# Applying Permuation Tests to Tree-Based Statistical Models: Extending the R Package rpart

CS-TR-4581, UMIACS-TR-2004-24

Michael P. Cummings[1]*, Daniel S. Myers[1] and Marci Mangelson[2]

[1]Center for Bioinformatics and Computational Biology,
Institute for Advanced Computer Studies, University of Maryland, MD 20724
[2]Brigham Young University, Provo, UT 84602

**Abstract**

Tree-based statistical models are useful for evaluating relationships between predictor and response variables and for generating predictions when the response is unknown. However, current methods of constructing tree-based models do not provide a probabilistic assessment of the models produced. Here we describe our work to use permutation tests to quantitatively estimate the probability of tree-based statistical models. We have extended the rpart (recursive partitioning) package of the R system for statistical data analysis. This extension, rpart.permutation, executes the permutations in parallel, using MPI (Message Passing Interface) to greatly decrease the time necessary to complete the analysis.

*Author for correspondence: mike@umiacs.umd.edu

# Introduction

## Tree-Based Statistical Models

Numerous methods have been developed in the fields of statistics and machine learning to deal with supervised learning classification problems and regression problems. Tree-based statistical models [6], also referred to as Classification and Regression Trees (CART) [4] or as decision trees, are one such method. Tree-based statistical models operate by recursively creating bi-partitions of a given data set so as to best create homogeneous collections of a nominal or ordinal response variable (classification), or to best separate low and high values of a continuous response variable (regression). Each bi-partition may be considered as a question of the following form:

Is the observation $\mathbf{x}_i \in A$? Where $A$ is a region of the variable space.

Answering such a question for all observations produces two groups: those observations for which the answer is *yes* (those in region $A$) and those observations for which the answer is *no* ($\mathbf{x}_i \ni A$, those in the complement of $A$).

The recursive bi-partitioning process continues until stopping criteria (variously defined) are met. The result of this process is a classification or a regression tree, which is a hierarchical series of data bifurcations depicting the split definitions and describing the data subsets defined by each split.

Several criteria are used to choose among the possible bi-partitions in classification and regression contexts. Different criteria may result in slightly different trees for some data sets. The conceptual bases for specific criteria are often interrelated, and include concepts such as impurity, entropy/information, deviance/likelihood, least squares, and least average deviation. There are excellent expositions of split criteria, their mathematical foundations, and their implications [4, 6, 18].

Regardless of the chosen splitting criterion, at each level of recursive splitting, the search through possible bi-partitions is exhaustive. For each of $n$ distinct values of ordinal or continuous predictor variables, there are $n - 1$ possible partitions. For each unordered categorical variable (nominal variable) there are $2^{n-1} - 1$ possible partitions, which can be a very large number. For example, in an analysis of peptide binding to the Major Histocompatibility Complex (MHC) class I molecule there were as many as 20 amino acids observed at a site across the protein sequences examined [17]. This many levels results in $\sim 5.24 \times 10^5$ partitions to be evaluated for each of these single variables.

While the optimality of each individual bi-partition is assured through exhaustive evaluation of alternative splits, the global tree structure may not be optimal in terms of minimizing error over the whole tree. Most algorithms for tree-based statistical analysis (*i.e.*, those implemented in CART, S-PLUS, rpart, and other programs) are one-step optimal (greedy), meaning they look for the best split only at the node under consideration during the growing process, without consideration of the effect of these splits on further partitioning of the data. The alternative procedure (finding globally optimal trees) requires recursively evaluating every candidate split at each level of recursion. In the absence of an algorithmic shortcut, the resulting combinatorial explosion often renders this procedure prohibitively demanding from a computational perspective.

Although all current methods for generating tree-based statistical models of which we are aware provide evaluative-predictive models, none provide a probabilistic assessment of the models. Here we provide a means to quantitatively estimate the probability of tree-based statistical models.

## Permutation Testing

The null hypothesis we wish to test is that the relationship of predictor variables to the response variable is random (i.e., no causal or correlative association between predictor and response variable exists). An appropriate means to test this null hypothesis is through a permutation where we construct a data set corresponding to the null hypothesis by holding the predictor variables constant and permuting the response variables. This procedure randomizes the association between the two sets of variables. Analyzing the permuted data, we obtain a model for a random instance of the data. By generating many such permuted datasets and determining a model for each, we can determine the frequency of models equal to or better than that which we observed from the original data. This frequency is our $P$-value, the

probability of observing a model equal to or better than that which we observed by chance alone.

The $P$-value for a permutation test is calculated from the sorted distribution of results from analysis of the permuted data. Specifically, let us conduct $n$ permutations of the original model as described above. Let $b$ denote the number of times that the generated models attained an accuracy better than or equal to the accuracy of the original model. The $P$-value for the hypothesis test is thus $(b+1)/(n+1)$. The 1 added to the numerator and denominator is for the original model.

There is a wide range of applications for permutation testing because it is based on the empirical observations at hand, and thus subsumes any idiosyncrasies embodied by the original data. Permutation tests require no assumptions about underlying distributional model of either the data or the statistics examined. Furthermore permutation tests are statistically valid for the regression and classification problems to which they are applied here through rpart. Another attractive feature of permutation tests is statistical power, which is usually equal to the most powerful parametric alternatives where these can be applied [2]. Permutation tests are also exact in that the $P$-values estimated are accurate with precision determined by the number of permutations evaluated [11, 13, 14]. Permutation tests have been applied in the field of genetics to determine the probability of association between genetic markers and quantitative trait loci (QTLs) [5, 9].

## Adding a permutation test to tree-based statistical models

We implemented permutation testing of tree-based statistical models in the R statistical language [12]. R has an excellent package for tree-based statistical models, rpart (recursive partitioning) [18]. Our package is an an extension of rpart, and is called rpart.permutation. In rpart.permutation we provide a function that accepts a model object as returned by rpart and performs permutation testing as described above. It augments the complexity parameter table of the rpart object with three new columns: a $P$-value for the relative error, a $P$-value for the cross-validation error, and the number of permutations used to generate those $P$-values (see below).

There is one implementation detail that should be noted. When rpart returns a tree, it returns error values for several possible depths of that tree (numbers of splits). For example, a call to rpart might return a tree with 7 splits, and rpart might provide error values for the 0, 1, 5, 6, and 7-split subtrees. The specific number of splits at which subtree error information is returned, however, can vary from dataset to dataset. In particular, when a dataset is permuted subtree error information for each subtree present in the original model may not be present in the complexity parameter table for every permutation, because of stochasticity associated with the permutation. Therefore, it may be necessary to conduct more than $n$ permutations in order to ensure that every subtree in the original model has at least $n$ associated new models. Moreover, this means that some subtrees may have more than $n$ associated new models. To make this process somewhat more transparent, we report the exact number of permutations associated with each $P$-value to the user.

## Performance evaluation of parallel implementation in rpart

Given that one may wish to conduct a large number of permutations (e.g., $10^4$) for a single test, and that such a large number of permutations can take a long time to compute, we have included support to parallelize the processing across distributed memory platforms using the Rmpi [20] and snow [19] R packages. (Rmpi provides MPI bindings for R, and snow uses those bindings to provide a high-level interface to a network of workstations.) As each individual permutation in a permutation test is wholly independent of all other permutations, permutation testing constitutes a parallel computing opportunity that is well-suited to distributed memory computing environments. We have also used both the Rmpi and snow R packages to create a parallel implementation of the randomForest R package [15].

We conducted performance testing of our parallel implementation on a local computing cluster using an unpublished dataset from our own research. Performance testing on a computing cluster showed reasonable results, although with a deviation from perfect linear speedup (see Figure 1). This deviation is due to the subtree-error-information problem described above. Each node in the parallel execution is assigned a certain number of permutations to compute for all subtrees in the original model

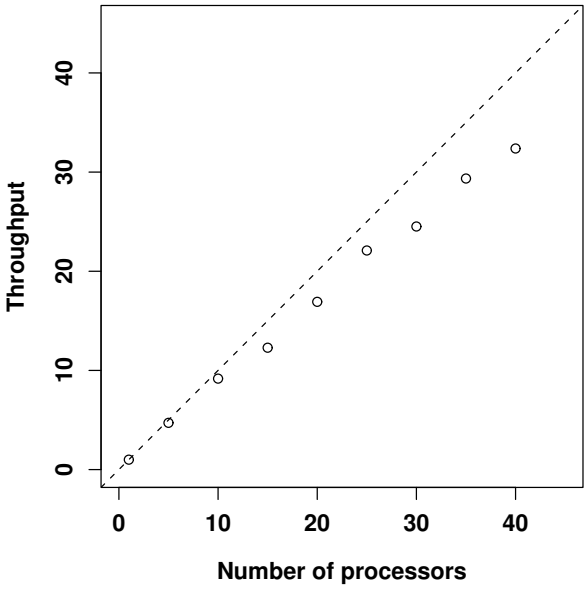


Figure 1: Plot showing the relationship of the number of processors and throughput (number of permutations / unit time).

(roughly number of permutations / number of processors). Because of the subtree-error problem, however, subtree-error values may not distribute uniformly across computing nodes. As an example, suppose we have two computing nodes, and we want 100 permutations for the one- and two-split subtrees of a two-split tree. At some point in the computation, node one may have 40 permutations of the one-split tree and 60-permutations of the two-split tree, while node two may have 60 permutations of the one-split tree and 40-permutations of the two-split tree. Overall, there are enough permutations of both trees, but the nodes are not aware of this fact and will keep computing until they have at least 50 each split, which results in an inefficiency. This problem could be corrected by a more intelligent load-balancing strategy.

# Examples

In this section we show several examples of the application of permutation tests to tree-based statistical models. We begin by permutation testing a classification tree built on the famous *Iris* dataset

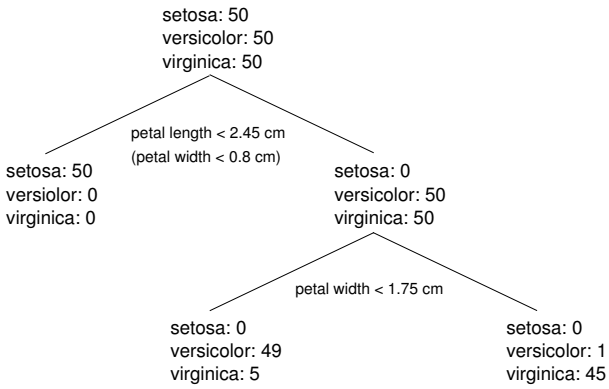


Figure 2: Tree-based statistical model for the *Iris* data. The number of observations for each species (*Iris setosa, I. versicolor,* and *I. virginica*) is given at each node. The definitions are given for each split (with equivalent definitions in parentheses).

```
data(iris)
fit <- rpart(Species~., data=iris, cp=0)
print(fit$cptable)
    CP nsplit rel error xerror       xstd
1 0.50      0      1.00   1.21 0.04836666
2 0.44      1      0.50   0.66 0.06079474
3 0.00      2      0.06   0.08 0.02751969
```

Figure 3: Summary of the rpart results for the *Iris* data. Column headings in the complexity parameter table: CP, cost-complexity; nsplit, number of splits (partitions); rel error, relative error; xerror, error estimate based on cross-validation; xstd, standard deviation of xerror.

[1, 10]. The four predictor variables are measurements (in centimeters) for sepal length, sepal width, petal length, and petal width. The response variable is species designation (*Iris setosa, I. versicolor,* and *I. virginica*). The problem is to construct a model that accurately classifies the 150 observations into species. Figure 2 shows the tree-based statistical model for these data.

First, for comparative purposes, we present the standard complexity parameter table output from rpart (Figure 3).

Next, the results from a model using $10^4$ permutations and distributing the work over 10 processors are given in Figure 4. Note that that the $P$-values for all splits in the model indicate statistically sig-

nificant relationships between the predictor variables and the response variable.

It is instructive to contrast these results with those when no significant relationship exists between predictor variables and response variable. To provide such an example, we take the *Iris* data and eliminate the original relationships between the predictor variables and the response variable by randomization. As expected, the $P$-values for all splits in the model are not significant (Figure 5).

For our final example, we use glass identification data from the University of California, Irvine Machine Learning Repository [3]. The nine predictor variables consist of one physical property measurement (refractive index), and content measurements for eight elements (sodium, magnesium, aluminum, silicon, potassium, calcium, barium, iron). The problem is to construct a model that accurately classifies the 214 observations into seven different types of glass.

Here again the $P$-values for all splits in the model indicate statistically significant relationships between the predictor variables and the response variable (Figure 6). Also note that the total number of permutations required to achieve a minimum of $10^4$ permutations evaluated for each model split was $> 22,824$.

# Conclusions

Permutation tests can be appropriately applied to tree-based statistical models and provide an estimate of the probability associated with such models. We have developed a package in R, permutation.rpart, for executing permutation testing on rpart model objects using parallel computing. The permutation test for tree-based statistical models described here has been applied effectively in several studies in including an examination of cytidine-to-uridine editing of RNA in plant mitochondria [7] and resistance to the antibiotic rifampin in *Mycobacterium tuberculosis* [8].

# Program Availability

Our extension to the R package rpart, rpart.permutation, provides a convenient means to perform these permutation tests and is available through the Comprehensive R Archive Network (CRAN) [16].

# Acknowledgments

We thank Adam Bazinet and Maile Neel for comments on the draft versions of this document, and Kurt Hornik for help properly packaging the release of our extension for CRAN (Comprehensive R Archive Network).

# References

[1] E. Anderson. The irises of the Gaspe Peninsula. *Bulletin of the American Iris Society*, 59:2–5, 1935.

[2] P. J. Bickel and W. R. Van Zwet. Asymptotic expansion for the power of distribution free tests in the two-sample problem. *Annal. Statist.*, 6:987–1007, 1978.

[3] C. L. Blake and C. J. Merz. UCI repository of machine learning databases. University of California, Irvine, Deptartment of Information and Computer Sciences, 1998. http://www.ics.uci.edu/∼mlearn/MLRepository.html.

[4] L. Breiman, J. H. Friedman., R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Pacific Grove, CA, 1984.

[5] G. A. Churchill and R. W. Doerge. Empirical threshold values for quantitative trait mapping. *Genetics*, 138:963–971, 1994.

[6] L. A. Clark and D. Pergibon. *Statistical Models in S*. Chapman and Hall, London, 1993.

[7] M. P. Cummings and D. S. Myers. Simple statistical models predict C-to-U edited sites in plant mitochondrial RNA. *BMC Bioinformatics*, 5:132, 2004.

[8] M. P. Cummings and M. R. Segal. Few amino acid positions in *rpoB* are associated with most of the rifampin resistance in *Mycobacterium tuberculosis*. *BMC Bioinformatics*, accepted.

```
data(iris);
fit <- rpart(Species~., data=iris, cp=0);
fit <- doPermutationTest(iris, 5, fit, Species~., 10000, 10);
print(fit$cptable);
    CP nsplit rel error xerror      xstd Rpvalue Xpvalue nreps
1 0.50      0      1.00   1.17 0.0507346      NA      NA    NA
2 0.44      1      0.50   0.61 0.0601609       0       0 14334
3 0.00      2      0.06   0.11 0.0319270       0       0 10000
```

Figure 4: Summary of the rpart.permutation results for the *Iris* data showing $P$-values for each split in the tree model. Column headings in complexity parameter table are the same as in Figure 3, with some additions: Rpvalue, $P$-value for relative error; Xpvalue, $P$-value for error estimate based on cross-validation; nreps, number of permutations used to determine $P$-value.

```
data(iris);
iris[,5] <- sample(iris[,5]);
fit <- rpart(Species~., data=iris, cp=0);
fit <- doPermutationTest(iris, 5, fit, Species~., 10000, 10);
printcp(fit)
        CP nsplit rel error xerror     xstd Rpvalue Xpvalue nreps
1 0.080000      0      1.00   1.13 0.052795      NA      NA    NA
2 0.033333      1      0.92   1.08 0.054991 0.86292 0.42292 21418
3 0.020000      7      0.71   1.11 0.053722 0.82960 0.90200 10000
4 0.010000      9      0.67   1.08 0.054991 0.73181 0.82621 11462
5 0.000000     10      0.66   1.07 0.055384 0.72490 0.78418 10796
```

Figure 5: Summary of the rpart.permutation results for the randomized *Iris* data showing $P$-values for each split in the tree model. Column headings in the complexity parameter table are the same as in Figure 4.

```
glass <- read.table('data/glass.data.noid', header=T, sep=',')
fit <- rpart(type ~., data=glass, cp=0);
fit <- doPermutationTest(glass, 10, fit, type ~ ., 10000, 10);
printcp(fit);
           CP nsplit rel error  xerror     xstd Rpvalue Xpvalue nreps
1  0.60259194      0   1.00000 1.01293 0.090181      NA      NA    NA
2  0.12644549      1   0.39741 0.41587 0.064638       0       0 11960
3  0.03361436      2   0.27096 0.35616 0.064941       0       0 10144
4  0.02193767      3   0.23735 0.37248 0.066945       0       0 10036
5  0.01677633      4   0.21541 0.34312 0.063244       0       0 10396
6  0.00939699      5   0.19863 0.31991 0.057153       0       0 11126
7  0.00639936      6   0.18924 0.32062 0.056851       0       0 12034
8  0.00512931      7   0.18284 0.31600 0.053265       0       0 12850
9  0.00502811      8   0.17771 0.31834 0.053366       0       0 14004
10 0.00363930     11   0.16262 0.32041 0.054418       0       0 17480
11 0.00226887     13   0.15535 0.32032 0.054558       0       0 20652
12 0.00181853     14   0.15308 0.32402 0.056124       0       0 22076
13 0.00044297     15   0.15126 0.32416 0.056126       0       0 23140
14 0.00027278     16   0.15082 0.32486 0.056113       0       0 22824
15 0.00000000     17   0.15054 0.32526 0.056110       0       0 18280
```

Figure 6: Summary of the rpart.permutation results for the glass identification data showing $P$-values for each split in the tree model. Column headings in the complexity parameter table are the same as in Figure 4.


[9] R. W. Doerge and G. A. Churchill. Permutation tests for multiple loci affecting a quantitative character. *Genetics*, 142:285–294, 1996.

[10] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Ann. Eugenic.*, 7:179–188, 1936.

[11] P. Good. *Permutation Tests: A Practical Guide to Resampling Methods for Testing Hypotheses*. Springer-Verlag, New York, 1994.

[12] R. Ihaka and R. Gentleman. R: a language for data analysis and graphics. *Comput Graph Stat*, 5:299–314, 1996.

[13] B. F. J. Manly. *Randomization and Monte Carlo Methods in Biology*. Chapman & Hall, London, 1991.

[14] J. S. Maritz. *Distribution-free Statistical Methods*, volume 17 of *Monographs on Statistics and Applied Probability*. Chapman & Hall, London, second edition, 1995.

[15] D. S. Myers and M. P. Cummings. Parallel implementation of the machine learning/statistical method random forest (R package randomForest). Technical Report CS-TR-4584, UMIACS-TR-2004-27, Center for Bioinformatics and Computational Biology, Institute for Advanced Computer Studies, University of Maryland, 2004.

[16] Comprehensive R Archive Network. http://www.r-project.org.

[17] M. R. Segal, M. P. Cummings, and A. E. Hubbard. Relating genotype to phenotype: analysis of peptide binding data. *Biometrics*, 57:632–643, 2001.

[18] T. M. Therneau and E. J. Atkinson. An introduction to recursive partitioning using the RPART routines. Technical Report Mayo Foundation, 1997.

[19] L. Tierney, A. J. Rossini, and N. Li. The snow package: Simple network of workstations, 2003. http://cran.r-project.org/src/contrib/PACKAGES.html#snow.

[20] H. Yu. Rmpi package for R, 2003. http://www.stats.uwo.ca/faculty/yu/Rmpi/.