparison can be found in [31]. This paper shows that the effect of pruning depends on the abundance of training data relative to the complexity of the true structure underlying data generation. In particular, for sparse data, PEP and CV-0SE decrease the predictive accuracy of induced trees. This result is not surprising: Sparseness of data generally leads to decision trees with few covered cases per leaf, hence the problem of trading off bias and variance.

This paper on decision tree pruning is manifestly incomplete: Space constraints obliged us to neglect several other pruning methods presented in the literature.

One of them [27] is part of a decision tree induction process based on the minimum description length (MDL) principle [28]. Given an efficient technique for encoding decision trees and exceptions, which are examples misclassified by the decision tree, the MDL principle states that the best decision tree is the one that minimizes the total length of the codes for both the tree and the exceptions. The authors propose a two-phase process in which a decision tree is first grown according to the classical TDIDT approach and then pruned. An internal node t can be pruned only if all of its children are leaves, hence the bottom-up strategy. The data used in the second phase are the same as for building the tree. The reason why we did not consider the MDL-based pruning method is that it is closely related to the training phase in which the MDL selection criterion is adopted. However, as already pointed out in Section 3, our experimentation was intentionally restricted to trees grown by using the gain-ratio selection measure.

Another method we have not considered is the iterative growing and pruning algorithm proposed by Gelfand, Ravishankar, and Delp [11]. The reasons are mainly three. Firstly, because for space constraints we decided to concentrate our attention on non-backtracking top-down approaches to decision tree induction, while Gelfand et al. frame their pruning method into a growing-pruning approach [29]. Secondly, because the pruning method described by Gelfand et al. does not differ from REP. Thirdly, because it is appropriate for complete data sets and there is no guarantee of convergence of the iterative process when null values are managed, unlike C4.5. In any case, the reader can find some results on the Iris and Glass data in [9].

In this paper, we have focused our attention on the problem of obtaining a better *generalization performance* by means of pruning techniques. Nonetheless, a recent paper has looked at pruning as a way of *trading accuracy for simplicity* of a concept description [1]. More precisely, given a decision tree that accurately specifies a concept, Bohanec and Bratko set the problem of finding a smallest pruned tree that still represents a concept within a specified accuracy. The goal is no longer that of improving the generalization performance, but that of producing a sufficiently accurate, compact description of a given complexity. In other words, the idea is that of simplifying a decision tree to improve its comprehensibility, and in this context, an *optimally pruned tree* has the property of being the smallest pruned tree whose *apparent* error rate is not greater than some given level of misclassification. This definition should not be confused with that given in the paper, since this latter refers to the smallest tree with the highest accuracy on either the pruning or the test sets.

Bohanec and Bratko developed an algorithm, named OPT, that generates a sequence of optimally pruned trees, decreasing in size, in time $O\left(\left|\mathfrak{L}_{T_{\max}}\right|^{2}\right)$. Actually, OPT can be used to generate a sequence of pruned trees from which to choose the most accurate on an independent pruning set. However, as already pointed out in Section 2, any two-phase pruning method is under a disadvantage with respect to REP, which finds the best tree among all possible subtrees. The same Bohanec and Bratko observe that no significant gains in classification accuracy can be expected in general.

5 Conclusions

In this paper, a comparative study of six well-known pruning methods has been presented. In particular, each method has been critically reviewed and its behavior tested on several data sets. Some strengths and weaknesses of the theoretical foundations of many approaches for simplifying decision trees have been pointed out. In particular, we proved that the algorithm proposed for the REP finds the smallest subtree with the lowest error rate with respect to the pruning set, and we employed this property to build the optimally pruned trained/grown trees in our experiments. OPGTs and OPTTs can profitably be exploited in two ways. Firstly, they give an objective evaluation of the tendency to overprune/underprune showed by each method. Secondly, they are good tools for studying some properties of the available data sets, such as the decrease in size of optimally pruned trees with respect to the corresponding grown/trained trees or the increase in accuracy obtainable with optimal pruning.

To sum up, in this paper, we have shown that:

- MEP, CVP, and EBP tend to underprune, whereas CV-1SE, 1SE, and REP have a propensity for overpruning.
- Setting aside some data for pruning only is not generally the best strategy.
- PEP and EBP behave in the same way, despite their different formulation.
- Pruning does not generally decrease the predictive accuracy of the final trees; indeed, only one of the domains considered in our study could be classified as refractory to pruning.
- Almost all databases not prone to pruning have the highest base error, while those databases with a relatively low base error benefit of any pruning strategy.

Among other minor empirical results, it is worthwhile recalling the confirmation of some characteristics of the PEP and 1SE methods already observed in previous empirical studies.

Pruning methods have been implemented as an extension of C4.5, a system developed by J.R. Quinlan and distributed by Morgan Kaufmann. Only additional source files ("*Potato*" routines) developed at the Department of Informatics of the University of Bari are available upon request by e-mail to malerba@lacam.uniba.it.

Acknowledgments

The authors thank Maurizio Giaffreda and Francesco Pagano for their precious collaboration in conducting the experiments. They also wish to thank Ivan Bratko, Marko Bohanec, Hussein Almuallim, and the anonymous referees for their useful comments and criticisms. Many thanks to Lynn Rudd for her help in rereading the paper.

References

[1] M. Bohanec and I. Bratko, "Trading Accuracy for Simplicity in Decision Trees," *Machine Learning*, vol. 15, no. 3, pp. 223-250, 1994.

[2] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*. Belmont, Calif.: Wadsworth Int'l, 1984.

[3] W.L. Buntine, *A Theory of Learning Classification Rules*, PhD Thesis, Univ. of Technology, Sydney, 1990.

[4] W.L. Buntine and T. Niblett, "A Further Comparison of Splitting Rules for Decision-Tree Induction," *Machine Learning*, vol. 8, no. 1, pp. 75-85, 1992.

[5] B. Cestnik, I. Kononenko, and I. Bratko, "ASSISTANT 86: A Knowledge-Elicitation Tool for Sophisticated Users," *Progress in Machine Learning—Proc. EWSL-87*, I. Bratko and N. Lavrac, eds. Wilmslow: Sigma Press, pp. 31-45, 1987.

[6] B. Cestnik and I. Bratko, "On Estimating Probabilities in Tree Pruning," *Machine Learning: EWSL-91*, Y. Kodratoff, ed., *Lecture Notes in Artificial Intelligence*. Berlin: Springer-Verlag, no. 482, pp. 138-150, 1991.

[7] E.S. Edgington, *Randomization Tests*, 2nd ed., New York, N.Y.: Marcel Dekker, 1987.

[8] B. Efron and G. Gong, "A Leisurely Look at the Bootstrap, the Jackknife, and Cross-Validation," *The American Statistician*, vol. 37, pp. 36-48, 1983.

[9] F. Esposito, D. Malerba, and G. Semeraro, "Decision Tree Pruning as a Search in the State Space," *Machine Learning: ECML-93*, P. Brazdil, ed. *Lecture Notes in Artificial Intelligence*, Berlin: Springer-Verlag, no. 667, pp. 165-184, 1993.

[10] U. Fayyad and K.B. Irani, "The Attribute Selection Problem in Decision Tree Generation," *Proc. AAAI-92*, pp. 104-110, 1992.

[11] S.B. Gelfand, C.S. Ravishankar, and E.J. Delp, "An Iterative Growing and Pruning Algorithm for Classification Tree Design," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, no. 2, pp. 138-150, 1991.

[12] R.C. Holte, "Very Simple Classification Rules Perform Well on Most Commonly Used Datasets," *Machine Learning*, vol. 11, no. 1, pp. 63-90, 1993.

[13] R.C. Holte, L.E. Acker, and B.W. Porter, "Concept Learning and the Problem of Small Disjuncts," *Proc. 11th Int'l Joint Conf. on Artificial Intelligence*, pp. 813-818, 1989.

[14] J. Kittler and P.A. Devijver, "Statistical Properties of Error Estimators in Performance Assessment of Recognition Systems," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 4, no. 2, pp. 215-220, 1982.

[15] R. Kohavi and G.H. John, "Automatic Parameter Selection by Minimizing Estimated Error," *Proc. 12th Int'l Conf. on Machine Learning*, Lake Tahoe, Calif., pp. 304-312, 1995.

[16] D. Malerba, G. Semeraro, and F. Esposito, "Choosing the Best Pruned Decision Tree: A Matter of Bias," *Proc. Fifth Italian Workshop on Machine Learning*, Parma, Italy, pp. 33-37, 1994.

[17] D. Malerba, F. Esposito, and G. Semeraro, "A Further Comparison of Simplification Methods for Decision-Tree Induction," *Learning From Data: Artificial Intelligence and Statistics V*, D. Fisher and H. Lenz, eds., *Lecture Notes in Statistics.* Berlin: Springer, no. 112, pp. 365-374, 1996.

[18] J. Mingers, "Expert Systems—Rule Induction With Statistical Data," *J. Operational Research Society*, vol. 38, pp. 39-47, 1987.

[19] J. Mingers, "An Empirical Comparison of Selection Measures for Decision-Tree Induction," *Machine Learning*, vol. 3, no. 4, pp. 319-342, 1989.

[20] J. Mingers, "An Empirical Comparison of Pruning Methods for Decision Tree Induction," *Machine Learning*, vol. 4, no. 2, pp. 227-243, 1989.

[21] P.M. Murphy and D.W. Aha, "UCI Repository of Machine Learning Databases [Machine Readable Data Repository]," Univ. of California, Dept. of Information and Computer Science, Irvine, Calif., 1996.

[22] T. Niblett, "Constructing Decision Trees in Noisy Domains," *Progress in Machine Learning, Proc. EWSL 87*, I. Bratko and N. Lavrac, eds. Wilmslow: Sigma Press, pp. 67-78, 1987.

[23] T. Niblett and I. Bratko, "Learning Decision Rules in Noisy Domains," *Proc. Expert Systems 86*, Cambridge: Cambridge University Press, 1986.

[24] J.R. Quinlan, "Induction of Decision Trees," *Machine Learning*, vol. 1, no. 1, pp. 81-106, 1986.

[25] J.R. Quinlan, "Simplifying Decision Trees," *Int'l J. Man-Machine Studies*, vol. 27, pp. 221-234, 1987.

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

[27] J.R. Quinlan and L.R. Rivest, "Inferring Decision Trees Using the Minimum Description Length Principle," *Information and Computation*, vol. 80, pp. 227-248, 1989.

[28] J. Rissanen, "A Universal Prior for Integers and Estimation by Minimum Description Length," *Annals of Statistics II*, pp. 416-431, 1983.

[29] S.R. Safavian, and D. Landgrebe, "A Survey of Decision Tree Classifier Methodology," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 21, no. 3, pp. 660-674, 1991.

[30] C. Schaffer,"Deconstructing the Digit Recognition Problem," *Proc. Ninth Int'l Workshop on Machine Learning*, pp. 394-399. San Mateo, Calif.: Morgan Kaufmann, 1992.

[31] C. Schaffer, "Sparse Data and the Effect of Overfitting Avoidance in Decision Tree Induction," *Proc. AAAI-92*, pp. 147-152, 1992.

[32] C. Schaffer, "Overfitting Avoidance As Bias," *Machine Learning*, vol. 10, no. 2, pp.153-178, 1993.

[33] C.J.C.H. Watkins, "Combining Cross-Validation and Search," *Progress in Machine Learning, Proc. EWSL 87*, I. Bratko and N. Lavrac, eds. Wilmslow: Sigma Press, pp. 79-87, 1987.

[34] P.A. White and W.Z. Liu, "Bias in Information-Based Measures in Decision Tree Induction," *Machine Learning*, vol. 15, no. 3, pp. 321-329, 1994.

Floriana Esposito is full professor of computer science at the University of Bari. Her research activity started in the field of numerical models and statistical pattern recognition applied to the field of medical diagnosis. Then her interests moved to knowledge-based systems related to the problem of automatic knowledge acquisition and machine learning. In particular, she is involved in similarity based learning, trying to integrate numerical and symbolic methods, in order to cope with the problem of learning in noisy environments and to efficiently learn from both continuous and nominal data. A further interest is the integration of inductive and deductive learning techniques for the incremental revision of logical theories.

Donato Malerba received the Laurea degree with honors (computer science) from the University of Bari, Italy in 1987. In 1991 he joined the University of Bari where he is assistant professor in computer science. In 1992 he was a visiting scientist at the Department of Information and Computer Science, University of California at Irvine. His research interests focus on the analysis and synthesis of algorithms for inductive inference: classification and regression trees, causal inference, and induction of logic theories from numeric/symbolic data. Applications include intelligent document processing, knowledge discovery in databases, map interpretation, and interfaces.

Giovanni Semeraro received a Laurea degree with honors in computer science from the University of Bari, Italy, in 1988. From 1989 to 1991, he was a researcher at Tecnopolis CSATA Novus Ortus, Bari, Italy, in the area of advanced software technology. He joined the faculty of University of Bari in 1991, where he currently holds the rank of assistant professor in the Department of Computer Science. He has been a visiting scientist at the Department of Information and Computer Science, University of California at Irvine, in 1993. His research interests are centered on the logical and algebraic foundations of inductive inference, document classification and understanding, multistrategy learning, theory revision, and intelligent digital libraries.

View publication stats