35

A Further Comparison of Simplification Methods for Decision-Tree Induction

Donato Malerba, Floriana Esposito, and Giovanni Semeraro

Dipartimento di Informatica - Università degli Studi di Bari
via Orabona, 4 - 70126 Bari - Italy
{malerbad | esposito | semeraro} @ vm.csata.it

ABSTRACT This paper presents an empirical investigation of eight well-known simplification methods for decision trees induced from training data. Twelve data sets are considered to compare both the accuracy and the complexity of simplified trees. The computation of optimally pruned trees is used in order to give a clear definition of bias of the methods towards overpruning and underpruning. The results indicate that the simplification strategies which exploit an independent pruning set do not perform better than the others. Furthermore, some methods show an evident bias towards either underpruning or overpruning.

35.1 Introduction.

Various heuristic methods have been proposed for the construction of a decision tree, among which the most widely known is the top-down approach. In top-down induction of decision trees (TDIDT) it is possible to identify three main issues [BFOS84]:

1. The definition of a decision process associated with the tree.
2. The determination of the tests in the nodes.
3. The determination of the leaves.

This paper is mainly concerned with the third one, which is deemed to be important because of its influence on the overfitting problem. In general, a tree *overfits* the training data when some of its branches degrade the classification performance on unseen cases, and, as a minor consequence, reduce the comprehensibility of the tree itself. This problem is particularly felt when several sources of noise and uncertainty affect the training data. In this case, indeed, the decision of stopping the growth of a tree when all examples in each leaf belong to the same class is too weak, since it does not prevent the generation of harmful branches. A typical approach adopted in many TDIDT systems grows a tree, T_{max}, to maximum depth, and then retrospectively removes those branches that seem superfluous with respect to predictive accuracy [Nib87]. The final effect should be that of improving the intelligibility of a decision tree without really affecting its predictive accuracy. Recently, Fisher [Fis92] and Schaffer [Sch93] have shown that overfitting avoidance by means of 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. Nevertheless,

[1] *Learning from Data: AI and Statistics V.* Edited by D. Fisher and H.-J. Lenz.

this conclusion should not imply the abandonment of pruning techniques, rather, it should stimulate the investigation of the biases of different methods so to apply the appropriate method, if any, to the problem at hand.
This paper investigates the behaviour of eight well-known simplification methods on twelve data sets taken from the UCI Machine Learning Repository [MA94]. In particular, we show how the computation of optimally pruned trees can be used to give a definition of bias of a method towards either underpruning or overpruning. This experimental approach generalizes an approach by Holte [Hol93], who found upperbounds on accuracy when decision tree depth was limited to one. The upperbound was computed by randomly splitting the dataset into a training and a test set, and then measuring the highest accuracy on the test set of all 1-level decision trees built on the training set. Holte did not consider the possibility of simplifying the trees and consequently the complexity of the induced trees. Conversely, we are interested in finding lower bounds on the complexity of the *most accurate* simplified trees, as well as upper bounds on the accuracy achievable by simplifying trees. The experimental design of Section 3 embodies a number of differences from previous studies on pruning methods; these differences allow us to draw some conclusions on the performance of the methods that are at variance with those already published in the literature or expected from the formulation of the methods themselves. A summary of the empirical results is reported in Section 4.

35.2 A brief survey of simplification methods.

The simplification methods considered in our study are briefly described in Table 1. Some of them start the simplification process from the root of T_{max} and proceed towards the leaves (top-down), while others move from the leaves to the root (bottom-up). Moreover, some methods simplify the tree in two steps: they first generate a set of subtrees of T_{max} by taking into account the complexity of the tree, then they select the best one in the set according to its estimated accuracy. In some cases, the estimate of the accuracy of a tree is computed on a *pruning set* which is independent of the set used for building the tree T_{max}. This means that the *training set* is actually split into two subsets: the *growing set* for building the tree T_{max} and the *pruning set* for simplifying it. Two of the three variants of the cost-complexity pruning method proposed by Breiman et al. require a pruning set (0SE and 1SE), while the other one (CV-0SE) uses cross-validation sets to estimate the predictive accuracy of the subtrees. In this latter case T_{max} is built on the whole training set and not on the smaller growing set. The version of the minimum error pruning (MEP) method, that we implemented for our experiments, is based on techniques proposed by Cestnik and Bratko [CB91], but differs in that we use a pruning set for choosing amongst trees pruned with different values of the parameter m.[2] Furthermore, our reduced error pruning (REP) algorithm is based on Quinlan's original formulation instead of that given by Mingers [Min89], so that we are guaranteed that it finds the smallest version of the most accurate subtree with respect to the pruning set. Details of the methods REP, MEP, 0SE, 1SE, and pessimistic error pruning (PEP) can be found in [EMS93].

[2]The parameter m determines the impact of the prior probability on the estimation of the error rate. In our experiments, we selected the following values of m: 0.5, 1, 2, 3, 4, 8, 12, 16, 32, 64, 128, 512 and 1024. For each value, MEP returns a subtree of T_{max}. The pruning set is used to select the best subtree.

Method	Steps	Pruning Set	Traversal Strategy	Operators	Characteristics
Reduced Error Pruning (REP) [Quinlan, 1987]	1	yes	bottom-up	pruning any branch	Finds the optimally pruned tree w.r.t. the pruning set.
Pessimistic Error Pruning (PEP) [Quinlan, 1987]	1	no	top-down	pruning any branch	Uses the continuity correction.
Minimum Error Pruning (MEP) [Cestnik & Bratko, 1991]	1-2	yes *last version*	bottom-up	pruning any branch	Based on the m-probability estimate of the error rate.
Critical Value Pruning (CVP) [Mingers, 1989]	2	yes	bottom-up	pruning branches of depth one	Selects the best tree on a subset of the space of all possible subtrees.
Cost-Complexity Pruning (0SE, 1SE and CV-0SE) [Breiman et al., 1984]	2	yes *0SE and 1SE*	all nodes considered together	pruning any branch	Can use cross-validation sets to estimate the accuracy.
Error-Based Pruning (EBP) [Quinlan, 1993]	1	no	bottom-up	pruning and grafting any branch	Estimates confidence intervals on the training set.

TABLE 35.1. Simplification methods considered in the experiments.

Finally, another characteristic of the methods is the type of *simplification* operators used to trim the tree, namely

- *Pruning branches of depth one.* In this case a branch of depth n can be pruned in n steps.
- *Pruning branches of any depth.*
- *Grafting a sub-branch* of a node t onto the place of t itself, thus removing only some of the nodes of the subtree rooted in t.

Currently, the only simplification method that combines grafting and pruning operators is the error based pruning (EBP) implemented in C4.5 [Qui93]. All pruning methods have been implemented as an extension of C4.5, so that we could use a well-tested system to generate the decision trees T_{max}'s.

35.3 Design of the experiment.

Major properties of the data sets considered in our experiments are summarized in Table 2. In particular, for each database we report the following information:

- The number of cases available in the data set.
- The number of classes of the training cases.
- The number of attributes used to describe each example.
- The number of attributes that are treated as real-valued.
- The number of non-numerical attributes with more than two values.
- The presence of null values in the description of an example.[3]

[3]Null values are treated according to the standard procedure implemented in C4.5.

database	No. cases	No. classes	No. attributes	Real-valued attributes	Multi-valued attributes	Null values	% Base error	Noise level	Uniform distrib.
Iris	150	3	4	4	0	no	66.67	low	yes
Glass	214	7	9	9	0	no	64.49	low	no
Led	1000	10	7	0	0	no	90	10%	yes
Hypo	3772	4	29	7	1	yes	7.7	no	no
P.-gene	106	2	57	0	57	no	50	no	yes
Hepat.	155	2	19	6	0	yes	20.65	no	no
Cleveland	303	2	14	5	5	yes	45.21	low	yes
Hungary	294	2	14	5	5	yes	36.05	low	no
Switzerland	123	2	14	5	5	yes	6.5	low	no
Long Beach	200	2	14	5	5	yes	25.5	low	no
Heart	920	2	14	5	5	yes	44.67	low	yes
Blocks	5473	5	10	10	0	no	10.2	low	no

TABLE 35.2. Major properties of the data sets considered in the experimentation.

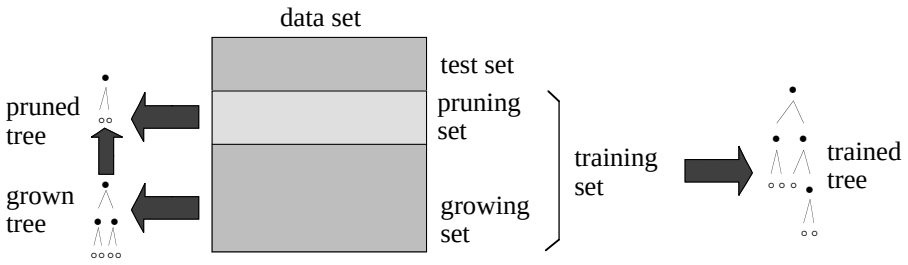

FIGURE 1. Partitioning of data available in each data set.

- The error rate obtained if the most frequent class is always predicted.
- The expected amount of noise in the database.
- The uniformity of distribution of training examples per class.

The data base Heart-disease is actually the join of four data sets with the same number of attributes but collected in four distinct places (Hungary, Switzerland, Cleveland and Long Beach). Of the 76 original attributes, only 14 have been considered, since they are the only ones deemed useful for the classification. Moreover, examples have been assigned to two distinct classes: no presence (value 0) and presence of heart diseases (values 1,2,3,4). For each data set considered, 25 experiments have been made by randomly partitioning the data set into three subsets: growing set (49%), pruning set (21%) and test set (30%) (see Figure 1). The union of the growing and pruning sets is called *training set*, and its size is just 70% of the whole data set. Therefore, the growing set contains the 70% of the cases of the training set, while the pruning set the remaining 30%. Both the growing and the training set are used to learn decision trees, that are called *grown tree* and *trained tree*, respectively. The former is used by those methods that need an independent pruning set in order to prune a decision tree, namely REP, MEP, CVP, and the cost-complexity pruning based on an independent test set and adopting the one standard error rule (1SE) or not (0SE) [BFOS84]. The latter is used by those methods that exploit the training set only, such as PEP, EBP, and the cost-complexity pruning based on cross-validation sets and not adopting the one standard error rule (CV-0SE).

In each experiment two statistics are recorded for pruned, grown and trained trees: the number of leaves (size) of the resultant tree and the error rate (e.r.) of the tree on the test set. As proposed by Buntine and Niblett [BN92], a two-tailed paired t-test is used to compare differences between pairs of either error or size averages. This is a major deviation from Mingers experimental design [Min89], where the ANOVA significance testing is preferred. Another major difference is that PEP, EBP, and CV-0SE have access to the whole training set rather than to the smaller growing set. Indeed, they simplify the trained trees and not the grown trees as the other ones. In a certain sense, there is a trade-off between the possibility of building a better tree T_{max} with all training cases and the lack of fresh cases for the pruning phase. Thus, *all* simplification methods can use the same set of examples, but REP, MEP, CVP, 0SE and 1SE will put aside some cases only for the pruning process. Finally, it is worthwhile to observe that Mingers investigated the possibility of interactions between four selection measures and pruning methods, while our study is limited to only one of those measures, namely the gain-ratio [Qui93].

In order to get an insight of the characteristics of some domains and pruning methods, we produced two decision trees for each experiment, called *optimally pruned grown-tree (OPGT)* and *optimally pruned trained-tree (OPTT)* respectively. The former is a grown tree that has been pruned by using the reduced error pruning on the *test set*. Thus, it is the best pruned tree we could produce from the grown tree because of the property of the reduced error pruning we mentioned in Section 2. 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. Obviously, such an estimate is rather optimistic: the error rates of these optimal trees can even be 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. Moreover, the magnitude of differences in accuracy of OPGTs and OPTTs can help to understand if the ideal goal of those simplification methods that requires a pruning set is far from the ideal goal of 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 the others.

The size of the optimally pruned trees can be exploited to establish 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, tree size is a coarse measure, since it may happen that a branch with p leaves is erroneously pruned while an equally-sized harmful branch is not. In this latter case, indeed, the size of the resultant tree is optimal even though there are both an overpruning and an underpruning problem. However, as we will show later, this does not prevent us from detecting the most serious of the two possible faults occurring in a simplification process. The real limitation of our empirical study concerns the comparison of the average size of OPTTs with that of trees produced by the EBP. This is due to the fact that REP, the method used to produce OPTTs, uses only a pruning operator, while EBP also adopts a grafting operator (see Table 1). Therefore, OPTTs are not optimal with respect to the space explored by EBP. Although

		Iris	Glass	Led	Hypo	P-gene	Hepat.	Clev.	Hung.	Switz.	L.B.	Heart	Block
GROWN	**size**	5.4	27.92	41.33	20.24	19	11.24	33.76	32.56	9.64	31.64	115.7	82.92
	e.r.	5.866	36.87	27.96	.622	27.12	22.81	30.07	25.5	13.84	33.27	24.72	3.65
OPGT	**size**	3.64	12.6	21.44	9.16	7.84	3.4	12.2	10.16	1.32	4.12	36.16	25.84
	e.r.	4.888	31.62	25.64	.399	18.50	15.83	22.99	16.91	4.622	23.6	18.77	2.609
TRAINED	**size**	6.84	36.32	44.68	27.24	25.6	16.76	50	47.88	11.76	43.48	164.6	111.9
	e.r.	5.598	35.38	27.48	.604	23.5	21.87	29.1	25.5	13.3	33.73	23.82	3.57
OPTT	**size**	4	15.08	22.04	9.36	10	4.36	16.36	9.64	1.16	4.92	45.96	30.44
	e.r.	4.442	28.31	25.31	.352	16.5	16.34	20.84	17.27	5.731	23.13	17.71	2.354

TABLE 35.3. Average size and error rate of the (optimally pruned) grown/trained trees.

we did not observe a frequent application of the grafting operator in our experiments, we plan to address this limitation in the future work.

35.4 Experimental results.

The average size and error rate of the (optimally pruned) grown/trained trees for each domain is reported in Table 3.

The ratio *(grown tree size/OPGT size)* ranges from 1.5 for the Iris data to 7.7 for the Long Beach data, while the ratio *(trained tree size/OPTT size)* is even greater than 10 for the Switzerland data. Such strong 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 2. Indeed, for the "incriminated" data set, there are only two classes, one of which contains only 6.5% of cases. Since the learning system fails in finding an adequate hypothesis for those cases, the pruning method will generally tend to prune the tree up to the root.

Another point is the greater accuracy of trained trees: this means that those methods that require a pruning set are under a disadvantage with respect to the others. The only exception is represented by the database Hungary, in which the greater number of training instances does not increase the accuracy of the trees but does increase their complexity.

By restricting our attention to only two domains (Iris and Led[4]) also used in Mingers' empirical comparison, we could detect a certain discrepancy between the two figures on the average size of unpruned trees:[5] 6.9 compared to 5.4 in Iris, and 56.6 compared to 41.33 in Led. Such a difference may be attributed to the percentages of cases used to build the trees (60% in Mingers' study versus 49% in ours). Indeed, by looking at the average size of trained trees which are built on 70% of cases, this discrepancy is significantly reduced, at least for the Iris domain. The discrepancy for the Led data should be mainly attributed to the tree collapsing process, implemented in C4.5, that stops branching when the best split does not reduce the apparent error rate. However, such differences do not compromise the validity of our results, since simplification methods should be able to operate on any tree built by a TDIDT system.

[4]The Led domain is named Digit in Mingers' paper.

[5]Unfortunately, we do not know if the average size reported in Mingers' paper refers to trees build using the gain-ratio measure as in our experiments.

database	REP	MEP	CVP	0SE	1SE	PEP	CV-0SE	EBP	trained
Iris	5.689	6.222	5.866	5.778	7.645	5.332	5.6	5.065	5.598
Glass	*38.5*	38.19	36.87	38	*40.69*	35.31	*38.94*	35.88	35.38
Led	*28.15*	*28.24*	*27.99*	28.16	*29.52*	27.91	27.77	27.36	27.48
Hypo	**.515**	**.51**	.541	**.508**	.576	**.505**	**.496**	**.447**	.604
P. gene	23	24.37	25.62	24	25.62	23.38	23.5	21.75	23.5
Hepatitis	20.34	21.28	20.6	20.17	20.42	21.1	21.19	21.36	21.87
Cleveland	27.65	28.88	30.07	29.10	29.89	29.01	29.58	28.88	29.1
Hungary	**21.68**	**22.14**	27.41	**21.73**	**21.77**	**21.73**	**21.23**	**22.36**	25.5
Switzerland	**6.272**	12.65	**2.393**	**6.164**	**5.839**	**6.163**	**6.271**	**6.163**	13.3
Long Beach	**26.27**	**28.53**	**26.93**	**27.47**	**25.67**	**27.47**	**26.93**	**29.33**	33.73
Heart	23.54	23.67	24.13	23.81	24.29	**22.94**	23.35	**22.72**	23.82
Blocks	**3.22**	**3.34**	3.602	**3.19**	**3.35**	**2.98**	**3.17**	**3.05**	3.57

TABLE 35.4. Average error rates for different databases.

In Table 4 the average error rates concerning different simplification methods (*sophisticated* strategies) are reported together with the average error rate of unpruned trained trees (*naive* strategy). Both the sophisticated and the naive strategies have access to the same data, the training set, but the sophisticated one 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. From a quick look at the table we can immediately conclude that the sophisticated strategies do generally worse than the naive one in the first three databases, namely Iris, Glass and Led. More precisely, sophisticated methods that exploit an independent pruning set yield trees that are more accurate than corresponding grown trees, which indicates that pruning is beneficial. Unfortunately, the grown tree is less accurate than the trained tree, so that the global effect is negative. The only two cases in which the sophisticated strategy seems to win are those in which the trained tree is pruned by means of the PEP and EBP methods, which do not use independent pruning sets. On the contrary, for the Hypo, Switzerland, Hungary, and Long Beach domains, almost all pruning methods perform well. In particular, for the Switzerland data, the worst result obtained with a sophisticated strategy is better than that obtained with the naive approach. This is not surprising, since we had already observed that the OPGT (OPTT) is much smaller than the corresponding grown (trained) tree, which means that in this domain techniques for simplifying decision trees are inherently beneficial. Finally, the 1SE shows the worst performance in five databases, while the EBP has the lowest error rate in five domains with a significant improvement in two of them.

In order to study the statistical significance of these results, we compared the error rates of the pruned trees with those of the corresponding trained trees. Thus, values reported in bold (italics) denote a significant improvement (worsening) in predictive accuracy at the confidence level of 0.10. It is easy to see that *tree simplification does not generally decrease the predictive accuracy.* The only exception is represented by the application of the 1SE rule with both an independent pruning set and cross-validation sets. Furthermore, no method is able to significantly reduce the error rate in six domains (Iris, Glass, Led, P-gene, Hepatitis and Cleveland), while all methods perform well when applied to the Long Beach database, and all but one improve the accuracy of the final tree in the Hungary and Switzerland domain. It is interesting to observe that PEP and EBP produce

database	REP	MEP	CVP	0SE	1SE	OPGT	PEP	CV-0SE	EBP	OPTT
Iris	3.4	4	*5.4*	3.44	**3**	3.64	3.76	4.48	*4.88*	4
Glass	11.04	*18.52*	*27.92*	14.12	**7.04**	12.6	*21.12*	18.16	*28.72*	15.08
Led	20.52	*25.28*	*36*	*25.64*	**12.32**	21.44	**18.32**	*30.24*	*30.6*	22.04
Hypo	**8.2**	*15.76*	*14*	8.64	**7.2**	9.16	9.32	9.68	*13.68*	9.36
P-gene	**6.4**	*10*	*13.36*	7.24	**4.36**	7.84	**8.44**	9.52	*15.28*	10
Hepatitis	2.64	*8.36*	3.6	2.56	**1.36**	3.4	5.44	**2.64**	*9.08*	4.36
Cleveland	10.92	15	*30.76*	9.88	**4.12**	12.2	*19*	**10.36**	*29.28*	16.36
Hungary	**7.32**	10.44	*21*	**5.24**	**2.28**	10.16	10.2	9.2	*17.12*	9.64
Switzerland	1.4	*8.92*	1.68	1.56	1	1.32	1.16	1.16	1.08	1.16
Long Beach	6.16	*17.16*	7.28	6.92	**1.4**	4.12	4.96	3.8	*10.88*	4.92
Heart	33.52	31.92	*58.2*	27.52	**7.2**	36.16	**21.92**	**19.17**	46.04	45.96
Blocks	24.68	*65.04*	*78.48*	28.12	**13.12**	25.84	*37.24*	**17.56**	*50.92*	30.44

TABLE 35.5. Average tree size for different databases.

significantly better trees for the same data sets, so that it is possible to postulate, on empirical grounds, the equivalence of the two methods that appear to be different in their formulation. By summarizing these results, we can conclude that there is no indication that methods exploiting an independent pruning set perform definitely better than the others. This claim is at a variance with that reported by Mingers [Min89], and should be attributed to the different design of the experiments.

The average size (number of leaves) of the simplified trees for different domains is reported in Table 5. As already pointed out, data concerning the REP, MEP, CVP, 0SE and 1SE should be compared to the average size of corresponding OPGTs, while the results of PEP, CV-0SE, and EBP should be related to those of OPTTs. At a glance we can say that REP is biased towards overpruning, since in nine domains it produces trees smaller than the optimal ones on the average. This result has a theoretical justification: REP takes the decision of removing a branch by considering data in the pruning set alone, without any care of the evidence provided by the cases used to build the tree. Therefore, when relatively few cases are reserved for pruning, this simple solution will lead to overpruning.[6] Another method clearly biased towards overpruning is 1SE. This is not surprising, since the one standard error rule was introduced to choose the simplest tree whose estimated accuracy is comparable to that selected by the 0SE. On the contrary, MEP, EBP and CVP underprune in almost all cases, and even worse, CVP does not prune at all in two domains, namely Iris and Glass. The reason for this odd result seems to be related to the use of the gain-ratio as selection measure. Indeed, when all the leaves in a tree are "pure", that is their examples belong to the same class, the gain-ratio computed in the deepest internal nodes is equal to the maximum value 1.0. Thus, the critical value should be greater than or equal to 1.0 in order to prune the nodes at the bottom of the tree,[7] but in this case all internal nodes should be pruned according to the same criterion. This means that in the first step critical value pruning may find only two trees, T_{max} and the

[6] Actually, in a separate study on this method we have also observed that, in most of the domains considered in this paper, the optimal size of the pruning set is 70% of the training set.

[7] Experiments on the CVP are made by setting a maximum critical value equal to 1.0 and a step equal to 0.01.

root tree, with the obvious consequence that the former one will be generally chosen in the second step. Obviously, this happens only in the case of pure leaves, as with the Iris and Glass domains. In fact, in all the other domains in which there is at least an impure leaf because of clashes (i.e., examples of distinct classes but with same attribute values) or because of tree collapse,[8] this issue is less evident. Also for the Switzerland data there is no underpruning problem, but in this case the reason is that the best tree is often the root tree. Results on the CVP reveal another discrepancy with Mingers' data, at least for the two common domains, Iris and Led. Unfortunately, in this case we have no explanation, and we can only hypothesize the adoption of different stopping criteria in Mingers' TDIDT system. The differences for the other two methods, namely REP and MEP, can be justified by the fact that our algorithms are different from his. Indeed, as already pointed out, the REP method we implemented is guaranteed to find the optimal tree with respect to the pruning set, while this seems not to be true for Mingers' formulation. Moreover, our MEP version is the latter proposed by Cestnik and Bratko [CB91], and not the original one proposed by Niblett and Bratko [NB86].

As done for the error rate, even for the tree size we tested the significance of the differences by means of two-tailed paired t-tests. This time a number in bold indicates a significant overpruning at a level of 0.10, while a number in italics represents a significant underpruning. Recall that the comparison involves OPGTs for those methods that operate on the pruning set, and OPTTs for the others. The tests confirm that MEP, CVP and EBP are biased towards underpruning, while 1SE prefer overpruning. It is worthwhile to note that the predictive accuracy is not necessarily increased when a simplification method produces trees which do not significantly differ in size from the correspondingly optimally pruned trees: in fact, pruning may help to simplify trees without improving their predictive accuracy. Moreover, by measuring the tree size we do not have detailed enough information to guarantee that only overpruning or underpruning occurred. Thus, we can observe a significant increase of error rate with respect to the optimal tree even when the size appears to be optimal (see the results of REP in the Glass domain). Finally, by ideally superimposing Tables 4 and 5, it is also possible to draw 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 [Hol93]. It is also a confirmation of the overfitting problem that affects TDIDT systems.

35.5 Conclusions.

In this paper, a new empirical study of eight well-known simplification methods for decision-tree induction has been presented. The two main differences with previous works performed by Quinlan [Qui87] and Mingers [Min89] are the diverse design of the experiments and the use of optimally pruned trees as yardsticks for comparing the accuracy and complexity of simplified and unsimplified trees. The main conclusions drawn in our

[8] Another default stopping rule implemented in C4.5 prevents from growing a tree when the number of cases per outcome is less than two. However, we invoked the system by setting the parameter $-m$ to 1, so forcing the system to consider also the cases of n-ary tests with one example per outcome.

study are the following:

- Putting aside some data for the pruning process is not generally the best strategy.
- In general, simplification methods do not significantly decrease the predictive accuracy of the final trees.
- MEP, CVP and EBP are biased towards underpruning, while 1SE tends to overprune.

Further work is needed to confirm and extend these results. In particular, artificial data sets with either attribute or class noise may provide us with meaningful information on the performance of different simplification methods.

35.6 References

[BFOS84] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and regression trees*. Wadsworth International, Belmont, CA, 1984.

[BN92] W. Buntine and T. Niblett. A further comparison of splitting rules for decision-tree induction. *Machine Learning*, 8:75–85, 1992.

[CB91] B. Cestnik and I. Bratko. On estimating probabilities in tree pruning. In *Proceedings of the EWSL-91*, pages 138–150, 1991.

[EMS93] F. Esposito, D. Malerba, and G. Semeraro. Decision tree pruning as a search in the state space. In P. Brazdil, editor, *Machine Learning: ECML-93*. Springer-Verlag, Berlin, 1993.

[Fis92] D. H. Fisher. Pessimistic and optimistic induction. Department of Computer Science, Vanderbilt University, 1992.

[Hol93] R. C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11:63–90, 1993.

[MA94] P. M. Murphy and D. W. Aha. UCI repository of machine learning databases [machine-readable data repository]. Department of Information and Computer Science, University of California, Irvine, 1994.

[Min89] J. Mingers. An empirical comparison of selection measures for decision-tree induction. *Machine Learning*, 4:227–243, 1989.

[NB86] T. Niblett and I. Bratko. Learning decision rules in noisy domains. In *Proceedings of Expert Systems 86*, Cambridge, 1986. Cambridge University Press.

[Nib87] T. Niblett. Constructing decision trees in noisy domains. In I. Bratko and N. Lavrac, editors, *Progress in Machine Learning*. Sigma Press, Wilmslow, 1987.

[Qui87] J.R. Quinlan. Simplifying decision trees. *International Journal of Man-Machine Studies*, 27:221–234, 1987.

[Qui93] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, 1993.

[Sch93] C. Schaffer. Overfitting avoidance as bias. *Machine Learning*, 10:153–178, 1993.