Working Paper Series
ISSN 1170-487X

Reduced-error pruning with significance tests

by Eibe Frank and Ian H Witten

Working Paper 99/10
June 1999

Reduced-error pruning with significance tests

EIBE FRANK eibe@cs.waikato.ac.nz
IAN H. WITTEN ihw@cs.waikato.ac.nz
Department of Computer Science, University of Waikato, Hamilton, New Zealand

Abstract. When building classification models, it is common practice to prune them to counter spurious effects of the training data: this often improves performance and reduces model size. "Reduced-error pruning" is a fast pruning procedure for decision trees that is known to produce small and accurate trees. Apart from the data from which the tree is grown, it uses an independent "pruning" set, and pruning decisions are based on the model's error rate on this fresh data. Recently it has been observed that reduced-error pruning overfits the pruning data, producing unnecessarily large decision trees. This paper investigates whether standard statistical significance tests can be used to counter this phenomenon.

The problem of overfitting to the pruning set highlights the need for significance testing. We investigate two classes of test, "parametric" and "non-parametric." The standard chi-squared statistic can be used both in a parametric test and as the basis for a non-parametric permutation test. In both cases it is necessary to select the significance level at which pruning is applied. We show empirically that both versions of the chi-squared test perform equally well if their significance levels are adjusted appropriately. Using a collection of standard datasets, we show that significance testing improves on standard reduced error pruning if the significance level is tailored to the particular dataset at hand using cross-validation, yielding consistently smaller trees that perform at least as well and sometimes better.

1. Introduction

When building classification models, it is common practice to discard parts of the model that describe spurious effects in the training sample rather than true features of the underlying domain. This process is called "pruning." Not only does it often improve performance, it also produces simpler models that are easier for users to understand. For pruning, an effective mechanism is needed to distinguish parts of a classifier that are due to chance effects from parts that describe relevant structure.

Statistical significance tests are theoretically well-founded methods for determining whether an observed effect is a genuine feature of a domain or just due to random fluctuations in the sampling process. Thus they can be used to make pruning decisions in classification models. Reduced-error pruning (Quinlan, 1987), a standard algorithm for post-pruning decision trees, does not take statistical significance into account, but it is known to be one of the fastest pruning algorithms, producing trees that are both accurate and small (Esposito, Malerba & Semeraro, 1997). This paper investigates whether significance tests can be used to improve on this well-known pruning procedure.

The primary hypothesis is the following:

Hypothesis 1 *Statistical significance tests can be used to improve the reduced-error pruning algorithm to make decision trees smaller and more accurate.*

Significance tests can be divided into those called "parametric tests" that make some mathematical assumptions about the underlying distribution function and those called "non-parametric tests" (Good, 1994) that are essentially assumption-free. Tests based on the chi-squared distribution, for example, belong to the former group: they assume that the test statistic is distributed according to the chi-squared distribution. Their use is questionable for small sample sizes because then the assumptions required for applying the chi-squared distribution cease to be valid. Permutation tests, on the other hand, make no assumptions about the functional form of the underlying distribution, and belong to the second group of tests. Consequently, they can be applied with any sample size.

Graceful behaviour for small samples is particularly important in learning algorithms like decision tree inducers, where pruning decisions have to be made for smaller and smaller subsets of data. Given these considerations, it is plausible that, for a given amount of pruning—which can be controlled by the test's significance level—decision trees pruned using a permutation test will be more accurate than those pruned using a parametric test. This leads to our secondary hypothesis:

Hypothesis 2 *If decision tree A is the result of pruning using a permutation test, and decision tree B is the result of pruning using a parametric test, and both trees have the same size, then A will be more accurate than B on average.*

The structure of this paper is as follows. Section 2 explains why it is important to consider statistical significance when pruning decisions are made. Section 3 details standard statistical tests that can be employed. These tests are compared on artificial and practical datasets in Section 4 and 5 respectively. Section 6 shows how the significance level can be chosen to minimize tree size without loss in accuracy, and Section 7 discusses related work. Section 8 summarizes the findings of this paper.

2. Decision tree pruning and statistical significance

Figure 1 depicts an unpruned decision tree. We assume that a class label has been attached to each node of the tree—for example, by taking the majority class of the training instances reaching that particular node. In Figure 1 there are two classes: A and B.

The decision tree depicted in Figure 1 is a complete classifier and can be used to predict the class of a test instance by filtering it to the leaf node corresponding to the instance's attribute values and assigning the class label attached to that leaf. However, using an unpruned decision tree for classification potentially "overfits" the training data: some of its structure, namely that due to random variation in the particular data sample used for training, might not be warranted. Consequently, it is advisable before the tree is put to use, to ascertain which parts of the tree truly reflect effects present in the domain—and discard those that do not—before the tree is put to use. This process is called "pruning."

A general, fast, and easily applicable pruning method is "reduced-error pruning" (Quinlan, 1987). The idea is to hold out some of the available instances—the

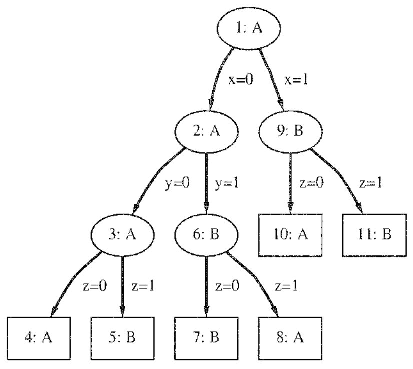


Figure 1. A decision tree with two classes A and B

x	y	z	class
0	0	1	A
0	1	1	B
1	1	0	B
1	0	0	B
1	1	1	A

Figure 2. An example pruning set

"pruning set"—when the decision tree is built, and prune the tree until the classification error on these independent instances starts to increase. Because the instances in the pruning set are not used for building the decision tree, they provide a less biased estimate of its classification error on future instances than the training data. Reduced-error pruning uses this estimate as a guideline for its operation.

Figure 2 shows an example pruning set for the decision tree from Figure 1. Figure 3 displays how reduced-error pruning proceeds for this example. In each tree, the classification errors committed by the individual nodes are given in brackets. A pruning operation involves replacing a subtree by a leaf. Reduced-error pruning will perform this operation if it does not increase the total number of classification errors. Traversing the tree in a bottom-up fashion ensures that the result is the smallest pruned tree that has minimum error on the pruning data (Esposito, Malerba & Semeraro, 1995). This traversal strategy is a direct result of the condition that a node can only be converted to a leaf if all subtrees attached to it have already been considered for pruning.

Assuming that the tree is traversed left-to-right, the pruning procedure will first consider for removal the subtree attached to node 3 (Figure 3a). Because the subtree's error on the pruning data (1 error) exceeds the error of node 3 itself (0 errors), node 3 will be converted to a leaf (Figure 3b). Subsequently, node 6 will be replaced by a leaf for the same reason (Figure 3c). Having processed both of its successors, the pruning procedure will then consider node 2 for deletion. However, because the subtree attached to node 2 makes fewer mistakes (0 errors) than node

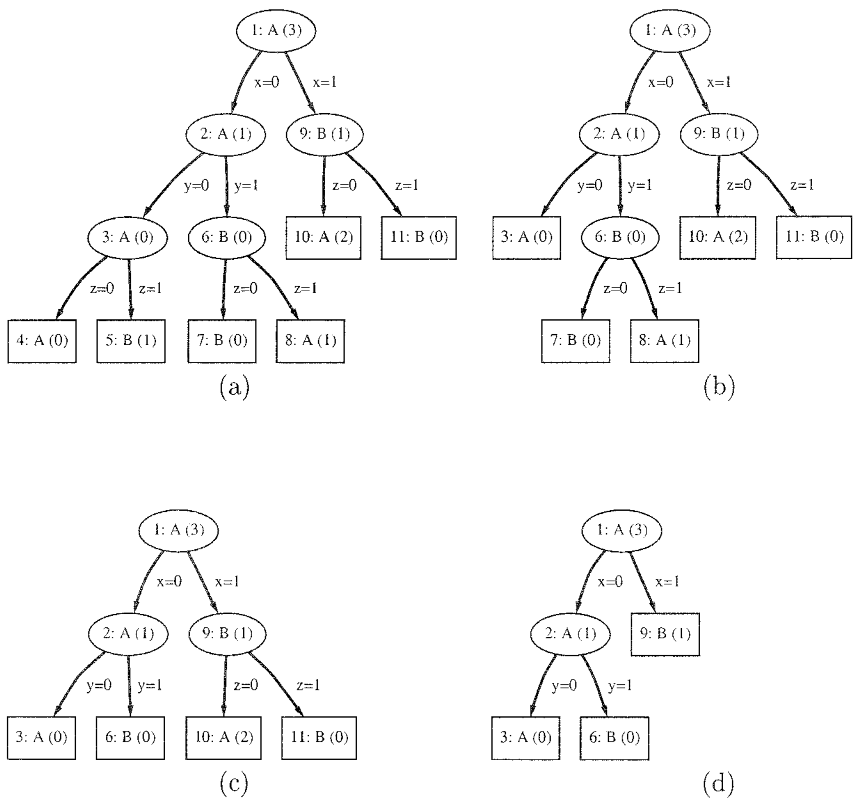


Figure 3. Reduced-error pruning example

2 itself (1 error), the tree will remain in place. Next, the subtree extending from node 9 will be considered for pruning, resulting in a leaf (Figure 3d). In the last step, node 1 will be considered for pruning, leaving the tree unchanged.

Unfortunately, there is a problem with this simple and elegant pruning procedure: it overfits the pruning data (Oates & Jensen, 1998). The consequence is the same as for overfitting the training data, namely an overly complex decision tree. A simple example shows why this happens.

Consider a dataset with 10 random binary attributes with uniformly distributed values "0" and "1." Let the class also be binary with an equal number of instances of each class, and class labels "A" and "B." Of course, the expected classification error for this domain is the same for every possible classifier, namely 50%, and the simplest conceivable decision tree for this problem—predicting the majority class from the training data—consists of a single leaf node. We would like to find this trivial tree because then we could correctly deduce that none of the attributes in this dataset provides any information about the class label associated with a particular instance.

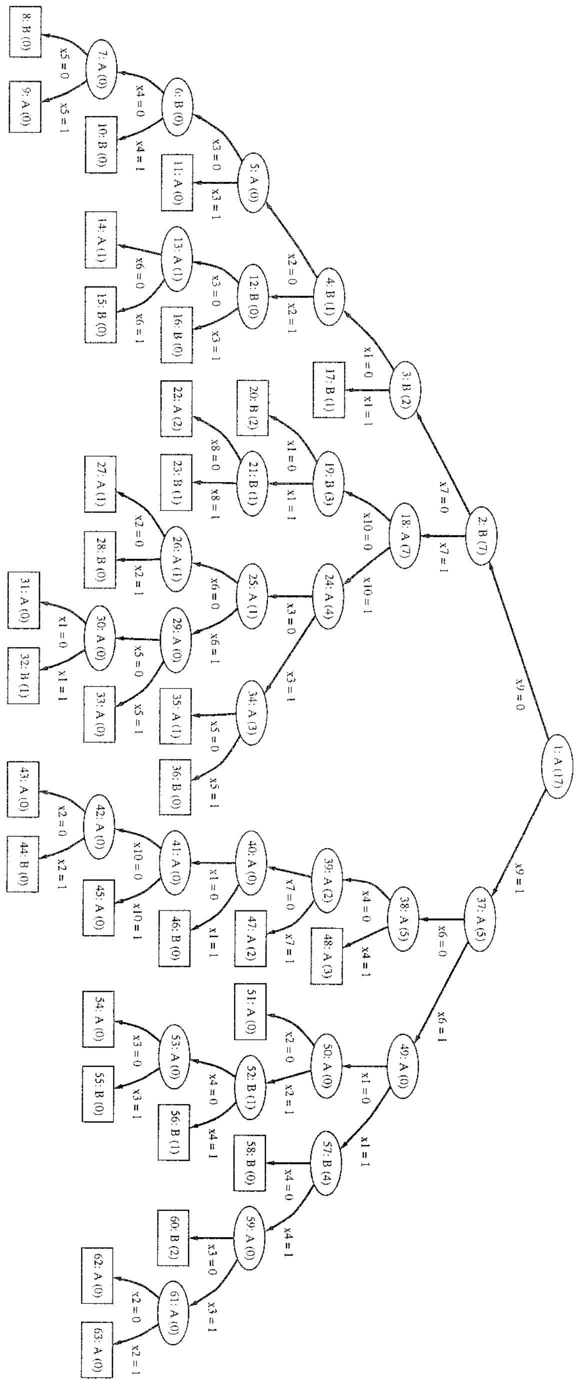


Figure 4. Tree for no-information dataset before pruning

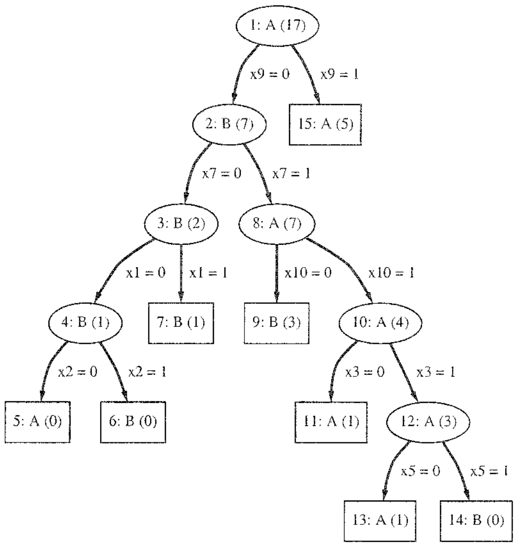


Figure 5. Tree for no-information dataset after pruning

Figure 4 shows a decision tree, before pruning, for a randomly generated sample of 100 instances. Figure 5 shows the same tree after pruning has taken place. Reduced-error pruning was employed, using one third of the instances as the pruning set (Oates & Jensen, 1999), and the standard information gain criterion (Quinlan, 1986) for selecting the tests at each node.

Figure 5 suggests that, although reduced-error pruning successfully reduces the size of the unpruned tree, it certainly does not generate the correct decision tree. This hypothesis can easily be confirmed by repeating the experiment with different randomly generated datasets (Jensen & Schmill, 1997). Figure 6 summarizes the results obtained by repeating it 100 times for each of 10 different training set sizes. The error bars are 95% confidence intervals for the mean of the decision trees' size; they show that reduced-error pruning indeed generates overly complex decision trees for this problem—independent of the particular training set.

A closer look at Figure 4 reveals the reason for this pathology. Because of the large number of subtrees that have to be considered for pruning, there are always some that fit the pruning data well *just by chance*. The pruning procedure incorrectly retains those trees. This also explains why the size of the pruned tree increases with the number of training instances: the larger the unpruned tree, the more subtrees are likely to fit the pruning data well by chance. In other words, this problem arises because reduced-error pruning does not take into account the fact that the pruning data results from a stochastic sampling process. The distribution of class values at the nodes of a decision tree does not necessarily reflect the true distribution, and this effect is particularly pronounced if the data samples at the nodes are small. Stated differently, the pruning procedure does not test whether

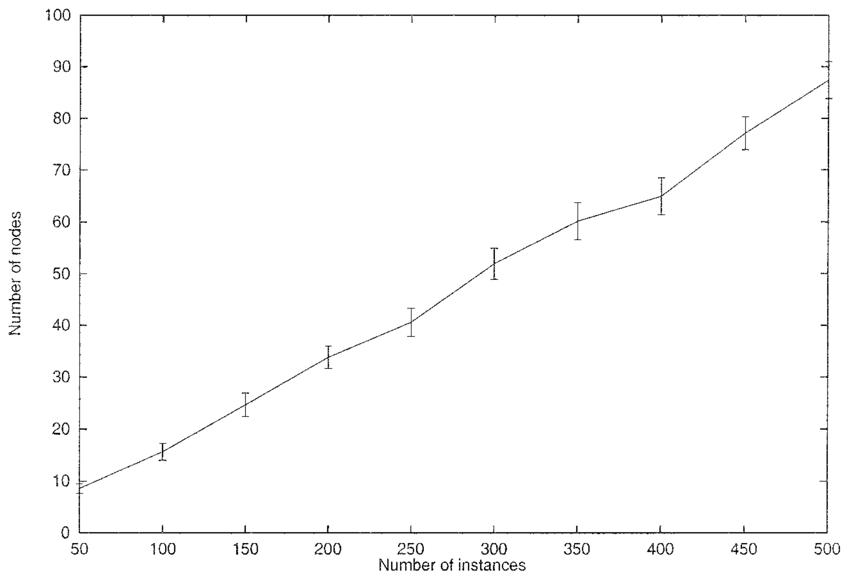


Figure 6. Size of pruned tree relative to training set size, using reduced-error pruning

the association between the predictions and the observed class values in the pruning data is statistically significant.

Statistical significance tests—more specifically, significance tests on contingency tables—suggest themselves as an obvious remedy: a subtree is only retained if there is a significant association between its predictions and the class labels in the pruning data. The following sections discuss these tests in detail, and compare their performance in pruning decision trees derived from artificial and practical datasets.

3. Significance tests on contingency tables

Tests for independence in a contingency table determine whether if there is a statistically significant dependence between the values of two nominal variables. In the pruning problem above, the two variables are (a) the actual class values in the pruning data, and (b) the class values predicted by the subtree. We want to know whether there really is a significant dependence between the true class values and the predicted ones, or whether it is likely that the observed correlation is just due to chance—that is, caused by random fluctuations in the particular sample of pruning data being employed.

Figure 7 shows the structure of a contingency table. The I columns and J rows correspond to the values of the two nominal variables being considered. Each cell of the table contains the number n_{ij} of times the corresponding combination of values has been observed in N instances. The column and row totals N_{i+} and N_{+j} are the sums of the entries in each row and column respectively.

n_{11}	n_{21}	...	n_{I1}	N_{+1}
n_{12}	n_{22}	...	n_{I2}	N_{+2}
...	...	...	...	...
n_{1J}	n_{2J}	...	n_{IJ}	N_{+J}
N_{1+}	N_{2+}	...	N_{I+}	N

Figure 7. Structure of a contingency table

		actual class values A	actual class values B	
predicted class values	A	1	0	1
	B	1	3	4
		2	3	5

Figure 8. Example confusion matrix

In the evaluation of classification algorithms, a contingency table comparing the algorithm's predictions to the actual class values is known as a "confusion matrix." Figure 8 shows the confusion matrix for the final pruned tree from Figure 3. The column totals represent the number of pruning instances of each class that reach the corresponding node—in this case, node 1. The row totals, also in boldface, correspond to the number of pruning instances that would be assigned to each class if the corresponding subtree—in this case, the full tree—were used for classification. A confusion matrix makes it easy to see how many pruning instances would be correctly classified by a subtree: the number of correctly classified instances is the sum of the diagonal elements in the matrix. Confusion matrices, as a particular type of contingency table, are the basis for the significance tests considered in this paper.

3.1. Testing significance

The hypothesis that two variables, such as the actual and predicted class values, are independent is called the "null hypothesis," and a significance test determines whether there is enough evidence to reject this hypothesis. When pruning decision trees, rejecting the null hypothesis corresponds to retaining a subtree instead of pruning it. The degree of association between the two variables is measured by the "test statistic." This can, for example, be the classification error. The significance test computes the probability that the same or a more extreme degree of association will occur by chance if the null hypothesis is correct. This quantity is called the "p-value" of the test. If the p-value is low, the null hypothesis can be rejected, that is, the observed degree of association is unlikely to be due to chance. Usually, this is done by comparing the p-value to a fixed significance level α, rejecting the null hypothesis if α is greater or equal to the p-value. A significance test can be applied to the pruning problem by computing the p-value for the observed association and comparing it to α, retaining or discarding the subtree accordingly.

Two quantities are important when evaluating a significance test: the probabilities of committing a "Type I error" and "Type II error." A Type I error occurs when the null hypothesis is incorrectly rejected, in other words, when a significant association is found although a real association does not exist. The probability of committing a Type I error is the significance level α, and usually fixed *a priori.* In the context of pruning, committing a Type I error corresponds to incorrectly deciding not to prune—also called "underpruning." A Type II error, on the other hand, occurs when the null hypothesis is incorrectly accepted, in other words, when a significant association is overlooked. Committing a Type II error corresponds to "overpruning"—deciding to discard a subtree although it would be better if it were kept.

The complement of the probability of committing a Type II error is called the "power" of the significance test. Ideally, neither type of error would occur. In that case, the significance level α would be zero, and the power would be one. Unfortunately, this is not feasible in practice, and there is a trade-off between the two flavors of error: the less likely a Type I error becomes, the more likely is a Type II one, and vice versa. Consequently one type of error can not be considered in isolation when comparing statistical tests, and one can only seek a test that maximizes the power for a given significance level α.

Statistical tests are based on the distribution of the test statistic under the null hypothesis. As mentioned above, they can be divided into two groups: parametric tests, which rely on the assumption that the distribution belongs to a particular class of parametric functions, and non-parametric tests, which do not require the distribution function to be of any particular form. The remainder of this section first discusses parametric tests based on the chi-squared distribution, and then a group of non-parametric tests known as "permutation tests."

3.2. Parametric tests

The most popular tests for independence in contingency tables are based on the fact that some test statistics have approximately a chi-squared distribution with $(r-1)(c-1)$ degrees of freedom if the null hypothesis is correct. The classic test statistic with this property is the chi-squared statistic (Agresti, 1990)

$$\chi^2 = \sum_i \sum_j \frac{n_{ij} - e_{ij}}{e_{ij}}, \tag{1}$$

where e_{ij} are the expected cell counts under the null hypothesis, calculated according to

$$e_{ij} = N\hat{p}_i\hat{p}_j = N\frac{N_{i+}}{N}\frac{N_{+j}}{N} = \frac{N_{i+}N_{+j}}{N}, \tag{2}$$

where $\hat{p}_i$ is the estimated probability that a particular observation will fall into column i, and $\hat{p}_j$ is the corresponding probability for row j. Because these two probabilities are independent under the null hypothesis, their product constitutes the probability that an observation will fall into cell (i, j).

An alternative to the chi-squared statistic, which also has a chi-squared distribution, is the "log likelihood ratio" (Agresti, 1990)

$$G^2 = 2 \sum_i \sum_j n_{ij} \log (n_{ij}/e_{ij}). \tag{3}$$

A disadvantage of tests based on the chi-squared distribution is that they are statistically invalid when the sample size is small (Agresti, 1990). The chi-squared distribution is an approximation to the test statistics' true sampling distribution under the null hypothesis, and this approximation is only accurate when the sample is large. Unfortunately, there is no single rule that can be used to determine when the approximation is valid (Agresti, 1990, page 247). Cochran (1954), for example, suggests that a test based on the χ^2 statistic can be employed if none of the expected cell counts is smaller than 1, and at most 20% of them have expected values below 20. Agresti (1990, page 247) writes that "...the chi-squared approximation tends to be poor for sparse tables containing both small and moderately large expected frequencies." However, it has also been shown that χ^2 works well with smaller sample sizes and more sparse tables than G^2 (Agresti, 1990, page 246). Depending on the expected cell counts, using the chi-squared distribution in conjunction with G^2 can result in a test that is either too conservative or too liberal (Agresti, 1990, page 247). A test that is too conservative produces p-values that are too large, while one that is too liberal produces p-values that are too small.

3.3. Non-parametric tests

Non-parametric tests have the advantage that they do not make assumptions about the specific functional form of the test statistic's distribution. Permutation tests are a class of non-parametric tests that compute the test statistic's distribution under the null hypothesis explicitly, by enumerating all possible permutations of the given data. In contrast to parametric tests using the chi-squared distribution, permutation tests are statistically valid in small-sample situations (Good, 1994). They are based on the fact that, under the null hypothesis, all possible permutations of a dataset are equally likely to occur. The p-value of a permutation test is the fraction of these permutations for which the test statistic has an equally or more extreme value than for the original data (Good, 1994).

In the case of classification problems, permutations of a dataset correspond to permutations of the class labels associated with its instances. Figure 9 shows all possible permutations for a small dataset with four instances, along with the classification error that they incur when classified as indicated. Note that there are permutations that produce identical columns in Figure 9. Consider, for example, the permutation that just swaps the class labels of the two instances belonging to class A: it produces the same column as the original data. Like the original data, four of the 24 permutations result in zero error. If a permutation test were per-

Predicted	Data	Permutations																							
A	A	A	A	A	A	A	A	B	B	B	B	B	B	B	B	B	B	B	B	A	A	A	A	A	A
B	B	B	B	B	B	A	A	A	A	B	B	A	A	A	A	B	B	A	A	A	A	B	B	B	B
B	B	B	A	B	A	B	B	B	A	A	A	A	B	B	A	A	A	A	B	B	B	A	B	A	B
A	A	A	B	A	B	B	B	A	B	A	A	B	A	A	B	A	A	B	A	B	B	B	A	B	A
Errors	0	0	2	0	2	2	2	2	4	2	2	4	2	2	4	2	2	4	2	2	2	2	0	2	0

Figure 9. Permutations of a datasets

$$
\begin{array}{cc|c|c|c}
 & & \multicolumn{2}{c}{\text{actual}} & \\
 & & \text{A} & \text{B} & \\
\text{pre-} & \text{A} & 2 & 0 & 2 \\
\text{dicted} & \text{B} & 0 & 2 & 2 \\
 & & 2 & 2 & 4
\end{array}
\qquad
\begin{array}{cc|c|c|c}
 & & \multicolumn{2}{c}{\text{actual}} & \\
 & & \text{A} & \text{B} & \\
\text{pre-} & \text{A} & 1 & 1 & 2 \\
\text{dicted} & \text{B} & 1 & 1 & 2 \\
 & & 2 & 2 & 4
\end{array}
\qquad
\begin{array}{cc|c|c|c}
 & & \multicolumn{2}{c}{\text{actual}} & \\
 & & \text{A} & \text{B} & \\
\text{pre-} & \text{A} & 0 & 2 & 2 \\
\text{dicted} & \text{B} & 2 & 0 & 2 \\
 & & 2 & 2 & 4
\end{array}
$$

Figure 10. Example contingency tables

formed on this dataset using the classification error as the test statistic, the p-value would be 4/24, or 1/6.

Each permutation of class labels can also be written in form of a contingency table, with some of the permutations mapping to the same table. The 24 permutations from Figure 9 result in only three tables, depicted in Figure 10. The first table corresponds to all permutations with 0 errors, the second one to all permutations with 2 errors, and the third one to all permutations with 4 errors. These three tables have one thing in common: they all share the same marginal totals. Permuting the class labels does not alter the number of instances that belong to each class. It also does not alter the number of instances assigned to each class by the classifier. In statistical terms, permutation tests on contingency tables derive a p-value by conditioning on the given marginal totals.

Identical contingency tables result in the same value for the test statistic. Thus the p-value of a permutation test can also be computed by summing up the probabilities of all contingency tables with an equally or more extreme value for the test statistic. The probability of a contingency table p_f—equivalent to the fraction of random permutations resulting in the table—can be written in closed form

$$p_f = \frac{\prod_i n_{i+}! \prod_j n_{+j}!}{n! \prod_i \prod_j n_{ij}!}. \tag{4}$$

This function is known as multiple hypergeometric distribution (Agresti, 1990). If s_f is the test statistic's value for the contingency table f, and s_o its value for the original data, then the p-value can be written as

$$p = \sum I(s_f \leq s_o) p_f, \tag{5}$$

where $I()$ is the indicator function, and the sum is over all contingency tables with the same marginal totals.

Unfortunately, both methods of computing the exact p-value of a permutation test—enumerating all possible permutations directly, or just enumerating all possible contingency tables—are computationally infeasible for all but very small sample sizes. For some test statistics sophisticated "network algorithms" have been developed that only evaluate a small subset of all possible contingency tables in order to compute the exact p-value (Good, 1994). They make use of mathematical properties of the test statistic in order to cut down the search space. However, even these sophisticated algorithms are only applicable if the sample size is small because they are still computationally very expensive.

3.4. Approximation

The solution to this dilemma is to realize that the "exact" p-value is not required when performing the test. It is sufficient to approximate it to a degree that makes it clear beyond reasonable doubt whether it is greater than the significance level α or not. This can be done by randomly sampling permutations from the space of all possible permutations, and computing the proportion $\hat{p}$ of these for which the test statistic has an equal or smaller value than for the original data (Good, 1994). The proportion $\hat{p}$ constitutes an approximation to the exact p-value, and the precision of this approximation increases with the number of random samples that are generated. Alternatively, one could sample from the space of all possible contingency tables and use Equation 5 to approximate the exact p-value. This is advantageous if extra speed is important, because there are efficient algorithms for generating random contingency tables with a given set of marginal totals (Patefield, 1981).

It remains to determine how many random samples are needed to give an approximation that is accurate enough. Statisticians have designed a procedure for this purpose, called the "sequential probability ratio test" (Lock, 1991). Figure 11 summarizes the decision rules for this test, where n is the number of random samples. It employs three constants that determine how closely the approximation emulates the exact test: p_a, p_o, and A, where $0 < p_a < \alpha < p_o < 1$, and $0 < A < 1$. Recommended values are $p_a = 0.8 * \alpha$, $p_b = 1.2 * \alpha$, and $A = 0.1$ (Lock, 1991).

In principle, any test statistic can be employed in conjunction with a permutation test, and this is one of its major advantages. When testing for independence in a contingency table, the test statistic should measure the degree of association between two nominal variables. Depending on the particular type of association being investigated, different test statistics sometimes lead to different results.

3.5. Test statistics

Two possible statistics have already been discussed in the context of parametric tests: χ^2 and G^2. Both can also be used to form permutation tests (Good, 1994). The significance level is simply the fraction of random permutations for which the statistic's value is at least the same as for the original data—because both statistics increase monotonically with the degree of association that is present. The

$$\begin{array}{rcl}
\text{If } \hat{p} * n \geq c * n + \dfrac{\log A}{\log K} & \rightarrow & \text{accept null hypothesis} \\
\text{If } \hat{p} * n \leq c * n - \dfrac{\log A}{\log K} & \rightarrow & \text{reject null hypothesis} \\
\text{Otherwise} & \rightarrow & \text{continue sampling}
\end{array}$$

$$c = \log\left(\frac{1-p_o}{1-p_a}\right) / \log K \qquad K = \frac{p_a(1-p_o)}{p_o(1-p_a)}$$

Figure 11. Sequential probability ratio test

		actual A	actual B	
predicted	A	2	0	2
	B	0	2	2
		2	2	4

$p - value = 2/6$

		actual A	actual B	
predicted	A	1	1	2
	B	1	1	2
		2	2	4

$p - value = 1$

		actual A	actual B	
predicted	A	0	2	2
	B	2	0	2
		2	2	4

$p - value = 2/6$

Figure 12. Three tables and their p-values

chi-squared distribution, which is the basis for the parametric tests discussed earlier, is in fact only an approximation to the permutation distribution of the two statistics, and, as mentioned above, this approximation is unreliable for small sample sizes (Agresti, 1990). Note that, although the sequential probability ratio test is only an approximation to the exact test, it is guaranteed to closely approximate the true p-value, whereas this is not the case for tests based on the chi-squared distribution.

Another potential test statistic has also already featured above, although it did not play the role of a test statistic. The probability p_f of a contingency table under the null hypothesis—given by the multiple hypergeometric distribution—is an alternative to χ^2 or G^2 (Good, 1994). The idea is that a rare contingency table, having a low value for p_f, indicates a strong association between the two variables involved. The test's significance level is the fraction of random permutations for which p_f is no greater than for the original data—because the greater the association, the smaller the probability. When both variables in the contingency table are binary, this permutation test is known as the two-sided version of Fisher's exact test (Agresti, 1990). In the general case it is sometimes called the Freeman and Halton test (Good, 1994).

All permutation tests share the disadvantage that the distribution of p-values is very sparse[1] when the sample size is extremely small (Agresti, 1990). This is due to the small number of contingency tables that are possible when there are very few instances.

3.6. Sparseness

The problem of sparseness is illustrated in Figure 12. It shows the three possible contingency tables with two instances in each row and column, and the p-values of Fisher's exact test for each one: 2/6, 1, and 2/6 respectively. It can be shown that the large gaps between the individual p-values have the consequence that the actual probability of committing a Type I error can be much lower than the significance level α (Agresti, 1990). This causes the test to become overly conservative.

There is a solution to this problem known as "randomization on the boundary" (Agresti, 1990). Let p_1 be the p-value of the contingency table under investigation, and p_2 be the next smaller p-value of a table with the same marginal totals (or zero if p_1 is the smallest p-value possible). Moreover, let α be equally large or larger than p_2. (Otherwise, the null hypothesis will be accepted.) Randomization on the boundary means that the null hypothesis will be rejected with probability $(\alpha - p_2) \,/\, (p_1 - p_2)$ even if α is smaller than p_1—which would normally mean that it will be accepted. In the example from Figure 12, randomization on the boundary will reject the null hypothesis for the leftmost table at a significance level of $\alpha = 1/10$ with probability 3/10; in other words, it will reject the null hypothesis in 30% of the cases that would normally cause it to be accepted. It can be shown that Fisher's exact test is uniformly most powerful among all unbiased tests for comparing binomial populations, if randomization on the boundary is performed (Agresti, 1990).

An approximation to this randomization procedure is the "mid-p value" method (Lancaster, 1961). Here, the mid-point half-way between p_1 and p_2, given by $(p_1 + p_2)/2$ is used instead of p_1. The mid-p values for the three tables from Figure 12 are 1/6, 2/3, and 1/6 respectively. The mid-p value has the advantage that it is more uniformly distributed under the null hypothesis than the ordinary p-value, and its expected value is 0.5—standard properties of significance tests with *continuous* p-values. Statisticians recommend the mid-point procedure to avoid problems arising from sparseness, and argue that it is a good compromise between a conservative test and performing randomization on the boundary (Agresti, 1990).

4. Experiments on artificial data

The overall qualities of different pruning methods can best be judged by looking at two extremes in the space of potential learning scenarios: on the one hand, a situation where all pruning is beneficial, and on the other, a situation where any kind of pruning is harmful. Focusing on these two cases, this section compares the performance of the tests from the previous section using artificially generated data.

Section 2 has already introduced an artificial problem that requires a maximum amount of pruning. In this problem, the class is completely independent of the predictor attributes, and a single root node is sufficient to achieve optimum predictive performance. As discussed in Section 2, reduced-error pruning fails to identify the null model as the correct model for this learning problem. How do the significance tests from the previous section fare?

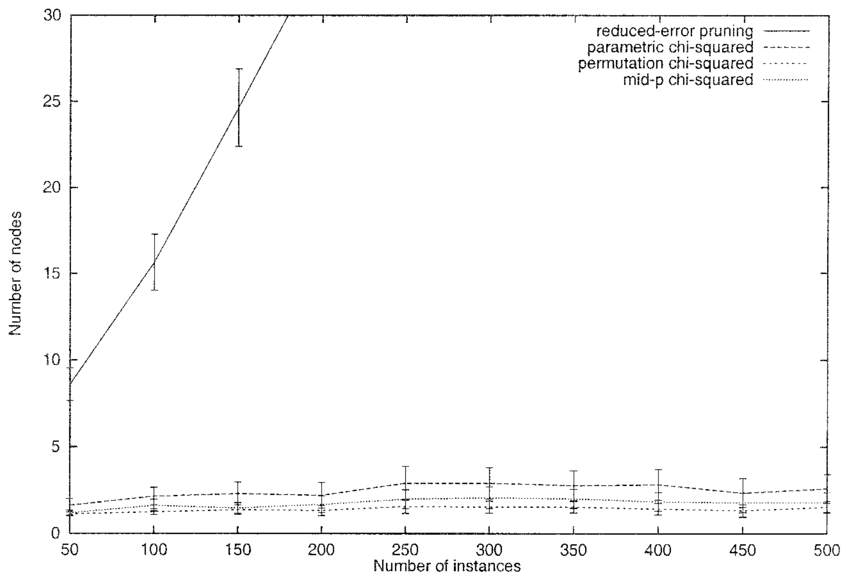


Figure 13. Performance of tests for no-information dataset

Figure 13 shows how successful they are in identifying and eliminating subtrees that are retained incorrectly by reduced-error pruning. Each time reduced-error pruning decides to retain a subtree, this decision is verified using a significance test. As in Figure 6, 100 different random datasets were generated to obtain one data point in the graph, and the error bars are 95% confidence intervals on the mean. The figure shows graphs for the ordinary parametric chi-squared test and the permutation test based on the chi-squared statistic. For the chi-squared permutation test, it also shows the results obtained using the mid-p adjustment. The graphs for the Freeman and Halton test are almost identical to those for the chi-squared permutation test, and therefore omitted. For all permutation tests involved, the sequential probability ratio test was used to determine how many permutations had to be investigated before the null hypothesis could be accepted or rejected. In all tests—the parametric tests as well as the non-parametric ones—a significance level α of 0.1 was used.

The results in Figure 13 show that significance testing successfully reduces the number of subtrees that are incorrectly retained by the pruning procedure. Compared to Figure 6 from Section 2, the average number of nodes in the pruned decision trees is dramatically reduced. However, there is some variation in the degree to which the different significance tests achieve this reduction. The most liberal of the tests is the parametric chi-squared test: it consistently produces the trees with the highest average number of nodes—that is, more often than the other tests, it incorrectly rejects the null hypothesis. Considering the tests in order of their performance, the next best is the mid-p test based on the chi-squared statis-

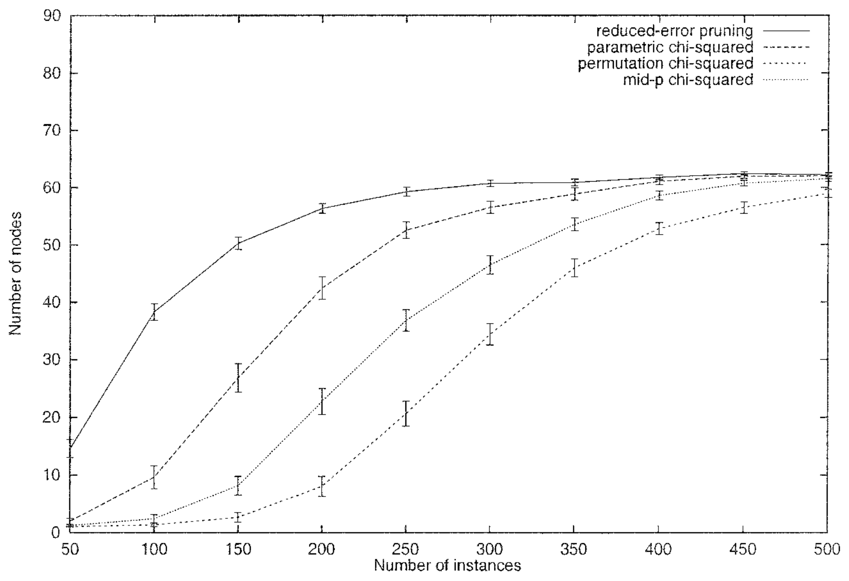


Figure 14. Performance of tests for parity dataset

tic. As expected from the discussion in the previous section, using the mid-p value results in a test that is more liberal than the ordinary permutation test—the best-performing test in this particular task. Considering the definition of the problem, this finding comes as no surprise, because the dataset is devoid of any structure whatsoever; hence no test can be too conservative, and the "best" test is one that always accepts the null hypothesis. Clearly, this is not useful in practice: the question is how sensitive is the test when structure is present.

In order to investigate this question, we move to the other end of the problem space, and consider a dataset where the best way of pruning is not to prune at all: a dataset with a binary class, whose value is 1 if and only if the number of ones in the (binary) attributes is odd. This is known as the "parity problem," and the correct tree contains exactly one leaf for each possible attribute combination. Again, instances are generated by randomly sampling attribute values from the uniform distribution, followed by assigning the appropriate class value. The pruned tree's expected classification accuracy on fresh data is proportional to the number of leaves in the tree. Since the total number of nodes in a (binary) tree, which includes the leaves, is twice the number of leaves plus one, the expected accuracy is also proportional to the number of nodes.

Figure 14 shows the results for the different pruning strategies, derived in the same way as for Figure 13. Again, the Freeman and Halton test is omitted because its results are almost identical to those for the permutation test based on the chi-squared statistic. The performance of the tests is as expected from the previous scenario: again, the parametric chi-squared test is the most liberal, in this case

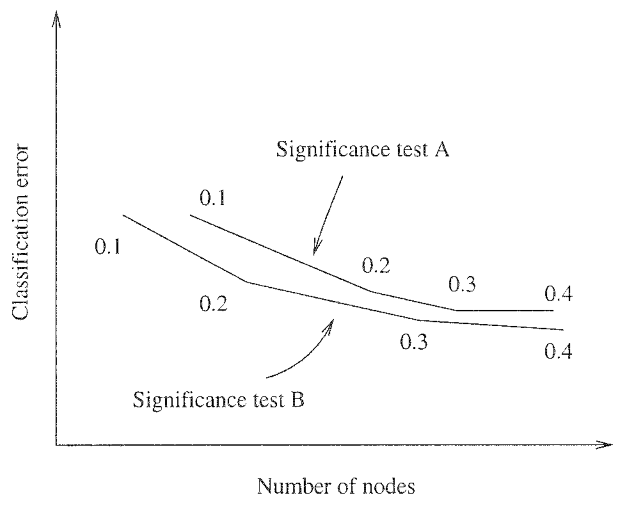


Figure 15. Significance test A uniformly dominates test B

producing the best result, followed by the mid-p permutation test, which gives the second-best result, and the ordinary permutation test based on the chi-squared statistic, which performs worst.

The results for these two artificial scenarios show that, for a given significance level, the different significance tests can be ordered according to the amount of pruning that they incur. It remains the question whether this difference in behaviour can be eliminated by adjusting the significance level for each test according to the properties of the domain. It is, for example, possible that the parametric chi-squared test at a significance level s_1 results in the same amount of pruning as the chi-squared permutation test at a significance level s_2—if s_1 is set to some appropriate value smaller than s_2.

This hypothesis can be tested by plotting the classification error and tree size for each significance level as depicted in Figure 15. In this hypothetical situation, there are two tests A and B, and four significance levels: 0.1, 0.2, 0.3, and 0.4. Assuming that the tests are sufficiently well-behaved, performance at intermediate levels can be interpolated by connecting the data points for these four significance levels with straight lines.

In this contrived example, it is clear that test A really is fundamentally different from test B: for all potential tree sizes, A produces more accurate results than B. In other words, there is no reason to use test B because A always performs better if the significance level is chosen appropriately. However, it is unlikely that the situation is so clear-cut for practical datasets.

Table 1. Datasets used for the experiments

Dataset	Instances	Missing values (%)	Numeric attributes	Nominal attributes	Classes
anneal	898	0.0	6	32	5
audiology	226	2.0	0	69	24
australian	690	0.6	6	9	2
autos	205	1.1	15	10	6
balance-scale	625	0.0	4	0	3
breast-cancer	286	0.3	0	9	2
breast-w	699	0.3	9	0	2
german	1000	0.0	7	13	2
glass (G2)	163	0.0	9	0	2
glass	214	0.0	9	0	6
heart-c	303	0.2	6	7	2
heart-h	294	20.4	6	7	2
heart-statlog	270	0.0	13	0	2
hepatitis	155	5.6	6	13	2
horse-colic	368	23.8	7	15	2
ionosphere	351	0.0	34	0	2
iris	150	0.0	4	0	3
labor	57	3.9	8	8	2
lymphography	148	0.0	3	15	4
pima-indians	768	0.0	8	0	2
primary-tumor	339	3.9	0	17	21
sonar	208	0.0	60	0	2
soybean	683	9.8	0	35	19
vehicle	846	0.0	18	0	4
vote	435	5.6	0	16	2
vowel	990	0.0	10	3	11
zoo	101	0.0	1	15	7

5. Experiments on practical datasets

Experiments on datasets from the real world are the only way to evaluate whether a particular method is likely to perform better than other methods in practice. The UCI repository of machine learning datasets is a popular source of suitable benchmark learning problems (Blake, Keogh & Merz, 1998). This section presents experimental results for 27 of these datasets in order to compare the performance of the statistical tests in a more realistic setting. The datasets are the 27 smallest ones used by Frank and Witten (1998a). They are listed in Table 1.

For each dataset, a fully grown tree is post-pruned using reduced-error pruning in conjunction with a significance test. Each time reduced-error pruning decides to retain a subtree, this is verified using the significance test and the subtree is retained or discarded accordingly. One third of the instances are used as the pruning set (Oates & Jensen, 1999), and the standard information gain criterion (Quinlan, 1986) is employed for selecting the tests at each node. Missing attribute values

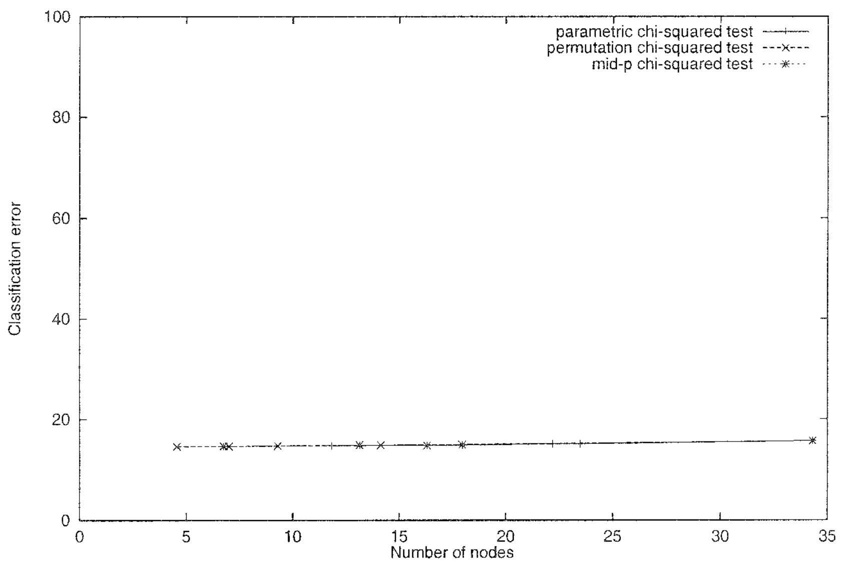


Figure 16. Comparison of significance tests for australian dataset

are dealt with in the simplest possible way by assigning them to the most popular branch.

As mentioned above, it is not enough to compare the performance of a set of significance tests at a particular, fixed significance level α, because it is possible that any performance difference vanishes when α is adjusted appropriately for each test. Diagrams like the one in Figure 15 are the only reliable way to detect whether a particular test consistently makes "better" pruning decisions than other tests.

Figures 16, 17 and 18 contain diagrams for three of the 27 datasets. The remaining 24 diagrams can be found in Appendix A. These three examples have been chosen because they represent typical cases. Each diagram contains results for three different significance tests: the parametric chi-squared test, the chi-squared permutation test, and the mid-p version of the latter one. The data points for each significance level represent estimates from ten-fold cross-validation repeated ten times. As in Figure 15, they are connected by straight lines in increasing order. The graphs also contain results for standard reduced error pruning, which corresponds to applying the significance tests with a significance level of 1. This is the rightmost data point in each of the diagrams, and it is shared by all three of the curves representing the different significant tests. For each test, the data points corresponding to the four significance levels are ordered left-to-right because they lead to increasingly larger decision trees (as in Figure 15.)

The graphs present strong evidence that, in practice, the fundamental behaviour of the three tests is almost identical. For every dataset, overlaying their curves produces a very smooth result—despite the fact that the individual curves overlap

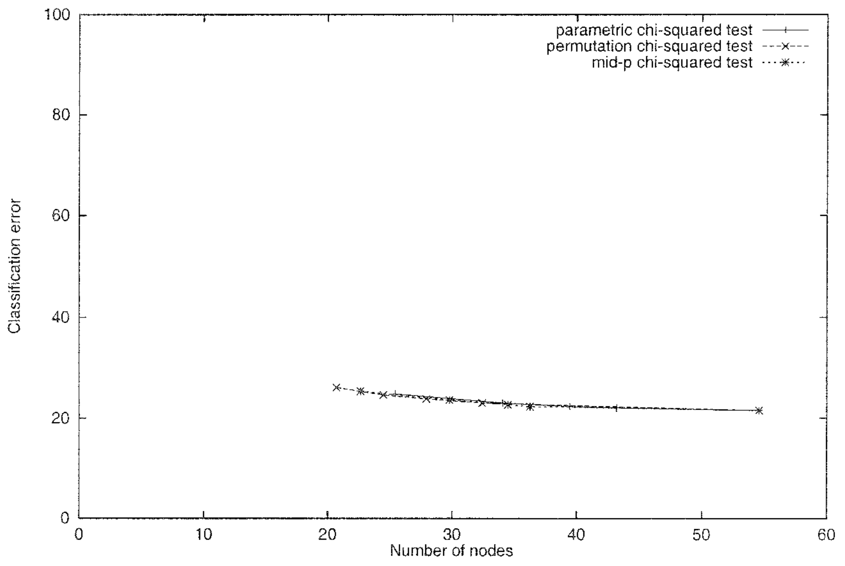


Figure 17. Comparison of significance tests for balance-scale dataset

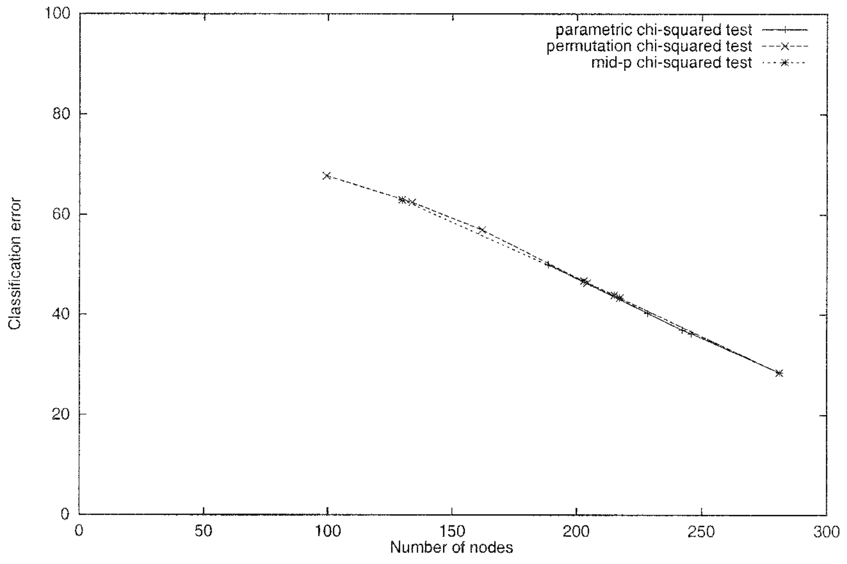


Figure 18. Comparison of significance tests for vowel dataset

significantly. It is safe to conclude that, in the context of post-pruning decision trees, the three tests closely approximate each other if the significance level is chosen appropriately. The figures also show that, as in the artificial examples from the previous section, the tests can be ordered according to how aggressively they prune, given a particular fixed significance level. The permutation chi-squared test is the most aggressive test, followed by its mid-p version, followed by the parametric chi-squared test.

A second result is that when significance testing is performed, the trees are often substantially smaller but no less accurate. The reduction in tree size is particularly dramatic for the australian, breast-cancer, heart-h, hepatitis, horse-colic, pima-indians, and vote datasets. In each of these, the estimated number of nodes is reduced by at least 50% if the most aggressive kind of significance testing is performed. For the breast-cancer dataset, the maximum reduction is close to 94% (3.1 instead of 48.6 nodes), indicating that this dataset contains very little information. Even when the most liberal test is used for pruning—in other words, the parametric chi-squared test at a significance level of 0.4—the resulting trees are often significantly smaller.

Sometimes, for example in the australian dataset, significance testing increases accuracy. However, the gain is small. In at least three cases, namely autos, vowel, and zoo, pruning with significance tests decreases accuracy. However, mild pruning often produces negligible loss of accuracy, and significant loss only occurs with more aggressive pruning. Examples for when aggressive pruning is harmful are the audiology, balance-scale, glass-2, primary-tumor, and soybean datasets.

6. Minimizing tree size

The size of the pruned decision tree can be determined by adjusting the significance level of the statistical test. However, the experiments in the previous section show that there is a trade-off: if the significance level is set too small, accuracy declines significantly—and sometimes no additional pruning at all is warranted. Ideally, we want to prune the tree to the point just before accuracy starts to deteriorate, and no further. However, as the results from the previous section show, the "best" significance level depends on the particular properties of the domain—for example, the amount of noise that is present. Hence the appropriate value must be found individually for each domain.

This is an optimization problem that occurs very frequently in machine learning: a parameter setting is required which optimizes predictive performance on future test data. The standard solution is to derive a ten-fold cross-validation estimate of the accuracy for each parameter setting, and choose the setting that maximizes this estimate. Once the optimum value of the parameter has been determined, the learning algorithm is then applied to the full training dataset using this value.

When this method is used to choose a pruning parameter, there is one additional complication. Often parameter setting s_1 produces a significantly smaller classifier than parameter setting s_2, at the cost of a small but statistically insignificant drop in accuracy (Breiman, Friedman, Olshen & Stone, 1984). Because of statistical fluctuations in the cross-validation estimate, this can happen even if the expected accuracies for the two parameter settings are in fact the same. The experimental results from the previous section exhibit several datasets where this is likely to occur—all those where the graph is parallel to the x-axis. A standard way of circumventing this problem is to consider all parameter values for which the estimated error is within one standard error of the smallest error estimate observed, and choose the one that produces the smallest trees (Breiman, Friedman, Olshen & Stone, 1984). When pruning with significance tests, the size of the tree decreases with the significance level. Thus optimization chooses the smallest significance level that produces a tree whose error is within one standard deviation of the most accurate tree.

The remainder of this section discusses results obtained by applying this procedure—both with and without the one-standard-error rule—in conjunction with the parametric chi-squared test. As before, the significance test is applied whenever reduced-error pruning decides to retain a subtree. During optimization, seven significance levels are considered: 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, and 1. Using the last value in this list, all subtrees are considered significant, and the resulting tree is the same as for standard reduced-error pruning. The values 0.1, 0.2, 0.3 and 0.4 were used for the experiments in the previous section. The first two values are included because the experimental results from the previous section show that sometimes even more aggressive pruning is beneficial.

Table 2 shows the accuracies for the 27 datasets introduced above, and Table 4 shows the tree sizes in number of nodes. As well as results for standard reduced-error pruning the tables include results for pruning with significance testing, both, without (SIG) and with (SIG-SE) the one-standard-error rule. All estimates are derived by repeating ten-fold cross-validation ten times. Note that these cross-validation runs are performed *in addition* to the cross-validation performed during internal optimization. The standard deviation for the ten data points corresponding to the ten cross-validation estimates are also shown in the tables. The $\circ$ ($\bullet$) symbol indicates for which datasets pruning based on significance testing produces more (less) accurate trees than reduced error pruning—or smaller (larger) trees respectively. A difference is considered to be significant if the corresponding cross-validation estimates are significantly different at the 0.05% level according to a paired two-sided t-test for the ten data points involved.

The results of the significance tests for accuracy and size are summarized in Tables 3 and 5 respectively. In these tables, each entry indicates the number of datasets for which the method associated with its column significantly outperforms the method associated with its row.

Tables 3 shows that pruning with significance testing (SIG) produces trees that are at least as accurate as those for reduced-error pruning (REP) if the one-standard-error rule is *not* used. More specifically, SIG is significantly more accurate than REP on two datasets (second column, first row) and never less accurate (first column, second row). On the other hand, if the one-standard-error rule is used (SIG-SE), performance degrades significantly compared to REP on 11 datasets. This indicates

Table 2. Accuracies for reduced-error pruning (REP) compared to significance pruning without (SIG) and with (SIG-SE) the one-standard-error rule

Dataset	REP	SIG		SIG-SE	
anneal	98.5±0.4	98.6±0.2		98.4±0.4	
audiology	72.8±1.5	72.1±2.1		69.9±1.0	●
australian	84.3±0.9	85.0±0.4	○	85.4±0.2	○
autos	63.3±2.8	63.2±2.9		61.0±3.1	
balance-scale	78.4±0.9	77.7±1.4		76.5±1.2	●
breast-cancer	68.8±1.6	69.1±1.3		69.8±0.6	
breast-w	94.3±0.7	93.7±0.8		93.5±0.4	●
german	72.4±1.4	71.5±0.8		70.4±1.2	●
glass-2	79.0±2.3	78.6±3.0		77.3±3.4	
glass	66.6±3.4	66.0±2.9		62.5±2.6	●
heart-c	75.8±2.3	76.3±1.5		74.8±1.4	
heart-h	78.0±1.7	78.1±1.4		78.9±1.5	
heart-statlog	76.3±2.8	77.2±3.1		75.6±3.6	
hepatitis	80.8±2.5	79.6±2.0		80.0±1.7	
horse-colic	84.5±0.8	84.0±0.8		83.9±1.1	
ionosphere	89.0±0.8	89.7±0.9	○	89.8±0.8	○
iris	94.6±0.8	94.0±1.0		93.9±1.1	
labor	79.5±3.9	77.5±7.5		75.8±6.9	
lymphography	75.2±1.7	73.8±2.5		72.7±1.9	●
pima-indians	74.0±0.6	73.6±1.4		73.8±1.6	
primary-tumor	37.7±1.9	36.7±1.3		35.0±1.5	●
sonar	71.3±2.4	70.1±3.0		68.5±2.7	●
soybean	85.0±0.9	84.6±1.0		83.5±0.9	●
vehicle	71.0±1.1	70.1±1.1		69.2±1.5	●
vote	95.6±0.5	95.5±0.3		95.6±0.1	
vowel	71.6±1.5	71.0±1.5		71.0±1.5	
zoo	90.2±2.3	88.5±2.2		86.3±2.5	●

Table 3. Results of paired t-tests (p=0.05) for accuracies: number indicates how often method in column significantly outperforms method in row

	REP	SIG	SIG-SE
REP	–	2	2
SIG	0	–	1
SIG-SE	11	13	–

that SIG-SE overprunes on these 11 datasets: it produces decision trees that are too small and do not capture all the relevant information.

Table 4. Tree sizes for reduced-error pruning (REP) compared to significance pruning without (SIG), and with (SIG-SE) one-standard-error rule

Dataset	REP	SIG		SIG-SE	
anneal	43.8±2.7	43.1±2.5		42.3±2.8	
audiology	40.2±2.6	36.9±2.3	∘	28.2±1.6	∘
australian	34.3±7.9	9.4±3.1	∘	4.2±1.5	∘
autos	53.6±2.8	53.5±1.9		48.0±2.3	∘
balance-scale	54.6±3.2	48.6±2.0	∘	33.8±2.5	∘
breast-cancer	48.6±5.3	13.8±6.0	∘	1.9±1.1	∘
breast-w	15.1±1.5	14.1±2.4		9.9±1.5	∘
german	95.8±9.5	66.7±10.8	∘	32.8±11.3	
glass-2	12.7±2.3	9.6±1.0	∘	7.6±1.3	∘
glass	23.4±1.8	22.3±2.4		15.4±1.9	∘
heart-c	21.1±1.6	15.4±2.6	∘	8.3±1.2	∘
heart-h	15.3±1.8	8.8±2.6	∘	5.2±0.9	∘
heart-statlog	15.7±1.6	12.8±2.3	∘	7.1±1.5	∘
hepatitis	6.5±1.9	3.6±1.0	∘	1.9±0.3	∘
horse-colic	17.3±3.9	11.1±3.0	∘	6.4±0.9	∘
ionosphere	10.6±2.1	7.6±0.8	∘	6.8±0.6	∘
iris	6.4±0.5	5.7±0.4	∘	5.6±0.3	∘
labor	6.7±0.8	5.4±0.8	∘	4.8±0.7	∘
lymphography	14.6±3.0	12.1±2.2	∘	7.4±1.3	∘
pima-indians	35.8±5.4	21.6±4.5	∘	9.0±2.5	∘
primary-tumor	49.9±4.7	42.3±3.6	∘	21.5±3.9	∘
sonar	11.0±1.3	8.1±1.3	∘	5.3±0.9	∘
soybean	89.1±1.9	88.4±3.4		73.3±3.7	∘
vehicle	65.5±4.5	58.3±4.9	∘	37.1±3.9	∘
vote	7.5±1.2	4.2±0.6	∘	3.4±0.5	∘
vowel	280.9±4.3	281.4±4.5		281.4±4.5	
zoo	14.1±0.4	13.5±0.6	∘	12.0±1.0	∘

Table 5. Results of paired t-tests (p=0.05) for sizes: number indicates how often method in column significantly outperforms method in row

	REP	SIG	SIG-SE
REP	–	21	25
SIG	0	–	24
SIG-SE	0	3	–

Tables 5 shows how the methods compare with respect to the size of the pruned trees. Because SIG and SIG-SE can never prune less than REP, they never produce larger trees. On the other hand, SIG produces significantly smaller trees than REP for 21 datasets. For several datasets, for example, australian, breast-cancer, heart-h, hepatitis, pima-indians, and vote, the reduction in tree size is quite dramatic. SIG-SE produces even smaller trees than SIG—on 24 datasets they are significantly smaller. However, as Tables 3 shows, it often produces less accurate trees.

Taken together, these results mean that pruning with significance tests successfully improves on reduced-error pruning if the significance level is chosen according to the properties of the domain. It often produces much smaller trees, and the trees are never significantly less accurate, and sometimes more accurate. The appropriate significance level can be identified automatically by cross-validation. If the one-standard-error rule is used in conjunction with the cross-validation estimate, tree sizes can be reduced even further. However, this sometimes results in a significant loss of accuracy.

7. Related work

Pruning methods for decision trees are one of the most extensively researched areas in machine learning. Several surveys of induction methods for decision trees have been published (Safavian & Landgrebe, 1991; Kalles, 1995; Murthy, 1998), and they also discuss different pruning strategies. In addition, empirical comparisons of a variety of different pruning methods have been conducted. This section first discusses the most popular pruning methods in the context of published experimental comparisons, highlighting their weaknesses as well as proposed remedies. Then we summarize prior work on the problem of underpruning, of which a particular instance is tackled in this paper. Finally, we briefly discuss less well known pruning techniques and modifications to existing procedures.

Quinlan (1987) was the first to perform a comparison of pruning methods. He presents experimental results for three of them: cost-complexity pruning, reduced-error pruning, and pessimistic pruning. Another experimental comparison of several pruning methods has been performed by Mingers (1989). As well as the three methods investigated by Quinlan, Mingers includes two more procedures in his comparison: critical value pruning and minimum-error pruning. However, his comparison has been criticized because he does not give all pruning methods access to the same amount of data. Also, he uses a non-standard version of reduced-error pruning. The critics, Esposito, Malerba, and Semeraro (1997), published a paper comparing essentially the same pruning algorithms. In order to make a fair comparison, their experimental procedure assures that all algorithms have access to the same amount of data when generating the pruned tree. In contrast to Mingers, they use Quinlan's original version of reduced-error pruning, as well as a more recent incarnation of minimum error pruning. Their paper includes results for a successor of pessimistic error pruning called error-based pruning.

7.1. Cost-complexity pruning

Cost-complexity pruning was introduced in the classic CART system for inducing decision trees (Breiman, Friedman, Olshen & Stone, 1984). It is based on the idea of pruning first those subtrees that, relative to their size, lead to the smallest increase in error on the training data. The increase in error is measured by a quantity α that is defined to be the average increase in error per leaf of the subtree. CART uses α to generate a sequence of increasingly smaller pruned trees: in each iteration, it prunes all subtrees that exhibit the smallest value for α. Each tree corresponds to one particular value α_i. In order to choose the most predictive tree in this sequence, CART either uses a hold-out set to estimate their classification error, or employs cross-validation. Cross-validation poses the additional problem of relating the α_j^k values observed in training fold k to the α_i values from the original sequence of trees. These values are usually different. CART solves this problem by computing the geometric average α_i^{av} of α_i and α_{i+1} for tree i from the original sequence. Then, for each fold k of the cross-validation, it picks the tree that exhibits the largest α_j^k value smaller than α_i^{av}. The average of the error estimates for these trees is the cross-validation estimate for tree i.

Discussion It can be shown that the sequence of pruned trees can be generated in time that is quadratic in the number of nodes (Esposito, Malerba & Semeraro, 1997). This is significantly slower than the pruning methods investigated in this paper, which are linear in the number of nodes. It implies that CART's runtime is quadratic in the number of training instances if the number of nodes increases linearly with the number of training instances—a realistic scenario in noisy real-world datasets. Note that, as well as allowing for cross-validated error estimates, CART also introduced the one-standard-error rule discussed in Section 6.

Quinlan (1987, page 225) notes that it is unclear why the particular cost-complexity model used by CART "...is superior to other possible models such as the product of the error rate and number of leaves," and he also finds that "...it seems anomalous that the cost-complexity model ...is abandoned when the best tree is selected." Consequently he introduces two new pruning methods: reduced-error pruning, which has been discussed above, and pessimistic error pruning.

7.2. *Pessimistic error pruning*

Pessimistic error pruning is based on error estimates derived from the training data. Hence it does not require a separate pruning set. More specifically, pessimistic error pruning adds a constant to the training error of a subtree by assuming that each leaf automatically classifies a certain fraction of an instance incorrectly. This fraction is taken to be 1/2 divided by the total number of instances covered by the leaf, and is called a "continuity correction" in statistics (Wild & Weber, 1995). In that context it is used to make the normal distribution more closely approximate the binomial distribution in the small sample case. During pruning this adjustment is used in conjunction with the one-standard-error rule above. A tree is made into a leaf if the *adjusted* error estimate for the leaf is smaller or equal to the *adjusted* error of the tree plus one standard error of the latter estimate. The standard error is computed by assuming that the *adjusted* error is binomially distributed. A subtree is considered for replacement *before* its branches are pruned.

Discussion In contrast to reduced-error pruning, which proceeds in a bottom-up fashion, pessimistic error pruning uses the top-down approach and considers pruning a tree before it prunes its subtrees. Hence it is marginally faster in practice. However, its worst-case time complexity is also linear in the number of nodes in the unpruned tree. As Breslow and Aha (1997b) note, top-down pruning methods suffer from the "horizon effect": a tree might be pruned even when it contains a subtree that "... would not have been pruned by the same criterion."

In his experiments comparing cost-complexity pruning, reduced-error pruning, and pessimistic error pruning, Quinlan (1987) finds that cost-complexity pruning tends to over-prune: it generates the smallest trees but they are slightly less accurate than those produced by the other two methods. He concludes that pessimistic error pruning is the preferred method since it does not require a separate pruning set.

Buntine (1992) also compares pessimistic pruning, and variants of cost-complexity pruning. However, none of these variants seems to implement the version of cost-complexity pruning as defined by Breiman *et al.* (1984): Buntine uses the *pruning* data to compute the cost-complexity of a tree for a given α instead of the *training* data used in the original formulation. In his terminology, the pruning data is called the "test set," and he claims that "The substitution error estimate is usually computed on a test set" (page 93). He later acknowledges that "The most likely place for bugs in the existing implementation [of the tree learner used in his experiments] is in the cost complexity pruning module,..." (page 99). Buntine's implementation is likely to produce overly complex decision trees because it gives the algorithm a chance to fit the pruning data before the final pruned tree is selected.

7.3. Critical value pruning

Critical value pruning (Mingers, 1987) is a bottom-up technique like reduced-error pruning. However, it makes pruning decisions in a fundamentally different way. Whereas reduced-error pruning uses the estimated error on the pruning data to judge the quality of a subtree, critical-value pruning looks at information collected during tree growth. Recall that a top-down decision tree inducer recursively employs a selection criterion to split the training data into increasingly smaller and purer subsets. At each node it splits in a way that maximizes the value of the splitting criterion, for example, the information gain. Critical value pruning uses this value to make pruning decisions. When a subtree is considered for pruning, the value of the splitting criterion at the corresponding node is compared to a fixed threshold, and the tree is replaced by a leaf if the value is too small. However, one additional constraint is imposed: if the subtree contains at least one node whose value is greater than the threshold, it will not be pruned. This means that a subtree is only considered for pruning if all its successors are leaf nodes.

Discussion The performance of critical value pruning depends on the threshold used for the pruning decisions. Assuming that the splitting criterion's value increases with the quality of the split, larger thresholds result in more aggressive pruning. Of course, the best value is domain-dependent and must be found using a hold-out set or cross-validation. The main difference between critical value pruning and other post-pruning methods is that it looks solely at the information provided by the splitting criterion, only indirectly taking account of the accuracy of the subtree considered for pruning. Therefore, its performance depends critically on how well the splitting criterion predicts the subtree's generalization performance. Most splitting criteria, however, do not take account of the number of instances that support the subtree. Consequently the procedure overestimates the quality of subtrees that cover only a few instances. Among those splitting criteria used by Mingers, only the chi-squared distribution is an exception. However, because it is used to grow the tree, the derived probabilities are highly biased (Jensen & Schmill, 1997).

7.4. Minimum error pruning

Minimum-error pruning was invented by Niblett and Bratko (1986). It is similar to pessimistic error pruning in that it uses class counts derived from the training data. However, it differs in the way it adjusts these counts in order to more closely reflect a leaf's generalization performance. In its initial version, which was used by Mingers (1989), the adjustment is a straightforward instantiation of the Laplace correction, which simply adds one to the number of instances of each class when the error rate is computed. Like reduced-error pruning, minimum-error pruning proceeds in a bottom-up fashion, replacing a subtree by a leaf if the estimated error for the former is no smaller than for the latter. In order to derive an estimate of the error rate for a subtree, an average of the error estimates is computed for its branches, weighted according to the number of instances that reach each of them.

Discussion In a later version of minimum error pruning, Cestnik and Bratko (1991) refine the Laplace heuristic. Instead of adding one to the count for each class, they add a constant $p_i \times m$, where p_i is the class' prior probability in the training data and m is a factor that determines the severity of the pruning process. Higher values for m generally produce smaller trees because they reduce the influence of the training data and result in a "smoothing" effect that tends to equalize the probability estimates at different leaves. However, a higher value does not automatically produce a smaller tree (Esposito, Malerba & Semeraro, 1997). This is a significant disadvantage because it means that for each value of m considered, the procedure must begin with an unpruned tree. Since the best value for m can only be found by estimating the error of the pruned tree on a hold-out set, or using cross-validation, this property makes minimum error pruning significantly slower than reduced error pruning.

Mingers (1989) compares the five pruning methods discussed above. In his experiments, those methods that use a separate pruning set outperform ones that are just based on estimates from the training data. However, the comparison has been criticized for the experimental methodology employed (Esposito, Malerba & Semeraro, 1997). In Mingers' experiments, methods with a separate pruning set for parameter selection or error estimation have an unfair advantage because the pruning data is provided *in addition* to the training data used for the other methods. His experimental results for critical value pruning, error-complexity pruning and reduced-error pruning on the one side, and minimum error pruning and pessimistic error pruning on the other side, can therefore not be compared directly.

However, his results *can* be used to compare the performance of the methods within each group (Breslow & Aha, 1997b). They show, for example, that minimum-error pruning produces less accurate and larger trees than pessimistic error pruning. They also show that cost-complexity pruning produces smaller trees than reduced-error pruning with similar accuracy. However, this latter result is questionable because Mingers' version of reduced-error pruning differs from the original algorithm proposed by Quinlan (1987). In Mingers' experiments, critical value pruning is less accurate than both cost-complexity and reduced-error pruning, and it produces larger trees than cost-complexity pruning. Mingers also investigated whether different splitting criteria and pruning methods interact, and found that this is not the case. This means that these two components of a decision tree inducer can be studied independently (Breslow & Aha, 1997b).

7.5. Error-based pruning

Error-based pruning is the strategy implemented by the well-known decision tree inducer C4.5 (Quinlan, 1992). A similar strategy has also been proposed by Kalkanis (1993). Like pessimistic error pruning, it derives error estimates from the training data, assuming that the errors are binomially distributed. However, instead of the one-standard-error rule employed by pessimistic error pruning, it computes a confidence interval on the error counts based on the fact that the binomial distribution is closely approximated by the normal distribution in the large sample case. Then, the upper limit of this confidence interval is used to estimate a leaf's error rate on fresh data. In C4.5, the confidence interval is set to 25% by default. Like reduced-error pruning—and in contrast with pessimistic error pruning—a bottom-up traversal strategy is employed: a subtree is considered for replacement by a leaf after all its branches have already been considered for pruning. Replacement is performed if the error estimate for the prospective leaf is no greater than the sum of the error estimates for the current leaf nodes of the subtree. As well as subtree replacement, C4.5 also performs a pruning operation called "subtree raising" that replaces a subtree with its most popular branch if this does not increase the estimated error.

Discussion Using a confidence interval is a heuristic way of reducing the optimistic bias in the error estimate derived from the training data, but it is not statistically sound, and Quinlan (1992) acknowledges this fact. From a statistical perspective this pruning procedure shares the problems of pessimistic error pruning. The use of the normality assumption is also questionable because it is only correct in the limit. For small samples with less than 100 instances, statisticians use Student's distribution instead of the normal distribution (Wild & Weber, 1995). In decision tree induction, small samples are exactly those which are most likely to be relevant in the pruning process.

Esposito *et al.* (1997) claim that error-based pruning and pessimistic error pruning behave in the same way. However, this assertion seems to conflict with the experimental results presented in their paper. For all but one of 15 datasets investigated, pessimistic error pruning produces significantly smaller trees than error-based pruning (see Table 8 in their paper). With respect to accuracy, however, the differences between the two methods are only minor. Considering these results, it is not clear why error-based pruning has replaced pessimistic error pruning in C4.5.

In contrast to Mingers (1989), Esposito *et al.* (1997) do not find that methods operating with a pruning set produce more accurate trees than those that rely solely on the training data. As mentioned above, this difference is due to the particular experimental procedure that Mingers employs.

Esposito *et al.* (1997) also introduce the notion of an "optimally pruned tree." This tree is derived by applying reduced error pruning using the *test set* as the pruning data. The authors claim that it is possible to determine if a particular method overprunes or underprunes by comparing its tree size to the size of the optimally pruned tree. However, this neglects the fact that reduced-error pruning often overfits the pruning data. As the results from this paper show, their procedure is likely to detect overpruning when in fact even more pruning is warranted. However, it is safe to say that it correctly detects underpruning, and it shows that minimum error pruning, critical value pruning, and error-based pruning generally produce trees that are too large.

Breslow and Aha (1997b) summarize the main results from Mingers (1987) and Esposito *et al.* (1997) by showing differences and similarities in their findings. They conclude that pessimistic error pruning and error-based pruning produce the most accurate trees among the methods compared. However, this claim seems too general. The results from Esposito *et al.* show that these two pruning methods are only superior when little pruning is warranted; in several cases they produce less accurate trees than, for example, reduced-error pruning. This is a direct result of their tendency to underprune, which is also mentioned by Breslow and Aha. They conclude that "...these findings on post-pruning algorithms are preliminary ..." and that "...further investigation is needed ..."

The same authors (1997a) have also published an empirical comparison of "tree-simplification procedures" that includes two pruning methods: error-based pruning in Revision 8 of C4.5 (Quinlan, 1996), and ITI's pruning procedure (Utgoff, Berkman & Clouse, 1997) that is based on the minimum description length principle (Quinlan & Rivest, 1989). They found that C4.5 generally performed best. They also performed an experiment in which they tuned C4.5's pruning parameters using nine-fold cross-validation. Unfortunately they compare these results to those of an *unpruned* tree. Consequently it is difficult to judge whether parameter tuning improves on the default parameter settings. Kohavi (1995, page 122) also uses cross-validation to choose parameter settings for C4.5. He reports that it is "... hard to judge whether C4.5 with automatic parameter tuning ... is significantly superior to C4.5." However, he only looks at the accuracy of the resulting trees, not their size.

7.6. Underpruning

Oates and Jensen (1997) were the first to observe that bottom-up procedures like error-based pruning and reduced-error pruning can produce overly large decision trees. They find that the size of pruned decision trees can be reduced significantly, with only a small loss in accuracy, if a random subsample of the original training data is passed to the induction algorithm instead of the full dataset. This implies that the pruning process does not simplify the tree sufficiently when more data becomes available. Later, they argue that the reason for this overfitting is a phenomenon they call "error propagation" (Oates & Jensen, 1998). Error propagation is due to to the bottom-up fashion in which pruning proceeds: a subtree is considered for replacement by a leaf *after* its branches have been considered for pruning. Recall that a subtree is only retained if it has lower estimated error than the corresponding leaf. Because the subtree has already been modified before the error estimate is computed, it is optimistically biased, rendering it less likely to be pruned—spurious correlations in the lower parts of the subtree are propagated to its root node, making it unduly likely that it will be retained. Of course, the deeper the subtree, the more opportunities there are for these spurious associations to occur, and the more likely it is that the subtree will survive. Oates and Jensen (1999) show that it is possible to quantify the survival probability in an approximate way by making some simplifying assumptions. They also propose two ways of preventing overly complex decision trees from being built. Note that this paper shows how standard significance tests can be used to detect spurious correlations, thereby successfully preventing error propagation in reduced-error pruning.

Randomization pruning The first method proposed by Oates and Jensen (1998), called "randomization pruning," is based on the idea of a permutation test, and is applied as a post-processing step to simplify the pruned tree. For each subtree of the pruned tree, the probability $\hat{p}$ that an equally or more accurate subtree would have been generated *by chance alone* is computed. To do this, the procedure collects the training data from which the subtree was built, and randomly permutates its class labels. Then it applies the decision tree inducer to this randomized data, and records the accuracy of the resulting tree. This randomization procedure is repeated N times. The probability $\hat{p}$ is the fraction of times for which the randomized subtrees are no less accurate than the original subtree. If $\hat{p}$ is greater than a certain threshold—the authors suggest 0.05—the subtree is discarded and replaced by a leaf node. In the three datasets investigated by the authors, this procedure successfully reduces the size of the pruned trees built by error-based pruning, and slightly reduces accuracy in only one of them.

This randomization procedure applies a significance test just like the methods investigated in this paper. However, it differs in that it is computationally very expensive. The induction algorithm is applied N times for each subtree in the original pruned tree—and in order to obtain accurate probability estimates, the required value for N can be anywhere between several hundred and several thousand. Therefore this procedure is of theoretical rather than practical value.

Reduced-overlap pruning The second approach proposed by Oates and Jensen (1999) is designed to improve reduced-error pruning. It is based on the idea that unbiased error estimates for each subtree can be obtained if a fresh set of pruning data is available to evaluate each of them. Using artificial datasets, where unlimited amounts of new data can be generated, the authors show that this successfully prevents reduced-error pruning from building overly complex decision trees. In practice, however, only a limited amount of pruning data is available, and so the authors propose to use random subsamples of the original pruning data instead of fresh data to derive error estimates at each node of the tree. The idea is to minimize the overlap between these random subsamples in order to minimize dependencies between the error estimates. Using random subsamples, each containing 50% of the original pruning data, Oates and Jensen find that this method leads to significantly smaller trees for 16 out of 19 practical datasets investigated, and significantly decreases accuracy on only one of them. However, it is plausible that the reduction in tree size is solely due to the smaller amount of pruning data that is used at each node, and does not result from the increased independence between the samples. We repeated their experiment using the datasets and the experimental setting from the last section, and found that it significantly decreased accuracy for 12 of the 27 datasets and produced significantly smaller trees for all 27.

7.7. *Other pruning methods*

Apart from the pruning algorithms discussed above, several less well-known methods have been proposed in the literature that are either modifications of existing algorithms or based on similar ideas.

Crawford (1989) uses cost-complexity pruning in conjunction with the .632 bootstrap (Efron & Tibshirani, 1993) for error estimation, substituting it for the standard cross-validation procedure. However, Weiss and Indurkhya (1994a,b) demonstrate that cross-validation is almost unbiased and close to optimal in choosing the right tree size. Kohavi (1995) shows that the .632 bootstrap has higher bias but lower variance than cross-validation, noting that it can be preferable for small sample sizes. Later, Efron and Tibshirani (1997) proposed an improved bootstrap estimator, the .632+ bootstrap, with lower bias. Gelfand *et al.* (1991) modify CART's pruning procedure by interleaving the growing and pruning phases: a tree is grown using one half of the data, then pruned using the other half. In subsequent iterations, the existing tree continues to be modified by these two steps, but in each iteration the roles of pruning and growing data are exchanged. According to results presented by Gelfand *et al.*, this procedure speeds up the pruning process and produces more accurate trees. However, the trees are also larger.

Minimum description length principle Several authors have proposed pruning methods based on the minimum description length (MDL) principle (Rissanen, 1978). These methods derive from the idea that a successful inducer will produce a classifier that *compresses* the data, and exploit the fact that the complexity of a model, as well as the complexity of a dataset, can be measured in "bits" given an appropriate coding scheme. Induction is considered to be successful if the cost of coding both the classifier, and its classification errors, is lower than the cost of coding the training data itself. Moreover, the greater the reduction in coding cost (the compression), the "better" the inducer. MDL pruning algorithms seek decision trees that maximally compress the data. They differ in the coding scheme they employ. Successful application of the MDL principle depends on how well the coding scheme matches the particular properties of the learning problem at hand. This is a direct consequence of the fact that it is a reformulation of Bayes' rule (Buntine, 1992), in which probabilities have been replaced by their logarithms. The prior probability in Bayes' rule determines the model's coding cost, and it is essential that the distribution of the prior probabilities is chosen appropriately. In the case of decision trees, for example, different prior distributions result in different amounts of pruning. Proponents of the MDL principle claim that it has two advantages: no parameters need to be chosen, and no pruning data needs to be set aside. However, they omit to mention that the choice of the prior distribution is a parameter in itself, and one can argue that it is a *disadvantage* that this parameter can not be freely adjusted.

Quinlan and Rivest (1989) were the first to use the MDL principle for pruning decision trees. They compare it experimentally to pessimistic error pruning and obtain mixed results. Wallace and Patrick (1993) point out flaws in their coding scheme, but acknowledge that these flaws do not affect the outcome. Forsyth (1994) also uses an MDL approach, as do Mehta *et al.* (1995). The latter authors report that their method produces smaller trees than both pessimistic error pruning and error-based pruning, but larger trees than cost-complexity pruning. Error rates are similar in each case.

Optimal pruning Another line of research investigates "optimal" pruning algorithms that produce a sequence of smaller and smaller pruned trees, where each tree has the property that it is the most accurate one on the training data among all pruned trees of the same size. Breiman *et al.* (1984) were the first to suggest a dynamic programming solution to this problem, and Bohanec and Bratko (1994) present a corresponding algorithm. They note that cost-complexity pruning, discussed above, produces a sequence of optimal trees that is a subset of the sequence generated by their method. The worst-case time complexity of their algorithm is quadratic in the number of leaves of the unpruned tree. Almuallim (1996) presents an improved optimal pruning algorithm, also based on dynamic programming that has slightly lower worst-case time complexity. Neither method addresses the question of how to choose tree size in order to maximize generalization performance. Bohanec and Bratko suggest that this can be done using cross-validation, but do not test this experimentally.

Pruning with costs Often, real-world learning problems involve costs because some classification errors are more expensive than others. The literature contains several approaches to cost-sensitive pruning (Knoll, Nakhaeizadeh & Tasend, 1994; Bradford, Kunz, Kohavi & Brunk, 1998; Vadera & Nechab, 1994). Ting (1998) presents an elegant solution for incorporating costs that is based solely on weighting the training instances. By resampling instances with a probability proportional to their weight, his methods can be applied to learning schemes that cannot make use of weights directly.

Pruning with significance tests Statistical significance tests have been applied to learning algorithms before, but in almost all cases using information derived during training—a procedure that is questionable if appropriate adjustments are not performed (Cohen & Jensen, 1997). In Quinlan's ID3 decision tree inducer (1986), for example, the parametric chi-squared test is used to decide when to stop splitting the training data into increasingly smaller subsets—the classical pre-pruning method. The same technique is also used by the decision tree inducer CHAID (Kass, 1980).

Jensen *et al.* (1997) apply critical value pruning in conjunction with the chi-squared distribution. However, instead of using the probabilities directly—which is incorrect because they have been used for training—they apply a statistical technique known as the "Bonferroni correction" to make an appropriate adjustment. Statisticians use the Bonferroni correction to adjust the significance levels of multiple statistical hypothesis tests (Westfall & Young, 1993). It requires that the p-values of the tests are independent, which is unlikely to be true in real-world learning situations (Jensen, 1992). If the independence assumption is not fulfilled, a permutation test on the original test's p-values is the statistically correct way of adjusting for multiple comparisons (Westfall & Young, 1993). In order to apply the Bonferroni adjustment, it is necessary to know *in advance* how many significance tests will be performed. However, because pruning is a dynamic process, this knowledge is impossible to achieve *a priori*. Despite these theoretical problems, Jensen and Schmill (1997) report good results for their method in practice: it often produces smaller trees than C4.5 and is seldom less accurate.

Some authors have investigated the use of permutation tests in learning algorithms. Gaines (1989) uses the one-sided version of Fisher's exact test to evaluate classification rules, and employs the Bonferroni correction to adjust for multiple tests. Jensen (1992) proposes a *conditional* permutation test based on the chi-squared statistic to decide when to modify and expand rules in a prototypical rule learner. In order to perform the significance test, he uses fresh data that has not been used for model fitting. Li and Dubes (1986) propose a version of Fisher's exact test for attribute selection and pre-pruning in binary domains. Frank and Witten (1998b) use the more general Freeman and Halton test for the same purpose, and find that post-pruning with error-based pruning performs better. Martin (1997) proposes the *test statistic* of the Freeman and Halton test for attribute selection and pre-pruning; however, results improve when the full significance test is used instead (Frank & Witten, 1998b). Hong *et al.* (1996) compute the expected value of the test statistic under the permutation distribution and use this to normalize the value from the original data. They propose to use this normalized value for attribute selection and pre-pruning.

Computational learning theory There has also been some work in computational learning theory on post-pruning algorithms for decision trees. Kearns and Mansour (1998) extend earlier work by Mansour (1997), and present an bottom-up algorithm similar to C4.5's error-based pruning that produces a near-optimum pruning so that—in a theoretical sense—its generalization error is almost as low as the one for the hypothetical best pruning of the tree. However, the experimental results presented by Mansour (1997) show that there is little difference between the two methods in practice.

8. Conclusions

This paper investigates whether standard significance tests can be used to improve the reduced-error pruning algorithm to make decision trees smaller and more accurate. The experimental results show that, if the tests' significance levels are adjusted according to the amount of pruning required by the domain, the pruned decision trees are indeed consistenly smaller, and at least as accurate, as with regular reduced-error pruning. They also show that an appropriate significance level can be found automatically using cross-validation. This supports the primary hypothesis of this paper.

Experiments comparing the performance of permutation tests and the parametric chi-squared test show that they all produce trees of different sizes for a given significance level. However, the differences can be eliminated by tuning the significance level for each test individually. Hence the secondary hypothesis of this paper turns out to be incorrect: in practice the parametric test and the permutation tests produce pruned trees with very similar size and accuracy if the significance levels for each test are chosen appropriately. Since permutation tests are computationally more expensive than parametric ones, there is no reason to use them in this particular application.

For a fixed significance level, the additional computational complexity incurred by a significance test is negligible when the parametric test is employed. However, for best results the significance level needs to be chosen via cross-validation or a hold-out set. Cross-validation, which is the preferred method for small datasets, increases run time by a constant factor. The fully expanded tree needs to be generated once for each fold of the cross-validation. The error estimates for the different significance levels can be obtained simultaneously because they produce a nested sequence of trees—decreasing the significance level always results in more pruning.

In time-critical applications, where the classifier's perspicuity is not an issue, there is often no advantage in using significance tests over the standard reduced-error pruning procedure. However, when comprehensibility is important, the extra time required for cross-validation is well spent.

Notes

1. Note that statisticians use the term "discrete" rather than "sparse."

References

Agresti, A. (1990). *Categorical Data Analysis.* New York: John Wiley & Sons.

Almuallim, H. (1996). An efficient algorithm for optimal pruning of decision trees. *Artificial Intelligence, 83(2)*, 347–362.

Blake, C., Keogh, E. & Merz, C. (1998). Uci repository of machine learning databases. Technical report, University of California, Department of Information and Computer Science, Irvine, CA. [www.ics.uci.edu/ mlearn/MLRepository.html].

Bohanec, M. & Bratko, I. (1994). Trading accuracy for simplicity in decision trees. *Machine Learning, 15(3)*, 223–250.

Bradford, J. P., Kunz, C., Kohavi, R. & Brunk, C. (1998). Pruning decision trees with misclassification costs. *Lecture Notes in Computer Science, 1398*, 131–??

Breiman, L., Friedman, J. H., Olshen, R. A. & Stone, C. J. (1984). *Classification and Regression Trees.* Belmont, California: Wadsworth.

Breslow, L. A. & Aha, D. W. (1997a). Comparing tree-simplification procedures. In *AISTATS-97: Proc Sixth International Workshop on AI & Statistics.* unpublished. Ft. Lauderdale, FL.

Breslow, L. A. & Aha, D. W. (1997b). Simplifying decision trees: A survey. *Knowledge Engineering Review, 12(1)*, 1–40.

Buntine, W. L. (1992). *A Theory of Learning Classification Rules.* PhD thesis, School of Computing Science, University of Technology, Sydney.

Cestnik, B. & Bratko, I. (1991). On estimating probabilities in tree pruning. In Kodratoff, Y. (Ed.), *Machine Learning: EWSL-91*, number 482 in Lecture Notes in Artificial Intelligence (pp. 138–150). Berlin: Springer Verlag.

Cochran, W. (1954). Some methods of strengthening the comman χ^2 tests. *Biometrics, 10*, 417–451.

Cohen, P. R. & Jensen, D. (1997). Overfitting explained. In *Preliminary Papers of the Sixth International Workshop on Artificial Intelligence and Statistics* (pp. 115–122). unpublished.

Crawford, S. L. (1989). Extensions to the CART algorithm. *International Journal of Man-Machine Studies, 31(2)*, 197–217.

Efron, B. & Tibshirani, R. (1997). Improvements on cross-validation: The .632+ bootstrap method. *Journal of the American Statistical Association, 92(438)*, 548–??

Efron, B. & Tibshirani, R. J. (1993). *An Introduction to the Bootstrap.* Chapman and Hall, London, UK.

Esposito, F., Malerba, D. & Semeraro, G. (1995). Simplifying decision trees by pruning and grafting: New results. In Lavrač, N. & Wrobel, S. (Eds.), *Proceedings of the 8th European Conference on Machine Learning*, Volume 912 of *LNAI* (pp. 287–290). Berlin: Springer.

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

Forsyth, R. (1994). Overfitting revisited: an information-theoretic approach to simplifying discrimination trees. *Journal of Experimental & Theoretical Aritificial Intelligence, 6(3)*, 289–302.

Frank, E. & Witten, I. H. (1998a). Generating accurate rule sets without global optimization. In Shavlik, J. (Ed.), *Proceedings of the Fifteenth International Conference, Morgan Kaufmann Publishers.* San Francisco, CA: Morgan Kaufmann.

Frank, E. & Witten, I. H. (1998b). Using a permutation test for attribute selection in decision trees. In Shavlik, J. (Ed.), *Proceedings of the Fifteenth International Conference, Morgan Kaufmann Publishers.* San Francisco, CA: Morgan Kaufmann.

Gaines, B. (1989). An ounce of knowledge is worth a ton of data. In *Proceedings of the 6th International Workshop on Machine Learning* (pp. 156–159). Morgan Kaufmann.

Gelfand, S. B., Ravishankar, C. S. & Delp, E. J. (1991). An iterative growing and pruning algorithm for classification tree design. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-13(2)*, 163–174.

Good, P. (1994). *Permutation Tests.* New York: Springer-Verlag.

Hong, S., Hosking, J. & Winograd, S. (1996). Use of randomization to normalize feature merits. In Dowe, D. L., Korb, K. B. & Oliver, J. (Eds.), *Information, Statistics and Induction in Science, Proceedings of the ISIS 96 conference* (pp. 10–19). Singapore: World Scientific.

Jensen, D. (1992). *Induction with Randomization Testing.* PhD thesis, Washington University, St. Louis, Missouri.

Jensen, D., Oates, T. & Cohen, P. R. (1997). Building simple models: A case study with decision trees. *Lecture Notes in Computer Science, 1280*, 211–??

Jensen, D. & Schmill, M. (1997). Adjusting for multiple comparisons in decision tree pruning. In Heckerman, D., Mannila, H., Pregibon, D. & Uthurusamy, R. (Eds.), *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97)* (p. 195). AAAI Press.

Kalkanis, G. (1993). The application of confidence interval error analysis to the design of decision tree classifiers. *Pattern Recognition Letters, 14*, 355–361.

Kalles, D. (1995). *Decision trees and domain knowledge in pattern recognition.* PhD thesis, Department of Computation, UMIST.

Kass, G. (1980). An exploratory technique for investigating large quantities of categorical data. *Applied Statistics, 29(2).*

Kearns, M. & Mansour, Y. (1998). A fast, bottom-up decision tree pruning algorithm with near-optimal generalization. In *Proc. 15th International Conf. on Machine Learning* (pp. 269–277). Morgan Kaufmann, San Francisco, CA.

Knoll, U., Nakhaeizadeh, G. & Tasend, B. (1994). Cost-sensitive pruning of decision trees. In Bergadano, F. & de Raedt, L. (Eds.), *Proceedings of the European Conference on Machine Learning*, Volume 784 of *LNAI* (pp. 383–386). Berlin: Springer.

Kohavi, R. (1995). *Wrappers for Performance Enhancements and Oblivious Decision Graphs.* PhD thesis, Stanford University, Department of Computer Science.

Lancaster, H. (1961). Significance tests in discrete distributions. *Journal of the American Statistical Association, 56*, 223–234.

Li, X. & Dubest, R. C. (1986). Tree classifier design with a permutation statistic. *Pattern Recognition, 19(3)*, 229–235.

Lock, R. (1991). A sequential approximation to a permutation test. *Communications in statistics: simulation and computation, 20(1)*, 341–363.

Mansour, Y. (1997). Pessimistic decision tree pruning based on tree size. In *Proc. 14th International Conference on Machine Learning* (pp. 195–201). Morgan Kaufmann.

Martin, J. K. (1997). An exact probability metric for decision tree splitting and stopping. *Machine Learning, 28(2,3)*, 257–291.

Mehta, M., Rissanen, J. & Agrawal, R. (1995). MDL-based decision tree pruning. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining* (pp. 216–221). AAAI Press.

Mingers, J. (1987). Expert systems—rule induction with statistical data. *Journal of the Operational Research Society, 38*, 39–47.

Mingers, J. (1989). An empirical comparison of pruning methods for decision tree induction. *Machine Learning, 4(2)*, 227-243.

Murthy, S. K. (1998). Automatic construction of decision trees from data: A multi-disciplinary survey. *Data mining and knowledge discovery journal, 2(4).*

Niblett, T. & Bratko, I. (1986). Learning decision rules in noisy domains. In *Proc Expert Systems 86.* Cambridge: Cambridge University Press.

Oates, T. & Jensen, D. (1997). The effects of training set size on decision tree complexity. In Douglas H. Fisher, J. (Ed.), *Proc Fourteenth International Conference on Machine Learning* (pp. 254–262). San Francisco, CA: Morgan Kaufmann. Nashville, Tennesse.

Oates, T. & Jensen, D. (1998). Large datasets lead to overly complex models: an explanation and a solution. In *Proc Fourth International Conference on Knowledge Discovery and Data Mining* (pp. 294–298). AAAI Press. New York City, New York.

Oates, T. & Jensen, D. (1999). Toward a theoretical understanding of why and when decision tree pruning algorithms fail. In *Proc Sixteenth National Conference on Artificial Intelligence.* AAAI Press.

Patefield, W. (1981). As159 an efficient method of generating random $r \times c$ tables with given row and column totals. *Applied Statistics, 30(1),* 91–97.

Quinlan, J. (1987). Simplifying decision trees. *International Journal of Man-Machine Studies, 27,* 221–234.

Quinlan, J. (1992). *C4.5: Programs for Machine Learning.* Los Altos, CA: Morgan Kaufmann.

Quinlan, J. (1996). Improved use of continuous attribute in C4.5. *Journal of Artificial Intelligence Research, 4,* 77–90.

Quinlan, J. & Rivest, R. (1989). Inferring decision trees using the minimum description length principle. *Information and Computation, 80,* 227–248.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning, 1(1),* 81–106.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica, 14,* 465–471.

Safavian, S. R. & Landgrebe, D. (1991). A survey of decision tree classifier methodolgy. *IEEE Transactions on Systems, Man, and Cybernetics, 21(3),* 660–674.

Ting, K. M. (1998). Inducing cost-sensitive trees via instance weighting. In Żytkow, J. M. & Quafafou, M. (Eds.), *Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD-98),* Volume 1510 of *LNAI* (pp. 139–147). Berlin: Springer.

Utgoff, P. E., Berkman, N. C. & Clouse, J. A. (1997). Decision tree induction based on efficient tree restructuring. *Machine Learning, 29,* 5–44.

Vadera, S. & Nechab, S. (1994). Id3, its children and their safety. *BCS Specialist Group on Expert Systems Newsletter, 31,* 11–21.

Wallace, C. S. & Patrick, J. D. (1993). Coding decision trees. *Machine Learning, 11,* 7–22.

Weiss, S. M. & Indurkhya, N. (1994a). Decision tree pruning: Biased or optimal? In *Proceedings of the 12th National Conference on Artificial Intelligence. Volume 1* (pp. 626–632). Menlo Park, CA, USA: AAAI Press.

Weiss, S. M. & Indurkhya, N. (1994b). Small sample decision tree pruning. In *Proc. 11th International Conference on Machine Learning* (pp. 335–342). Morgan Kaufmann.

Westfall, P. & Young, S. (1993). *Resampling-based multiple testing: Examples and methods for p-value adjustement.* New York: Wiley.

Wild, C. & Weber, G. (1995). *Introduction to Probability and Statistics.* University of Auckland.

Appendix A

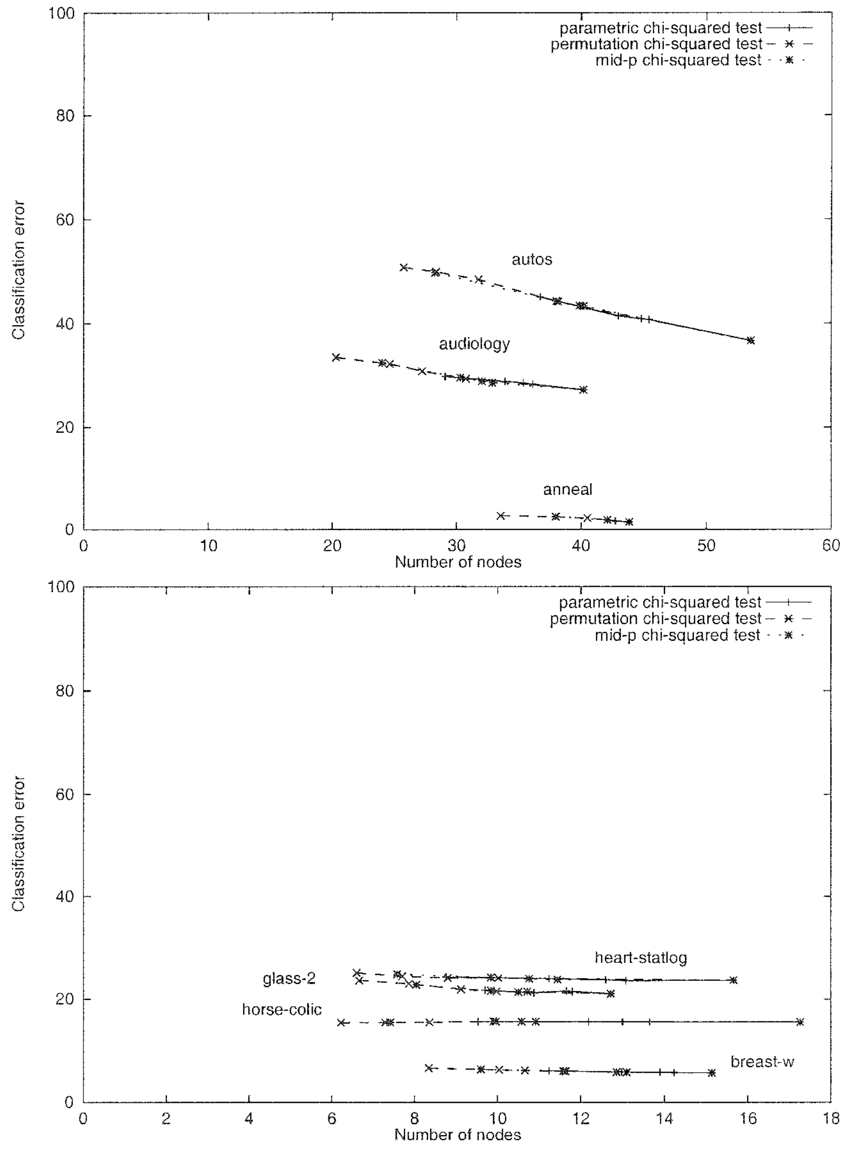
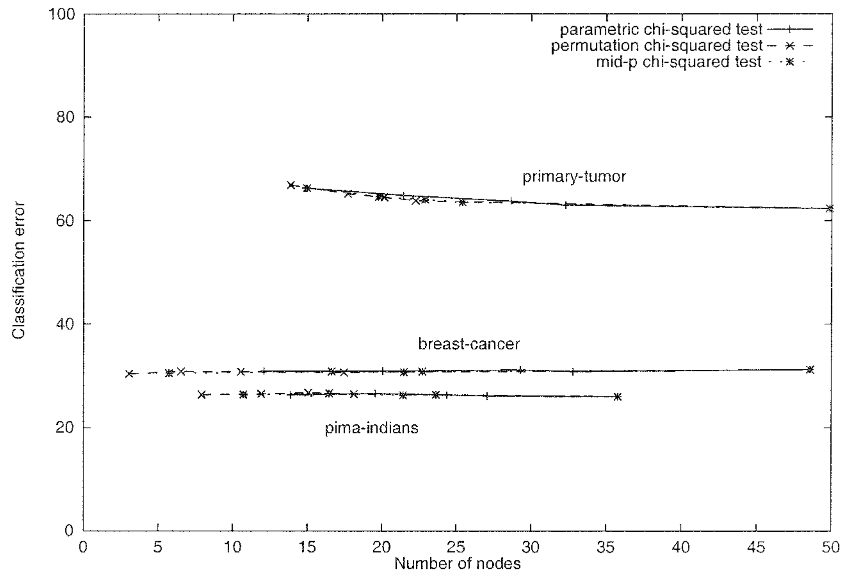


Figure A.1. Comparison of significance tests (a)



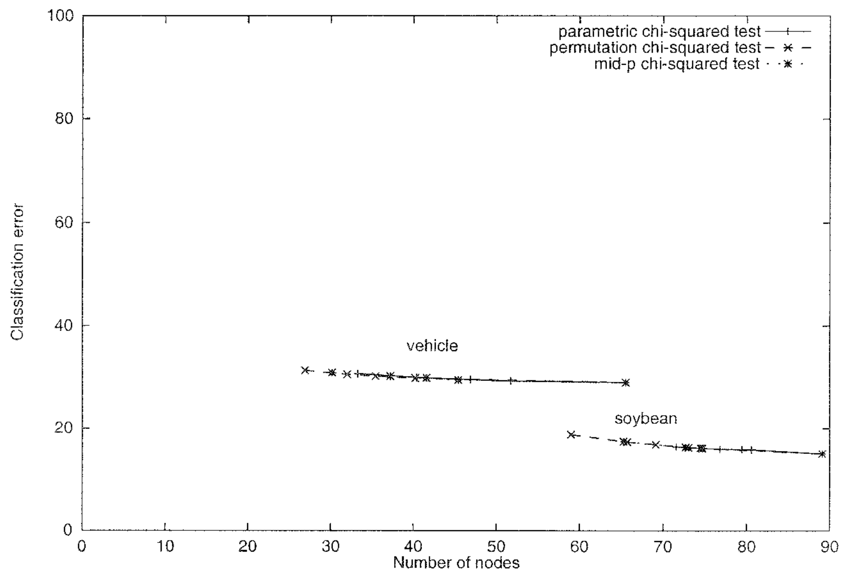
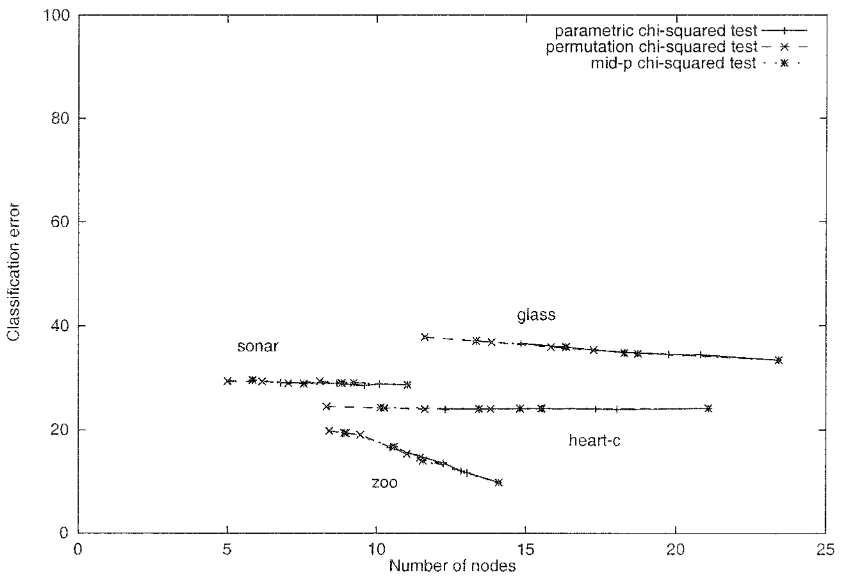


Figure A.2. Comparison of significance tests (b)



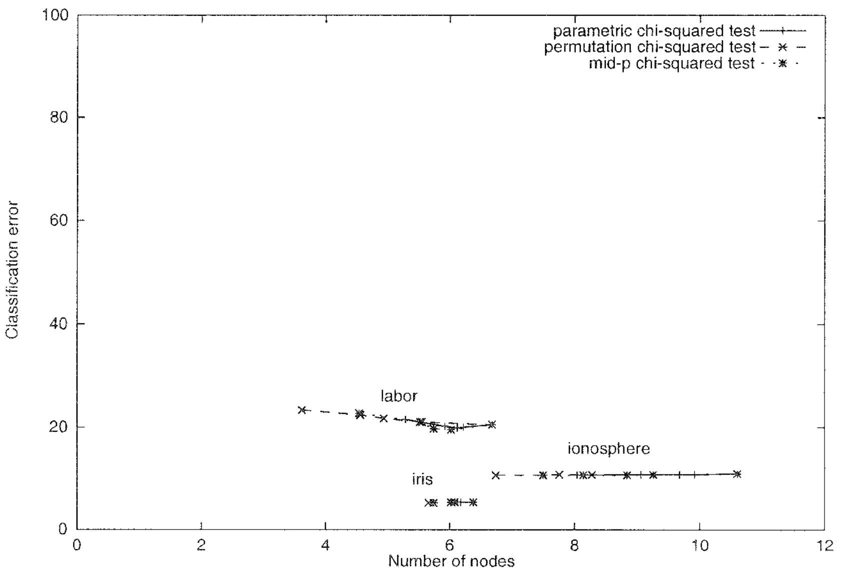
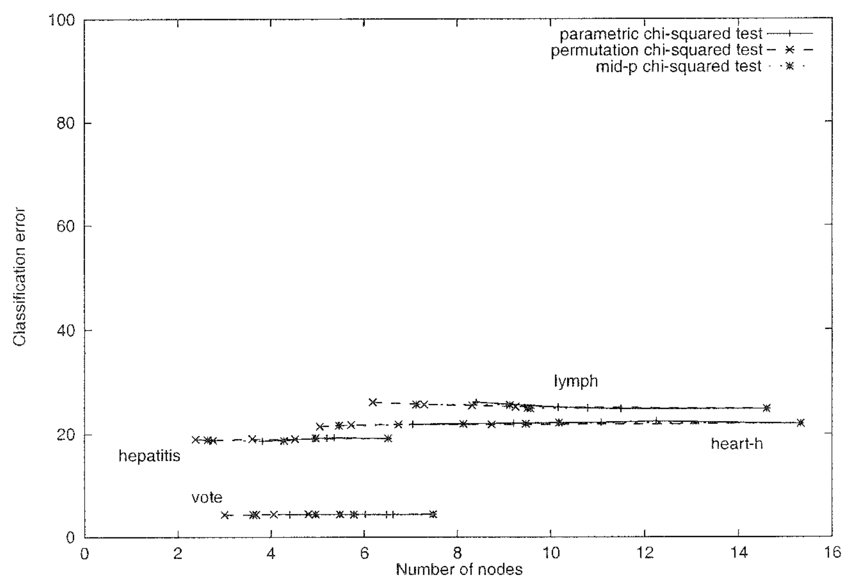


Figure A.3. Comparison of significance tests (c)



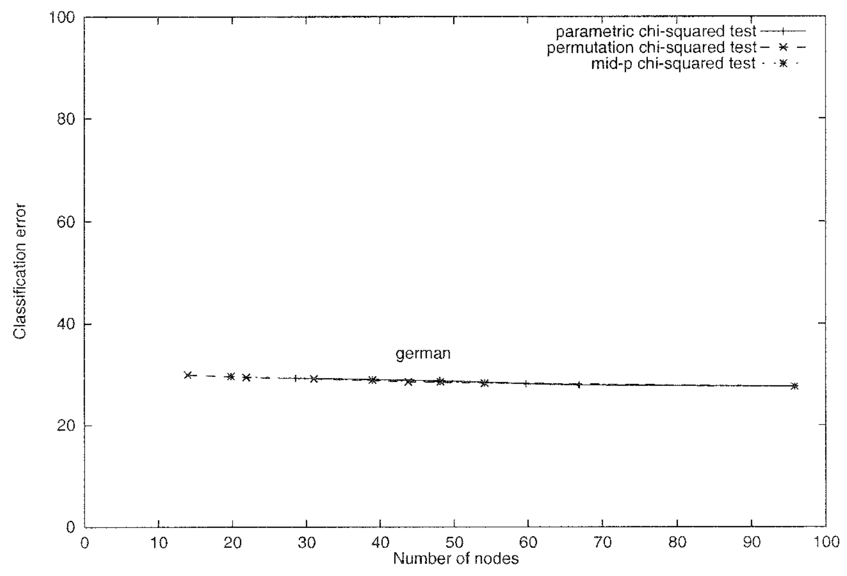


Figure A.4. Comparison of significance tests (d)