# Adjusting for Multiple Comparisons in Decision Tree Pruning

**David Jensen and Matt Schmill**
Department of Computer Science
University of Massachusetts
Amherst, MA 01003-4610
{jensen|schmill}@cs.umass.edu

## Abstract

Pruning is a common technique to avoid overfitting in decision trees. Most pruning techniques do not account for one important factor — multiple comparisons. Multiple comparisons occur when an induction algorithm examines several candidate models and selects the one that best accords with the data. Making multiple comparisons produces incorrect inferences about model accuracy. We examine a method that adjusts for multiple comparisons when pruning decision trees – Bonferroni pruning. In experiments with artificial and realistic datasets, Bonferroni pruning produces smaller trees that are at least as accurate as trees pruned using other common approaches.

## Introduction

Overfitting is a widely observed pathology of induction algorithms. Overfitted models contain unnecessary structure that reflects nothing more than random variation in the data sample used to construct the model. Portions of these models are wrong, and can mislead users. Overfitted models are less efficient to store and use than their correctly-sized counterparts. Finally, overfitting can reduce the accuracy of induced models on new data (Jensen 1992).

Decision tree algorithms (Breiman *et al.* 1984; Quinlan 1993) often use *pruning* to correct overfitting. Pruning examines individual subtrees and removes those subtrees deemed to be unnecessary. Pruning techniques primarily differ in the criterion used to judge subtrees. Common criteria include statistical significance tests (Quinlan 1986), corrected error estimates (Quinlan 1993), and minimum description length calculations (Quinlan & Rivest 1989).

Most common pruning techniques do not adjust for multiple comparisons. Multiple comparisons occur when an induction algorithm examines several candidate models and selects the one with the maximum score. Nearly all induction algorithms implicitly or explicitly search a space of candidate models. Decision tree algorithms examine many possible subtrees and select the best one. Pruning should adjust for the number of subtrees examined, because multiple comparisons affect the distribution of the maximum score on training data (Cohen & Jensen 1997).

This paper examines one pruning method that adjusts for multiple comparisons — *Bonferroni pruning.* Bonferroni pruning adjusts the results of a statistical hypothesis test for the number of subtrees examined at a particular node of a decision tree.

Copyright ©1997, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

## Bonferroni Pruning

Pruning techniques have been based on several different criteria. Statistical significance (Quinlan 1986) has generally been rejected based on its empirical performance. Error-based criteria, such as the approach used in C4.5 (Quinlan 1993), estimate true error rates by deflating the accuracy of subtrees on the training data. Minimum Description Length (MDL) criteria (Quinlan & Rivest 1989) characterize both datasets and models by the number of bits needed to encode them. The best tree is the one with the smallest total "description length" for the data, that is, the smallest sum of model description and description of the exceptions to the model's predictions. None of these approaches is intended to adjust for multiple comparisons.

Only a few algorithms explicitly account for multiple comparisons (Jensen 1992; Kass 1980) and several researchers call attention to multiple comparisons when *evaluating* induction algorithms (Feelders & Verkooijen 1995; Gascuel & Caraux 1992). However, nearly all commonly-used tree-building algorithms fail to adjust for multiple comparisons.

Why should multiple comparisons affect decision tree pruning? Consider the simplest case of pruning — deciding between a leaf node and a subtree (consisting of a test node using the attribute $A$ and two leaf nodes). One way of deciding whether to retain the tree is to produce a *score* for the tree, denoted $x$. The score could be the classification accuracy alone, or some more complex function. For the tree to be retained, $x$ must exceed some threshold — either the score for the leaf or some higher threshold. Call this threshold the *critical value*, denoted $x_c$.

One way of setting $x_c$ employs a statistical hypothesis test: $x_c$ is based on the reference distribution for the score under the null hypothesis $H_0$ that $A$ is unrelated to the class variable. The score $x$ is an estimate of some true score $x_t$. Because $x$ is based on a single sample, it will vary somewhat from the true value. Because of this, there is some probability that $x \geq x_c$ even though $x_t < x_c$. Hypothesis tests determine this probability under the null hypothesis $H_0$. The probability that $x$ will exceed $x_c$ given that the null hypothesis — $p(x \geq x_c|H_0)$ — is commonly denoted $\alpha$. The value of $x_c$ is usually set such that $\alpha$ is fairly small (e.g., 10%).

This approach, and nearly every *ad hoc* approach to setting $x_c$, ignores an important aspect of how decision trees are built: the variable $A$ was selected from a pool of potential variables. *Each* variables' scores had an opportunity to exceed $x_c$. If the potential variables are independently distributed, then $\alpha \neq 0.10$. Instead:

$$\alpha_t = 1 - (1 - \alpha_i)^k \tag{1}$$

where $\alpha_t$ is the probability that *at least one of* the $k$ variables' scores will equal or exceed $x_c$, and $\alpha_i$ is the probability that a *particular* variable's score will equal or exceed $x_c$. If $\alpha_i$ is 0.10 and 30 variables are examined, then $\alpha_t = 0.96$. Rather than a 10% probability of incorrectly selecting the tree, such selection becomes nearly certain. In practice, induction algorithms evaluate dozens or even hundreds of variables when selecting a variable for a test node. As a result, adjusting for multiple comparisons becomes essential for an accurate pruning. Multiple comparisons affect all threshold-based pruning techniques, regardless of whether they explicitly employ statistical tests.

Equation 1 is one of a class of Bonferroni equations (Kotz & Johnson 1982), that can be used to adjust for multiple comparisons. Specifically, we can set $\alpha_t$ and solve Equation 1 for $\alpha_i$, so that, for a given and number of comparisons $k$, an appropriate critical value can be selected based on a reference distribution for a single comparison. The next two sections present both experiments with artificial and realistic data to evaluate the effect of this approach.

## Experiments with Artificial Data

This experiment applies a simple tree-building algorithm to an artificially-generated data set. Data are created by randomly generating instances with 30 uniformly, independently, and identically distributed binary attributes. A binary classification variable is created by duplicating the first attribute. The values of the classification are systematically corrupted by complementing each value with a probability of 0.1, producing a theoretical upper bound of 90% on classification accuracy. The theoretically correct tree has three nodes — a decision node and two leaf nodes.

The tree-building algorithm uses information gain (Quinlan 1993) to choose the best attribute for a particular decision node. A leaf node is created when no partition can improve accuracy on the training set. The class label of the leaf node is determined by the majority class of the training instances present at that node. Trees are pruned bottom-up by each of four techniques: 1) FISHERS — Fisher's exact test (Kotz & Johnson 1982) with $\alpha = 0.10$, which does not adjust for multiple comparisons; 2) ERROR-BASED — The technique used in C4.5 (Quinlan 1993); 3) MDL — Minimum description length, using Utgoff's formulation (Utgoff 1995); and 4) BONFERRONI — Fisher's exact test with $\alpha$ adjusted for the number of comparisons using the Bonferroni adjustment.

Training set size $N$ is varied from 5 to 250 by increments of 5 instances. For each value of $N$, 100 trials are conducted. Each trial generates a training set, induces a tree, prunes the tree with each technique, and measures accuracy of the pruned trees on a test set of 1000 freshly generated instances. The complexity of pruned trees is determined by counting the total number of nodes. Results are averaged over the 100 trials.

Results are shown in Figure 1. In general, BONFERRONI produces the least complex and most accurate trees. When $N < 20$, no technique reliably produces the correct tree. BONFERRONI is the most conservative, producing less complex and less accurate trees than other pruning techniques. For $20 < N < 30$, accuracy of all pruned trees is roughly the same, and the average accuracy of UNPRUNED trees is much lower. For $N > 40$, BONFERRONI achieves nearly maximum average accuracy, while other techniques have lower average accuracy. This lower accuracy is due entirely to overfitting. Recall that all pruning methods begin with the same trees. Because BONFERRONI produces pruned trees with nearly optimal accuracy, it is clear that the true structure is present in the original trees. For UNPRUNED, FISHERS, and MDL, complexity is strongly associated with training set size. A related paper (Oates & Jensen 1997) explores this behavior for a wide variety of realistic datasets and several pruning approaches.

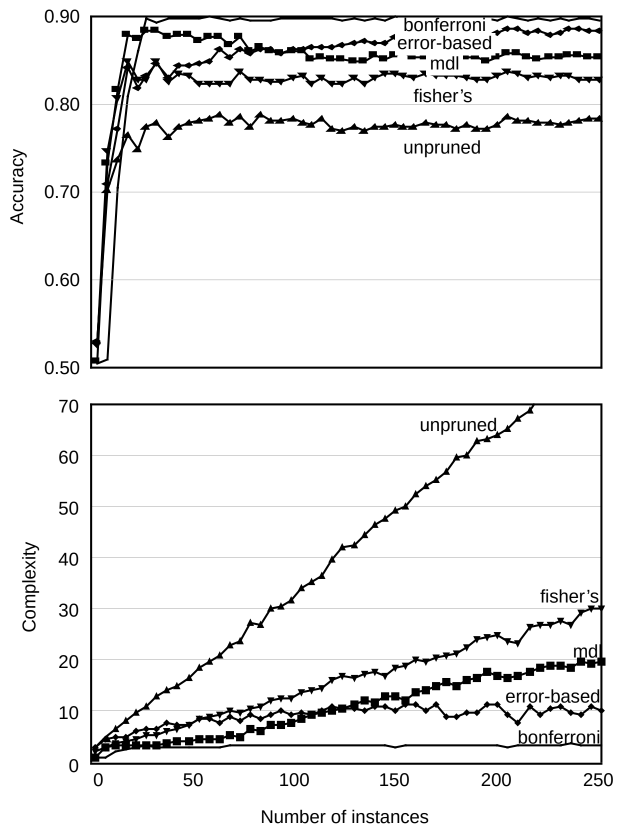

Figure 1: Accuracy and Complexity for Artificially-generated Data

## TBA: Tree-building with Bonferroni Adjustment

These experiments evaluate the performance of a new algorithm — Tree-building with Bonferroni Adjustment (TBA) — on 13 UCI datasets. TBA differs from other algorithms for top-down induction of decision trees (Quinlan 1986) in three important ways: its evaluation function, how it selects partitions during tree construction, and how it selects subtrees during tree pruning.

TBA uses the $G$ statistic to evaluate contingency tables during tree construction and pruning. The $G$ statistic is used because it has a known reference distribution and because it is additive, allowing portions of contingency tables to be evaluated individually.

During tree construction, TBA selects attributes using an approach suggested by Kass (Kass 1980) and Kerber (Kerber 1992). For each attribute, a contingency table is constructed with a row for each class value and a column for each of $a$ attribute values — every possible value for discrete attributes or every unique interval for discretized continuous attributes. Then, the pair of columns in the table with the least significant difference is merged. The merging process is repeated until all columns are significantly different. For continuous attributes, only adjacent columns, corresponding to adjacent numeric intervals, can be merged. The result is a node that partitions the sample into $b$ subsamples, where $1 \leq b \leq a$.

TBA selects attributes based on their probability values. These values are calculated by comparing the $G$ value for the merged contingency table to an appropriate reference distribution, and applying a Bonferroni adjustment. For a table with $r$ rows and $c$ columns, the appropriate reference distribution is chi-square with $(r-1)(c-1)$ degrees of freedom. Following Kass, the Bonferroni exponent $k$ is the number of possible ways of combining $a$ initial categories into $b$ final categories. TBA forms a decision node using the attribute with the lowest probability value, regardless of whether that value exceeds some threshold $\alpha$. The algorithm uses the decision node to partition the sample into $b$ subsamples based on the attribute's values, and repeats the attribute selection process for each subsample. Tree growth stops when no partition can improve accuracy on the training set.

TBA prunes the tree by examining the probability values calculated during tree construction. Recall that those probability values were adjusted for multiple comparisons *within* an attribute, but were not adjusted for multiple comparisons *among* the many attributes that could be used at an individual node. The latter adjustment is made at this stage, where the Bonferroni exponent $k$ is the number of attributes considered at that node.

TBA examines each frontier node of the tree — decision nodes that have only leaf nodes as children. Frontier nodes where $p \leq \alpha_t$ are retained; frontier nodes where $p > \alpha_t$ are converted to leaf nodes and labeled with the majority class of the appropriate training subsample. The process continues until all frontier nodes are significant. For all results reported here, TBA was run with $\alpha_t = 0.10$.

Table 1 shows the results of applying TBA, C4.5, and TBA-lesion (a version of TBA without Bonferroni adjustment) to thirteen UCI datasets. The experiments used ten-fold cross-validation with the folds held constant across the runs of all three algorithms. Italicized values are significantly different from the results of TBA. Bold values are significantly different from the results for TBA and are inconsistent with the hypothesis that TBA produces smaller trees that are at least as accurate as those of C4.5. Significance was judged using a two-tailed, paired t-test with $\alpha_t = 0.10$ and a Bonferroni correction to adjust for 13 independent

Table 1: Comparison of TBA with C4.5 and TBA-lesion

| | TBA | | C4.5 | | TBA-lesion | |
|---|---|---|---|---|---|---|
| Dataset | size | error | size | error | size | error |
| breast | 7 | 0.280 | 12 | 0.266 | *33* | 0.300 |
| cleveland | 8 | 0.254 | *46* | 0.252 | *23* | 0.236 |
| crx | 22 | 0.167 | *50* | 0.154 | *110* | 0.196 |
| glass | 14 | 0.342 | *51* | 0.289 | *32* | 0.297 |
| heart | 11 | 0.244 | *48* | 0.248 | *71* | 0.274 |
| hepatitis | 7 | 0.161 | *16* | 0.212 | *24* | 0.175 |
| hypothyr | 12 | 0.020 | 11 | **0.008** | *57* | 0.022 |
| iris | 4 | 0.047 | *9* | 0.067 | *8* | 0.060 |
| lymph | 4 | 0.261 | *26* | 0.211 | *20* | 0.243 |
| pima | 20 | 0.267 | *123* | 0.285 | *209* | 0.327 |
| votes | 6 | 0.046 | *14* | 0.055 | *27* | 0.062 |
| wine | 14 | 0.068 | **10** | 0.085 | *21* | 0.057 |
| wisconsin | 14 | 0.060 | 20 | 0.052 | *52* | 0.082 |

comparisons. This adjusts for comparisons within each column of the table, but not between columns. Thus, for each of the columns in the table, there is a 10% probability of incorrectly rejecting the null hypothesis for at least one of the datasets.

In nearly all cases, the results are consistent with the conjecture that TBA produces smaller trees than C4.5 without sacrificing accuracy. In nine of the thirteen datasets, TBA produced significantly smaller trees, in some cases dramatically smaller. TBA's trees are half the size, on average, of trees built by C4.5. In only one case was TBA's accuracy significantly lower than C4.5's. This dataset, `hypothyroid`, is the only set of the thirteen with substantial numbers of missing values. TBA uses a simple, but potentially inferior method of handling missing values. In all cases, TBA-lesion produces trees that are significantly larger, but not significantly more accurate, than trees produced by TBA.

## Acknowledgments

The authors would like to thank Paul Cohen, Tim Oates, Paul Utgoff, and two anonymous reviewers for their comments on earlier versions of this paper. M. Zwitter and M. Soklic of the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia provided the `breast` and `lymphography` datasets, and Dr. William H. Wolberg of the University of Wisconsin Hospitals provided the `wisconsin` dataset. This research is supported by DARPA/Rome Laboratory under contract No. F30602-93-C-0076. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements either expressed or implied, of the Defense Advanced Research Projects Agency, Rome Laboratory or the U.S. Government.

## References

Breiman, L.; Friedman, J.; Olshen, R.; and Stone, C. 1984. *Classification and Regression Trees.* Wadsworth and Brooks.

Cohen, P., and Jensen, D. 1997. Overfitting explained. In *Preliminary Papers of the Sixth International Workshop on Artificial Intelligence and Statistics.*

Feelders, A., and Verkooijen, W. 1995. Which method learns the most from data? methodological issues in the analysis of comparative studies. In *Preliminary Papers of the Fifth International Workshop on Artificial Intelligence and Statistics.*

Gascuel, O., and Caraux, G. 1992. Statistical significance in inductive learning. In *ECAI92: Proceedings of the 10th European Conference on Artificial Intelligence*, 435–439.

Jensen, D. 1992. *Induction with Randomization Testing: Decision-Oriented Analysis of Large Data Sets.* Ph.D. Dissertation, Washington University, St. Louis Missouri.

Kass, G. 1980. An exploratory technique for investigating large quantities of categorical data. *Applied Statistics* 29:199–127.

Kerber, R. 1992. Chimerge: Discretization of numeric attributes. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, 123–128.

Kotz, S., and Johnson, N., eds. 1982. *Encyclopedia of Statistical Sciences.* Wiley.

Oates, T., and Jensen, D. 1997. The effects of training set size on decision tree complexity. In *To appear in the Proceedings of the Fourteenth International Conference on Machine Learning.*

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

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

Quinlan, J. 1993. *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers.

Utgoff, P. 1995. Decision tree induction based on efficient tree restructuring. Technical Report 95-18, Department of Computer Science, University of Massachusetts, Amherst.