The Annals of Applied Statistics
2009, Vol. 3, No. 4, 1710–1737
DOI: 10.1214/09-AOAS260

IMPROVING THE PRECISION OF CLASSIFICATION TREES[1]

BY WEI-YIN LOH

University of Wisconsin

Besides serving as prediction models, classification trees are useful for finding important predictor variables and identifying interesting subgroups in the data. These functions can be compromised by weak split selection algorithms that have variable selection biases or that fail to search beyond local main effects at each node of the tree. The resulting models may include many irrelevant variables or select too few of the important ones. Either eventuality can lead to erroneous conclusions. Four techniques to improve the precision of the models are proposed and their effectiveness compared with that of other algorithms, including tree ensembles, on real and simulated data sets.

1. Introduction. Since the appearance of the AID and THAID algorithms [27, 28], classification trees have offered a unique way to model and visualize multi-dimensional data. As such, they are more intuitive to interpret than models that can be described only with mathematical equations. Interpretability, however, ensures neither predictive accuracy nor model parsimony. Predictive accuracy is the probability that a model correctly classifies an independent observation not used for model construction. Parsimony is always desirable in modeling—see, for example, McCullagh and Nelder [26], page 7—but it takes on increased importance here. A tree that involves irrelevant variables is not only more cumbersome to interpret but also potentially misleading.

Many of the early classification tree algorithms, including THAID, CART [5] and C4.5 [31], search exhaustively for a split of a node by minimizing a measure of node heterogeneity. As a result, if all other things are equal, variables that take more values have a greater chance to be chosen. This selection bias can produce overly large or overly small tree structures that obscure the importance of the variables. Doyle [9] seems to be the first to raise this issue, but solutions have begun to appear only in the last decade or so. The QUEST [24] and CRUISE [16] algorithms avoid the bias by first using F and chi-squared tests at each node to select the variable to split on. CTree [15] uses permutation tests. Other approaches are proposed in [18, 29] and [33].

Received August 2008; revised May 2009.

[1]This material is based upon work partially supported by U.S. Army Research Office Grants W911NF-05-1-0047 and W911NF-09-1-0205.

Key words and phrases. Bagging, kernel density, discrimination, nearest neighbor, prediction, random forest, selection bias, variable selection.

TABLE 1
Variables in mammography data

Name	Description	Values
ME	Mammography experience	1 (within 1 year), 2 (more than 1 year), 3 (never)
SYMP	You do not need a mammogram unless you develop symptoms	1 (strongly agree), 2 (agree), 3 (disagree), 4 (strongly disagree)
PB	Perceived benefit of mammography	5–20 (lower values indicate greater benefit)
HIST	Mother or sister with history of breast cancer	0 (no), 1 (yes)
BSE	Has anyone taught you how to examine your own breasts?	0 (no), 1 (yes)
DETC	How likely is it that a mammogram could find a new case of breast cancer?	1 (not likely), 2 (somewhat likely), 3 (very likely)

Unbiasedness alone, however, guarantees neither predictive accuracy nor variable selection efficiency. To see this, consider some data on the mammography experience (ME) of 412 women from Hosmer and Lemeshow [14], pages 264–287. The class variable is ME, with 104 women having had a mammogram within the last year ($\texttt{ME} = 1$), 74 having had one more than a year ago ($\texttt{ME} = 2$), and 234 women not having had any ($\texttt{ME} = 3$). [The ME codes here are different from the source; they are chosen to reflect a natural ordering.] Table 1 lists the five predictor variables and the values they take. Hosmer and Lemeshow fitted a polytomous logistic regression model for predicting ME that includes all five predictor variables, but with SYMP and DETC in the form of indicator variables $I(\texttt{SYMP} \leq 2)$ and $I(\texttt{DETC} = 3)$.

Let $C(i|j)$ denote the cost of misclassifying as class i an observation whose true class is j. Because the ME values are ordered, we set $C(i|j) = |i - j|$ for $i, j = 1, 2, 3$. Figure 1 shows the QUEST and CRUISE models. Each leaf node is colored white, light gray or dark gray as the predicted value of ME is 1, 2 or 3, respectively. The QUEST tree is too short; it splits only once and does not predict class 1. Note that class 1 constitutes less than 18% of the sample and that the Hosmer–Lemeshow model [14], page 277, Table 8.10, does not predict class 1 too. Another reason that the QUEST tree is shorter than the CRUISE tree is because the latter tests for pairwise interactions at each node, whereas the former does not. Thus, CRUISE can uncover more structure than QUEST.

Figure 2 shows the corresponding RPART [2] and J48 [35] trees. RPART is an implementation of CART in R and J48 is an implementation of C4.5 in JAVA. The two trees have much more structure, but the J48 tree reminds us that the comprehensibility of a tree structure diminishes with increase in its size. Further, there is a hint of over-fitting in the repeated and nonmonotonic splits on PB at the bottom

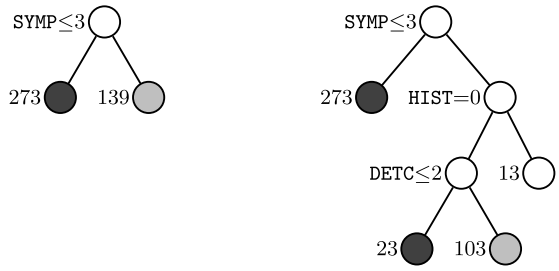

FIG. 1. *QUEST* (*left*) *and CRUISE* (*right*) *classification trees for mammography data. At each intermediate node, an observation goes to the left branch if and only if the condition shown there is satisfied. Leaf nodes classified as class* 1, 2 *and* 3 *are colored white, light gray, and dark gray, respectively. The number on the left of each leaf node is the sample size.*

of the J48 tree. It is difficult to tell which tree model has higher or lower predictive accuracy than the Hosmer and Lemeshow model. Empirical studies have shown that the predictive accuracy of logistic regression is often as good as that of classification trees for small sample sizes [22] but that C4.5 is more accurate as the

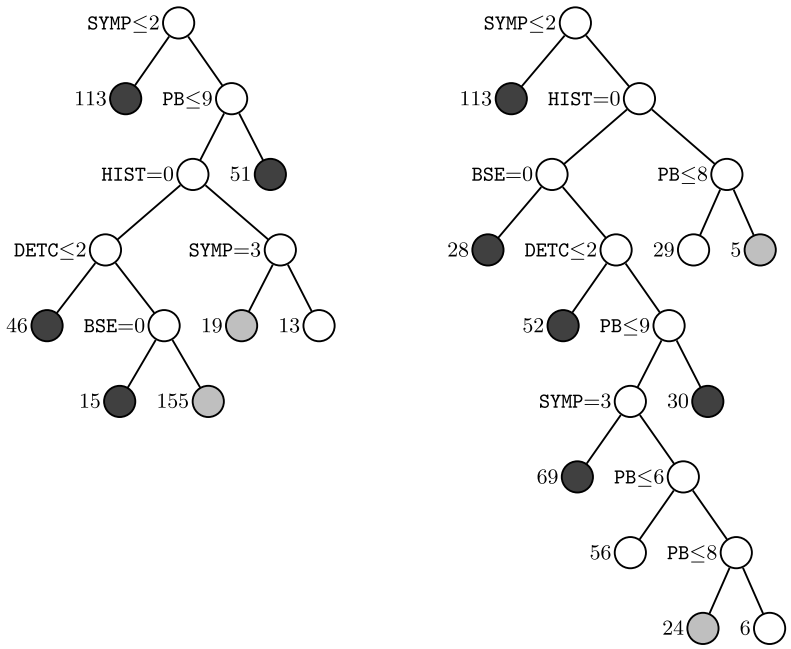

FIG. 2. *RPART* (*left*) *and J*48 (*right*) *classification trees for mammography data. At each intermediate node, an observation goes to the left branch if and only if the condition shown there is satisfied. Leaf nodes classified as class* 1, 2 *and* 3 *are colored white, light gray and dark gray, respectively. The number on the left of each leaf node is the sample size.*

sample size increases [30]. (Note: all the trees discussed in this article are pruned according to their respective algorithms. Trees constructed by the new methods to be described are pruned using CART's cost-complexity method with ten-fold cross-validation.)

Our goal here is to introduce and examine four techniques for constructing tree models that are parsimonious and have high predictive accuracy. We do this by controlling the search for local interactions, employing more effective variable and split selection strategies, and fitting nontrivial models in the nodes. The first technique limits the frequency of interaction tests, performing them only if no main effect test is significant. Besides saving computation time, this reduces the chance of selecting variables of lesser importance. The second technique utilizes a two-level split search when a significant interaction is found. This allows greater advantage to be taken of the information gained from interaction tests. The third technique considers linear splits on pairs of variables, if no main effect or interaction test is significant at a node. This can sometimes produce large gains in predictive accuracy as well as reduction in tree size. The fourth technique fits a nearest-neighbor or a kernel discriminant model, using one or two variables, at each node. This is useful in applications where neither univariate nor linear splits are effective. The result of applying the first three techniques to the mammography data is given in Figure 3. Its model complexity is in between that of CRUISE and QUEST on one hand, and that of RPART and C4.5 on the other.

The rest of this article is organized as follows. Section 2 discusses why and how we control the search for interactions and illustrates the solution with an artificial example. Section 3 introduces linear splits on pairs of variables and motivates the solution with a real example. Section 4 presents simulation results to show that the selection bias of our method is well controlled. Section 5 describes the use of kernel and nearest-neighbor models on pairs of variables to fit the data in each node

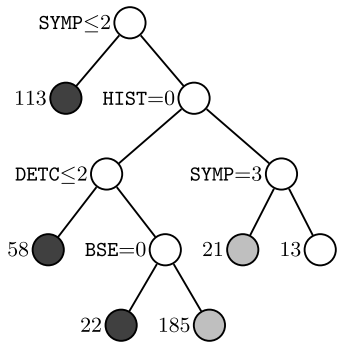

FIG. 3. *Tree for mammography data using the S method. At each intermediate node, an observation goes to the left branch if and only if the condition shown there is satisfied. Leaf nodes classified as class* 1, 2 *and* 3 *are colored white, light gray and dark gray, respectively. The number on the left of each leaf node is the sample size.*

and demonstrates their effectiveness with yet another example. Section 6 compares the predictive accuracy, tree size and execution time of the algorithms on forty-six data sets. Section 7 examines the effect of using tree ensembles and compares the results with Random Forest [4]. Section 8 concludes the discussion.

2. Controlled search for local interactions. The brevity of the QUEST tree in Figure 1 is attributable to its split selection strategy. At each node, QUEST evaluates the within-node (i.e., local) main effect of each predictor variable by computing an ANOVA p-value for each noncategorical variable and a Pearson chi-squared p-value for each categorical variable. Then it selects the variable with the smallest p-value to split the node. Although successful in avoiding selection bias, QUEST is insensitive to (local) interaction effects within the node. If the latter are strong and local main effects weak, the algorithm can select the wrong variable. The weakness does not necessarily lead to reduced predictive accuracy, however, because good splits may be found farther down the tree—this explains why algorithms with selection biases can still yield models with good predictive accuracy.

CRUISE searches over a larger number of splits at each node by including tests of interactions between all pairs of predictors. If there are K predictor variables, CRUISE computes K main effect p-values and $K(K-1)/2$ interaction p-values and splits the node on the variable associated with the smallest p-value. Because there are usually more interaction tests than main effect tests, the smallest p-value often comes from the former. As a result, CRUISE may select a variable with a weak main effect even though there are other variables with stronger ones. Further, if the most significant p-value is from an interaction between two variables, CRUISE chooses the variable with the smaller main effect p-value and then searches for a split on that variable alone.

To see that this can create difficulties, consider an extreme example where there are two classes and two predictor variables, X_1 and X_2, distributed on a square as in Figure 4. The square is divided equally into four sub-squares with one class located in the two sub-squares on one diagonal and the other class in the other two sub-squares. The optimal classification rule first splits the space into two equal halves at the origin using either X_1 or X_2 and then splits each half into two at the origin of the other variable. Since this requires a two-level search, CRUISE will likely require many more splits to accurately classify the data.

The above problems can be solved by making two adjustments. First, to prevent the interaction tests from overwhelming the main effect tests, we carry out the former only when no main effect test achieves a specified level of significance. Second, we perform a two-level search for the split points whenever a significant interaction is found. These steps are given in detail below, after the introduction of some necessary notation.

Let J denote the number of classes in the training sample and J_t the number of classes in node t. Let N_j denote the number of class j cases in the training

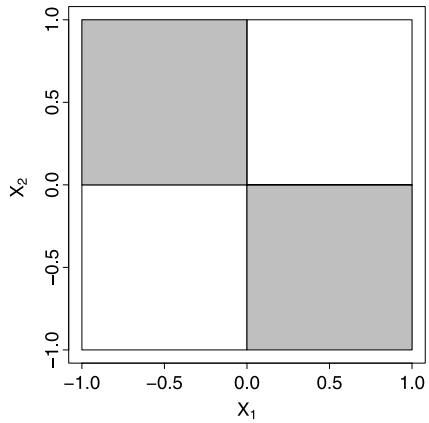

FIG. 4. *Class* 1 *observations uniformly distributed in the white squares and class* 2 *observations uniformly distributed in the gray squares.*

sample and $N = \sum_j N_j$. Let $N_j(t)$ and $N(t)$ denote the corresponding sample sizes in t, and let $\pi(j)$ be the prior probability for class j. The value of $\pi(j)$ may be specified by the user or it can be estimated from the data, in which case it is N_j/N. The estimated probability that a class j observation will land in node t is $p(j,t) = \pi(j)N_j(t)/N_j$. Define $p(t) = \sum_j p(j,t)$ and $p(j|t) = p(j,t)/p(t)$. The Gini impurity at t is defined as $g(t) = \sum_{i \neq j} p(i|t)p(j|t)$.

2.1. *Variable selection.* Following CRUISE, we use Pearson chi-squared tests to assess the main effects of the predictor variables. For a noncategorical predictor variable, we discretize its values into three or four intervals, depending on the sample size and the number of classes in the node. But we do not convert the chi-squared values to p-values as CRUISE does. Instead, for degrees of freedom (d.f.) greater than one, we use the Wilson–Hilferty [34] approximation to transform each chi-squared value to a standard normal deviate and then use its inverse transformation to convert it to a chi-squared value with one d.f. This technique, which is borrowed from GUIDE [23], avoids the difficulties of computing very small p-values. The detailed procedure is as follows.

PROCEDURE 2.1. Main effect chi-squared statistic for X at node t:

1. If X is a categorical variable:
   (a) Form a contingency table with the class labels as rows and the categories of X as columns.
   (b) Let ν be the degrees of freedom of the table after deleting any rows and columns with no observations. Compute the chi-squared statistic χ^2_ν for testing independence. If $\nu > 1$, use the Wilson–Hilferty approximation twice to

convert χ^2_ν to the 1-d.f. chi-squared

$$W_M(X) = \max\left(0, \left[\frac{7}{9} + \sqrt{\nu}\left\{\left(\frac{\chi^2_\nu}{\nu}\right)^{1/3} - 1 + \frac{2}{9\nu}\right\}\right]^3\right). \tag{2.1}$$

2. If X is a noncategorical variable:
   (a) Compute $\bar{x}$ and s, the mean and standard deviation, respectively, of the values of X in t.
   (b) If $N(t) \geq 20J_t$, divide the range of X into four intervals with boundary values $\bar{x}$ and $\bar{x} \pm s\sqrt{3}/2$. Otherwise, if $N(t) < 20J_t$, divide the range of X into three intervals with boundary values $\bar{x} \pm s\sqrt{3}/3$. If X has a uniform distribution, these boundaries will yield intervals with roughly equal numbers of observations.
   (c) Form a contingency table with the class labels as rows and the intervals as columns.
   (d) Follow step 1(b) to obtain $W_M(X)$.

Discretization in step 2(b) is needed to permit application of the chi-squared test. Although the boundaries are chosen for reasons of computational expediency, our empirical experience indicates that the particular choice is not critical for its purpose, namely, unbiasedness in variable selection. Note that the boundaries are not used as split points; a search for the latter is carried out separately in Algorithm 2.2 below.

We apply the same idea to assess the local interaction effects of each pair of variables, using Cartesian products of sets of values for the columns of the chi-squared table.

PROCEDURE 2.2. Interaction chi-squared statistic for a pair of variables X_1 and X_2 at node t:

1. If X_i $(i = 1, 2)$ is noncategorical, split its range into two intervals (A_{i1}, A_{i2}) at the sample mean $\bar{x}$ if $N(t) < 45J_t$, or three intervals (A_{i1}, A_{i2}, A_{i3}) at the points $\bar{x} \pm s\sqrt{3}/3$, if $N(t) \geq 45J_t$. If X_i $(i = 1, 2)$ is categorical, let A_{ik} denote the singleton set containing its kth value.
2. Divide the (X_1, X_2)-space into sets $B_{k,m} = \{(x_1, x_2) : x_1 \in A_{1k}, x_2 \in A_{2m}\}$, for $k, m = 1, 2, \ldots$.
3. Form a contingency table with the class labels as rows and $\{B_{k,m}\}$ as columns. Compute its chi-squared statistic and use (2.1) to transform it to a 1-d.f. chi-squared value $W_I(X_1, X_2)$.

To control the frequency with which interaction tests are carried out, we put a Bonferroni-corrected significance threshold on each set of tests and carry out the interaction tests only if all main effects are not significant. Let $\chi^2_{\nu,\alpha}$ denote the upper-α quantile of the chi-squared distribution with ν d.f. The algorithm for variable selection can now be stated as follows.

ALGORITHM 2.1 (Variable selection). Let $K > 1$ be the number of nonconstant predictor variables in the node. Define $\alpha = 0.05/K$ and $\beta = 0.05/\{K(K-1)\}$:

1. Use Procedure 2.1 to find $W_M(X_i)$ for $i = 1, 2, \ldots, K$.
2. If $\max_i W_M(X_i) > \chi^2_{1,\alpha}$, select the variable with the largest value of $W_M(X_i)$ and exit.
3. Otherwise, use Procedure 2.2 to find $W_I(X_i, X_j)$ for each pair of predictor variables:
   (a) If $\max_{i \neq j} W_I(X_i, X_j) > \chi^2_{1,\beta}$, select the pair with the largest value of $W_I(X_i, X_j)$ and exit.
   (b) Otherwise, select the X_i with the largest value of $W_M(X_i)$.

2.2. *Split set selection.* After Algorithm 2.1 selects a variable X at node t, we need to find a set of X-values to form the split $t = t_L \cup t_R$, where t_L and t_R denote the left and right subnodes of t. Let p_L and p_R denote the proportions of samples in t going into t_L and t_R, respectively. We seek the split that minimizes the weighted sum of Gini impurities

$$p_L g(t_L) + p_R g(t_R). \tag{2.2}$$

Let n be the number of distinct values of X in t. If X is noncategorical, there are $(n-1)$ possible splits, with each split point the mean of two consecutive order statistics. If X is categorical, the number of possible splits is $(2^{n-1} - 1)$, which grows exponentially with n. If $J = 2$, however, we use the following short-cut to reduce the search on the categorical variable to $(n-1)$ splits. It is a special case of a more general result proved in [5], Section 9.4.

THEOREM 2.1. *Suppose that $J = 2$ and that X is a categorical variable taking n distinct values. Let $r(a)$ denote the proportion of class 1 observations among those with $X = a$. Let $a_1 \prec a_2 \prec \cdots \prec a_n$ be an ordering of the X values such that $r(a_1) \leq r(a_2) \leq \cdots \leq r(a_n)$. Given any set A, let the observations be split into two groups $t_L = \{X \in A\}$ and $t_R = \{X \notin A\}$. The set A that minimizes the function $p_L g(t_L) + p_R g(t_R)$ is $A_i = \{a_1, a_2, \ldots, a_i\}$ for some $i = 1, 2, \ldots, n-1$.*

If X is noncategorical, we carry out an exhaustive search for the best split of the form $X \leq c$, with c being the midpoint of two consecutive order statistics. If X is categorical, an exhaustive search is done only if $n \leq 11$ or if $J = 2$ (using the above shortcut). Otherwise, a restricted search is performed as follows:

1. If $J \leq 11$ and $n > 20$, divide the observations in t into n groups according to the categorical values of X and find the class that minimizes the misclassification cost in each group. Map X to a new categorical variable X' whose values are the minimizing class labels. Carry out an exhaustive search for a split on X' and re-express it in terms of X.

2. If $J > 11$ or $n \leq 20$, transform the categorical values into 0–1 dummy vectors and apply linear discriminant analysis (LDA) to them to find the largest discriminant coordinate X''. Find the best split on X'' and then re-express it in terms of X. This technique is employed in Loh and Vanichsetakul [25].

The complete details are given as Procedure A.1 in the Appendix.

What to do if Algorithm 2.1 selects a pair of variables X_1 and X_2, say? Then the split search is more complicated, because the best split on X_1 may depend on how X_2 is subsequently split. Similarly, the best split on X_2 should consider the best subsequent splits on X_1. Suppose that t is split first by one variable into t_L and t_R, and that t_L is split into t_{LL} and t_{LR}, and t_R into t_{RL} and t_{RR}, by the other variable. Let p_{LL}, p_{LR}, p_{RL} and p_{RR} denote the proportions of samples in t_{LL}, t_{LR}, t_{RL} and t_{RR}, respectively. We select the split t_L that minimizes

$$p_{LL}g(t_{LL}) + p_{LR}g(t_{LR}) + p_{RL}g(t_{RL}) + p_{RR}g(t_{RR}) \tag{2.3}$$

over t_{LL}, t_{LR}, t_{RL} and t_{RR}.

Because this requires a two-level search, we restrict the number of candidate splits to keep computation under control. Let m_0 be a user-specified number so that only splits yielding at least m_0 cases in each subnode are permitted. For splits on noncategorical variables X_k, define $f = \min(100N^{-1}, 1)$. Let $\lfloor \cdot \rfloor$ be the greatest integer function and let

$$d = \min\{\max(\lfloor fN(t) \rfloor, 9), N(t) - 2m_0 + 1\}$$

be the number of split points to be evaluated. Clearly, $d < 100$. Let $x_k(i)$ denote the ith order statistic of X_k. The set of restricted split points on X_k are the members of the set

$$S_k = \{x_k(i_1), x_k(i_2), \ldots, x_k(i_d)\}, \tag{2.4}$$

where $i_j = m_0 + \lfloor j(N(t) - 2m_0)/(d+1) \rfloor$ and $j = 1, 2, \ldots, d$. Technical details of the search, covering the cases where 0, 1 or 2 variables are categorical, are given in Procedures A.2–A.4 in the Appendix. The entire split selection procedure can be stated as follows:

ALGORITHM 2.2 (Split selection).

1. Apply Algorithm 2.1 to the data in t to select the split variable(s).
2. If one variable is selected and it is categorical, use Procedure A.1 to split t on that variable.
3. If one variable is selected and it is noncategorical, search through all midpoints, c, of the ordered data values for the split $t_L = \{X \leq c\}$ that minimizes (2.2).
4. If two variables are selected, apply Procedure A.2, A.3 or A.4 to split t, depending on whether zero, one or two of the variables are categorical.

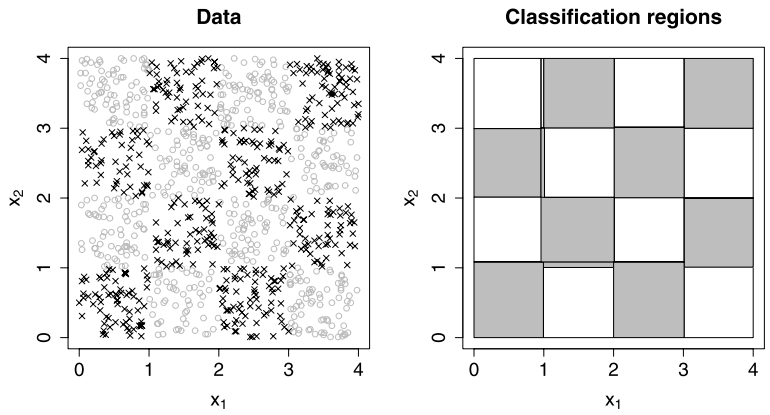

FIG. 5. *Data* (*left*) *and classification regions* (*right*) *for the S method.*

The power of this algorithm is best demonstrated by an artificial example. We simulate one thousand observations, each randomly assigned with probability 0.5 to one of two classes. Each class is uniformly distributed in the alternating squares of a 4×4 chess board in the (X_1, X_2)-plane. The left side of Figure 5 shows one realization, with 520 observations from class 1 and 480 from class 2. We add eight independently and uniformly distributed noise variables. The ideal classification tree should split on X_1 and X_2 only and have 16 leaf nodes. C4.5 and CTree yield trees with no splits and hence misclassify 480 observations each. RPART gives a tree with 13 leaf nodes, but splits on X_1 or X_2 only three times and misclassifies 347 observations. QUEST misclassifies 27 with a 46-leaf tree. CRUISE splits on X_1 or X_2 most of the time and misclassifies 3, but because it does not perform two-level searches, the tree is still large with 29 leaf nodes. Our algorithm splits on X_1 and X_2 exclusively and yields a 19-leaf tree that misclassifies 4 observations; its classification regions are shown on the right side of Figure 5.

3. Linear splits. Although univariate splits (viz. splits on one variable at a time) are best for interpretation, greater predictive accuracy can sometimes be gained if splits on linear combinations of variables are allowed. CART uses random search to find linear splits, while CRUISE and QUEST use LDA; see also [1, 7, 19, 36]. Nonlinear splits have been considered as well [10, 20].

To appreciate the necessity for linear splits, consider some data on fish from Finland obtained from the *Journal of Statistics Education* data archive (www.amstat.org/publications/jse/datasets/fishcatch.txt). See [8] and [17] for prior usage of these data. There are seven species (classes) in the sample of 159 fish. Their class labels (in parentheses), counts and names are as follows: (1) 35 Bream, (2) 11 Parkki, (3) 56 Perch, (4) 17 Pike, (5) 20 Roach, (6) 14 Smelt and (7) 6 Whitefish. Table 2 lists the seven predictor variables. The data are challenging for

TABLE 2
Predictor variables for fish data

`weight`	Weight of the fish (in grams)
`length1`	Length from the nose to the beginning of the tail (in cm)
`length2`	Length from the nose to the notch of the tail (in cm)
`length3`	Length from the nose to the end of the tail (in cm)
`height`	Maximal height as a percentage of `length3`
`width`	Maximal width as a percentage of `length3`
`sex`	Male or female

univariate splits because the three length variables are highly correlated. For example, Figure 6 shows that it is hard to separate the classes using only univariate splits on `length2` and `length3`. A split along the direction of the data points, however, can separate class 1 (bream) cleanly from the rest.

The CRUISE 2V algorithm [17] does well here because it fits a linear discriminant model to a pair of noncategorical variables in each node. We propose instead to keep the node models simple, but use LDA to find splits on two variables at a time. The restriction to two variables permits each split to be presented graphically. It also reduces the impact of missing values—the more variables are involved in a split, the fewer the number of complete cases for its estimation. To prevent outliers from having undue effects on the estimation of the split direction, we trim them before application of LDA in the following procedure. As before, we use chi-squared tests to select the variables for each linear split.

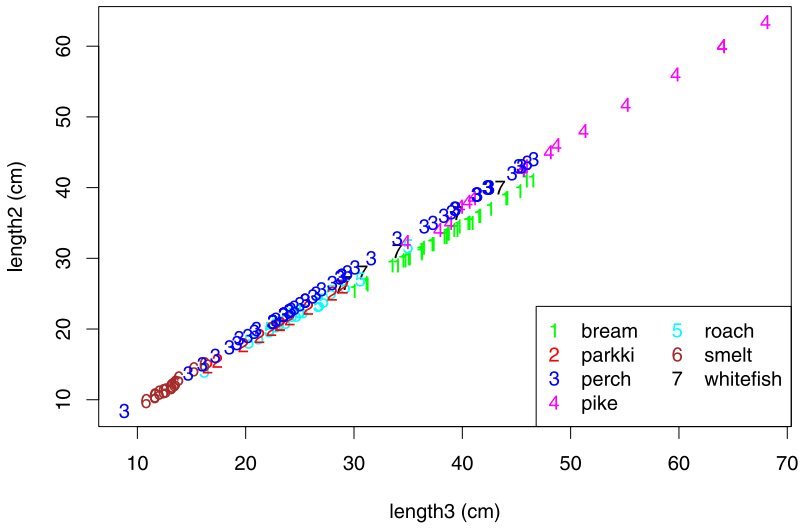

FIG. 6. *Plot of* `length2` *versus* `length3` *for fish data.*

PROCEDURE 3.1. Discriminant chi-squared statistic. Let X_1 and X_2 be two noncategorical variables to be used in a linear split of node t:

1. For class j ($j = 1, 2, \ldots, J$) and each X_i ($i = 1, 2$), compute the class mean $\bar{x}_{i,j}$ and class standard deviation $s_{i,j}$ of the samples in t.
2. Let S_j denote the set of class j samples in t lying in the rectangle $|X_i - \bar{x}_{i,j}| \leq 2s_{i,j}$ for $i = 1, 2$.
3. Find the larger linear discriminant coordinate Z from the observations in $S_1 \cup \cdots \cup S_J$.
4. Project the data in t onto the Z-axis and compute their Z-values.
5. Apply Procedure 2.1 to the Z-values to find the one-d.f. chi-squared value $W_L(X_1, X_2)$.

Although linear splits are more powerful than univariate splits, it is unnecessary to employ a linear split at every node. We see from Figure 6, for example, that almost all the smelts (class 6) can be separated from the other species (except for one misclassified perch) by a univariate split on `length2` or on `length3` at 14 cm. Therefore, we should invoke linear splits only if the main effect and interaction chi-squared tests are not significant, and then only if the linear split itself is significant. This differs from the linear split options in CART, CRUISE and QUEST, which always split on linear combinations of all the predictors. The next algorithm replaces Algorithms 2.1 and 2.2, if linear splits are desired.

ALGORITHM 3.1 (Split selection with the linear split option). Let $K > 0$ be the number of nonconstant predictor variables in node t and let K_1 be the number among them that are noncategorical. Let $\alpha = 0.05/K$ and let $\beta = 0.05/\{K(K - 1)\}$ if $K > 1$; otherwise, let $\beta = 0$. Further, let $\gamma = 0.05/\{K_1(K_1 - 1)\}$ if $K_1 > 1$, and let it be 0 otherwise:

1. If $K = 1$, define $W_M = \infty$ for the nonconstant variable. Otherwise, perform Procedure 2.1 to find $W_M(X_i)$ for $i = 1, 2, \ldots, K$.
2. If $\max_i W_M(X_i) > \chi^2_{1,\alpha}$, let X' be the variable with the largest value of $W_M(X_i)$:
   (a) If X' is categorical, split t on X' with Procedure A.1 and exit.
   (b) Otherwise, split t at the midpoint of the ordered X' values that minimizes the sum of Gini impurities (2.2) and exit.
3. If $\max_i W_M(X_i) \leq \chi^2_{1,\alpha}$, use Procedure 2.2 to find $W_I(X_i, X_j)$ for each pair of predictor variables:
   (a) If $\max_{i \neq j} W_I(X_i, X_j) > \chi^2_{1,\beta}$, select the pair with the largest value of $W_I(X_i, X_j)$ and use Procedure A.2, A.3 or A.4 to split t, depending on whether 0, 1 or 2 variables are categorical, and exit.
   (b) If $K_1 = 1$, go to step 3(c)ii below.
   (c) If $K_1 > 1$, use Procedure 3.1 to find $W_L(X_i, X_j)$ for each pair of noncategorical variables:

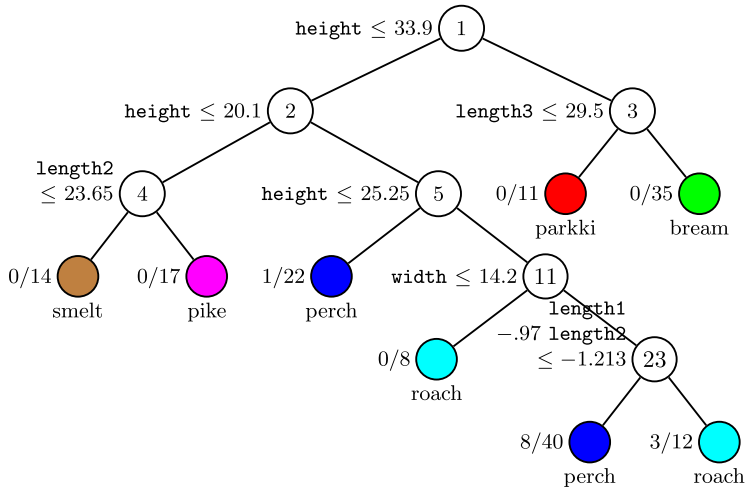

FIG. 7. *Tree model using the* S *method for the fish data. At each intermediate node, a case goes to the left child node if and only if the condition is satisfied. The predicted class and number of errors divided by sample size are printed below and to the left of each leaf node.*

i. If $\max_{i \neq j} W_L(X_i, X_j) > \chi^2_{1,\gamma}$, select the pair of variables with the largest $W_L(X_i, X_j)$-value, split t on their larger discriminant coordinate, and exit.
ii. Otherwise, let X' be the variable with the largest value of $W_M(X_i)$ and go to step 2 above.

Figure 7 shows the pruned tree for the fish data using this algorithm. The tree has univariate splits everywhere except at node 23, where it uses a linear split on variables `length1` and `length2`, and misclassifies 12 observations. Figure 8 displays jittered plots of the data in the intermediate nodes with univariate splits. Figure 9 shows two views of the data in the node with the linear split; the left panel displays the data in terms of `length1` and `length2`, and the right panel shows them in terms of the linear discriminant coordinate. Obviously, it would be impossible to show the left panel if the linear split involves more than two variables.

If linear splits are disallowed, the tree has 12 leaf nodes and misclassifies 10 observations. The CRUISE, QUEST, CTree and RPART trees have 16, 5, 7 and 6 leaf nodes and misclassify 24, 26, 28 and 21 observations, respectively. CRUISE 2V, which employs bivariate linear discriminant leaf node models, yields a tree with 5 leaf nodes and misclassifies 3.

The variable definitions in Table 2 are taken directly from the data source. If the length, width and height variables are all expressed as percentages of `length3`, the resulting tree has 6 leaf nodes with no linear splits and misclassifies 7 observations. Conversely, if all are expressed in cm instead of percentages, the tree has

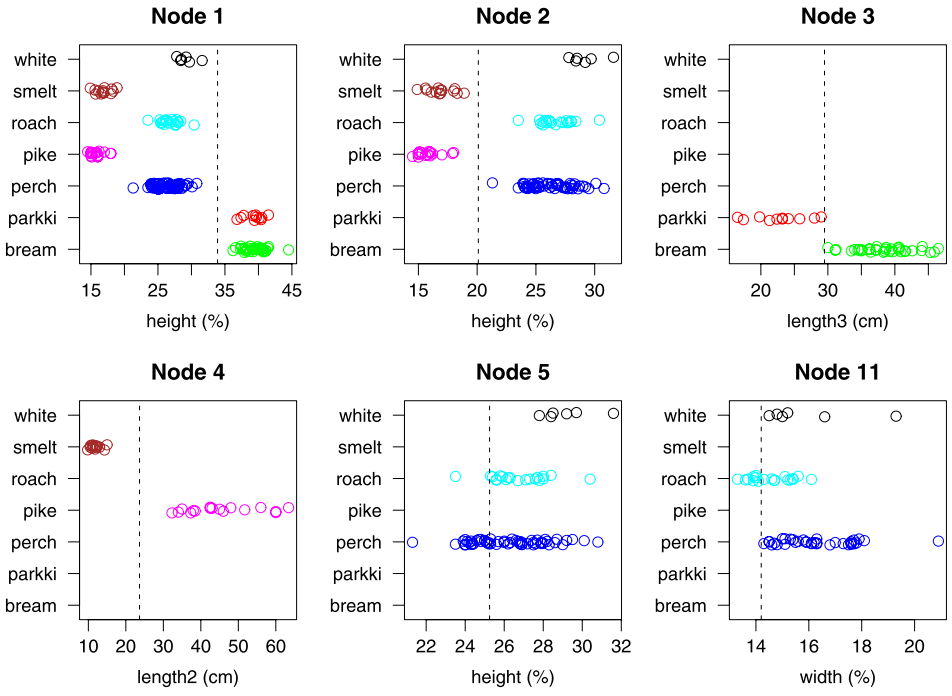

FIG. 8. *Plots of fish data in the intermediate nodes of the tree in Figure 7, with dashed lines marking the splits. Points are vertically jittered.*

15 leaf nodes, employs 4 linear splits and 1 interaction split, and misclassifies 17. Thus, transformations of the variables can make a difference. We note, further, that

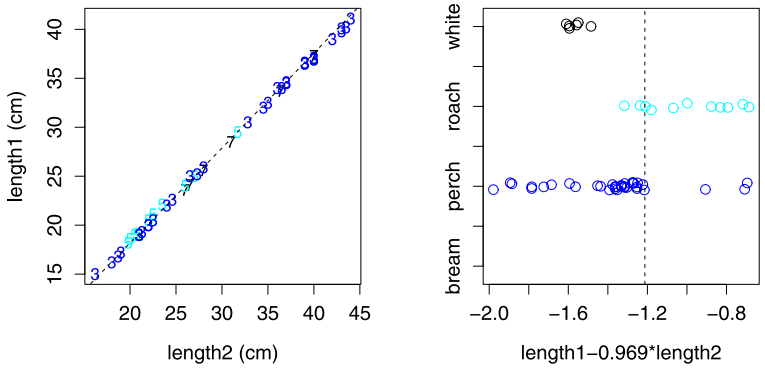

FIG. 9. *Plots of fish data in node* 23 *of the tree in Figure 7, with dashed lines marking the split. The left panel shows the data in terms of variables* `length1` *and* `length2`; *the right panel is a jittered plot of the same data in the linear discriminant direction. The plot symbols in the left panel are the same as those in Figure 6.*

the error rates are probably biased low, because they are estimated from the same data set used to build the models. Cross-validation estimates are used to compare algorithms in Section 6.

4. Selection bias. As mentioned earlier, it is important for an algorithm to be without variable selection bias. Unbiasedness here refers to the property that the X variables are selected with equal probability, when each is independent of the class variable. To show that Algorithm 3.1 is practically unbiased, we report the results of a simulation experiment with $N = 500$ and six X variables. The class variable takes two values with equal probabilities and is independent of the X variables. We consider two scenarios. In the *independence* scenario, the X variables are mutually independent with the following distributions:

1. X_1 is categorical with $P(X_1 = 1) = P(X_2 = 2) = 1/2$;
2. X_2 is categorical with $P(X_2 = 1) = 1/6$, $P(X_2 = 2) = 1/3$, and $P(X_2 = 3) = 1/2$;
3. X_3 is categorical taking six values with equal probability;
4. X_4 is chi-squared with one degree of freedom;
5. X_5 is normal;
6. X_6 is uniform on the unit interval.

In the *dependence* scenario, X_1 and X_6 are independent and distributed as before, but X_4 and X_5 are bivariate normal with correlation 0.7 and X_2 and X_3 have the joint distribution shown in Table 3.

Table 4 shows the number of times each variable is chosen over 10,000 simulation trials. Among univariate splits, there is a slightly lower probability that a non-categorical variable is selected (most likely due to discretization of the continuous variables or the Wilson–Hilferty approximation), but it is offset by the probability that such variables are selected through linear splits. Because two X variables are involved in a linear split, each one is double-counted in the three columns on the right side of Table 4. Therefore, to estimate the overall selection probability for each variable, we halve these counts before adding them to the corresponding univariate split counts. The results, shown in Table 5, are all within two simulation standard errors of 1/6 (the required value for unbiasedness).

TABLE 3
Joint distribution of X_2 and X_3 in the dependence scenario

	X_3					
X_2	**1**	**2**	**3**	**4**	**5**	**6**
1	1/12	1/12	1/24	1/24	1/24	1/24
2	1/24	1/24	1/12	1/12	1/24	1/24
3	1/24	1/24	1/24	1/24	1/12	1/12

TABLE 4

Number of times (*out of* 10,000 *simulation trials*) *that each variable is selected to split the root node. Variables* X_1, X_2 *and* X_3 *are categorical and can only appear in a univariate split*; *variables* X_4, X_5 *and* X_6 *can appear in a univariate or a linear split*

	Univariate splits						Linear splits		
Scenario	X_1	X_2	X_3	X_4	X_5	X_6	X_4	X_5	X_6
Independence	1713	1656	1620	1567	1492	1533	302	310	226
Dependence	1684	1643	1601	1544	1564	1503	356	342	224

5. Kernel and nearest-neighbor node models. So far, we have tried to make the tree structure more parsimonious and precise by controlling selection bias and improving the discriminatory power of the splits. Another way to reduce the size of the tree structure is to fit nontrivial models to the nodes. Kim and Loh [17] and Gama [11], for example, use linear discriminant models. Although effective in improving predictive accuracy, linear discriminant models are not as flexible as nonparametric ones and may not simplify the tree structure as much. We propose to use kernel and nearest-neighbor models instead. To allow the fits to be depicted graphically, we again restrict each model to use at most two variables. Further, to save computation time, we use Algorithm 2.2 to find the split variables, skipping the linear splits. Buttrey and Karo [6] also fit nearest-neighbor models, but they do this only after a standard classification tree is constructed. As a result, their tree structure is unchanged by model fitting. Since we fit a model to each node as the tree is grown, we should get more compact trees.

First, consider kernel discrimination, which is basically maximum likelihood with a kernel density estimate for each class in a node. If the selected X variable is categorical, its class density estimate is just the sample class density function. If X is noncategorical, we use a Gaussian kernel density estimate. Let s and r denote the standard deviation and the inter-quartile range, respectively, of a sample of observations, $x_1, x_2, \ldots, x_n$, from X. The kernel density estimate is $\hat{f}(x) = (nh)^{-1} \sum_{i=1}^{n} \phi\{(x - x_i)/h\}$, where ϕ is the standard normal density function and

TABLE 5

Probabilities of variable selection for a null model estimated by simulation. The simulation standard error is 0.0037. *If the method is unbiased, the probabilities should be all equal to* 0.1667

	X_1	X_2	X_3	X_4	X_5	X_6
Independent	0.1713	0.1656	0.1620	0.1718	0.1647	0.1646
Dependent	0.1684	0.1643	0.1601	0.1722	0.1735	0.1615

h is the bandwidth. The following formula, adapted from Stata [32],

$$h = \begin{cases} 2.5\min(s, 0.7413r)n^{-1/5}, & \text{if } r > 0, \\ 2.5sn^{-1/5}, & \text{otherwise,} \end{cases} \tag{5.1}$$

is used in the calculations reported here. This bandwidth is more than twice as wide as the asymptotically optimal value usually recommended for density estimation; Ghosh and Chaudhuri [12] find that the best bandwidth for discrimination is often much larger than that for density estimation. For our purposes, asymptotic formulas cannot be taken too seriously, because the node sample size decreases with splitting.

If a pair of noncategorical variables is selected, we fit a bivariate kernel density to the pair for each class. If the split is due to one categorical and one noncategorical variable, we fit a kernel density estimate to the noncategorical variable for each class and each value of the categorical variable, using an average bandwidth that depends only on the class. Averaging smoothes out the effects of small or highly unbalanced sample sizes. The details are given in the next algorithm.

ALGORITHM 5.1 (Kernel models). Let Y denote the class variable and apply Algorithm 2.2 to find one or two variables to split a node t:

1. If the split is due to a main effect chi-squared statistic (Procedure 2.1), let X be the selected variable. Fit a kernel density estimate to X for each class in t using bandwidth (5.1) with $n = N(t)$.
2. If the split is due to an interaction chi-squared statistic (Procedure 2.2), let X_1 and X_2 be the selected variables. Fit a bivariate density estimate to (X_1, X_2) for each class in t as follows:
   (a) If X_1 and X_2 are categorical, use their sample class joint density.
   (b) If X_1 is categorical and X_2 is noncategorical, then for each combination of (X_1, Y) values present in t, find a bandwidth $h(Y, X_1)$ using (5.1). Let $\bar{h}(Y)$ be the average of $h(Y, X_1)$. For each value of X_1 and Y, find a kernel density estimate for X_2 with $\bar{h}(Y)$ as bandwidth.
   (c) If both variables are noncategorical, fit a bivariate Gaussian kernel density estimate to each class with correlation equal to the class sample correlation and n equal to the class sample size in (5.1).

The predicted class is the one with the largest estimated density.

We use the same idea for nearest-neighbor models. For noncategorical variables, the number of nearest neighbors, k, is given by the formula

$$k = \max(3, \lceil \log n \rceil), \tag{5.2}$$

where n is the number of observations and $\lceil x \rceil$ denotes the smallest integer greater than or equal to x. We require k to be no less than 3 to lessen the chance of ties. The full details are given next.

ALGORITHM 5.2 (Nearest neighbor models). Let $\hat{Y}$ denote the predicted value of Y. Use Algorithm 2.2 to find one or two variables to split a node t:

1. If the split is due to a main effect chi-squared statistic (Procedure 2.1), let X be the selected variable:
   (a) If X is categorical, then $\hat{Y}$ is the highest probability class among the observations in t with the same X value as the one to be classified.
   (b) If X is noncategorical, then $\hat{Y}$ is given by the k-nearest neighbor classifier based on X with $n = N(t)$ in (5.2).
2. If the split is due to an interaction chi-squared statistic (Procedure 2.2), let X_1 and X_2 be the selected variables:
   (a) If both variables are categorical, then $\hat{Y}$ is the highest probability class among the observations in t with the same (X_1, X_2) values as the one to be classified.
   (b) If X_1 is categorical and X_2 is noncategorical, then $\hat{Y}$ is given by the k-nearest neighbor classifier based on X_2 applied to the set S of observations in t that have the same X_1 value as the one to be classified, with n being the size of S in (5.2).
   (c) If both variables are noncategorical, then $\hat{Y}$ is given by the bivariate k-nearest neighbor classifier based on X_1 and X_2 with the Mahalanobis distance and $n = N(t)$ in (5.2).

Figure 10 shows an artificial example that is very challenging for many algorithms. There are 300 observations, with 100 from each of three classes and eight predictor variables. Class 1 is uniformly distributed on the unit circle in the (X_1, X_2)-plane. Class 2 is uniformly distributed on the line $X_1 - X_2 = 0$, and class 3 on the line $X_1 + X_2 = 0$, with $|X_1| < 1$ and $|X_2| < 1$. Variables X_3, X_4 and X_5 are uniformly distributed on the unit interval and X_6, X_7 and X_8 are cat-

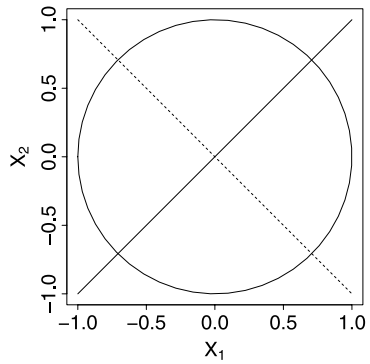

FIG. 10. *Three-class problem, with class* 1 *uniformly distributed on the circle and classes* 2 *and* 3 *each uniformly distributed on its own diagonal line.*

egorical, each taking 21 equi-probable values. The X variables are mutually independent.

QUEST with linear splits gives a 38-leaf tree that misclassifies 3 samples. CRUISE with LDA models gives a 17-leaf tree that misclassifies 10 samples. RPART, QUEST with univariate splits and our simple node method are about equal, misclassifying 85, 81 and 75 samples, respectively. C4.5 and CTree are the worst, with 134 and 200 misclassified and 84 and 1 leaf nodes, respectively. In contrast, our kernel and nearest-neighbor methods yield trees with no splits after pruning and misclassify 2 and 8 training samples, respectively.

6. Comparison on forty-six datasets. The error rates discussed so far are biased low because they are computed from the same data that are used to construct the tree models. To obtain a better indication of relative predictive accuracy, we compare the algorithms listed in Table 6 on forty-two real and four artificial data sets using ten-fold cross-validation. Each data set is randomly divided into ten roughly equal parts and one-tenth is set aside in turn as a test set to estimate the predictive accuracy of the model constructed from the other nine-tenths of the data. The cross-validation estimate of error is the average of the ten estimates. Equal misclassification costs are used throughout. As elsewhere in this article, the trees are pruned to have minimum cross-validation estimate of misclassification cost. All other parameter values are set at their respective defaults.

The four artificial data sets are those in Figures 5 (`int`) and 10 (`cl3`), and the digit (`led`) and waveform (`wav`) data in [5]. Sample size ranges from 97 to 45,222; number of classes from 2 to 11; number of categorical variables from 0 to 60; number of noncategorical variables from 0 to 69; and maximum number of categories among categorical variables from 0 to 41. We use the notations "`S`," "`K`" and "`N`" to refer to our proposed method with simple (i.e., constant), kernel and nearest-neighbor node models, respectively. `S` employs linear splits, but `K` and `N` do not.

Eleven data sets have missing values. In the `S`, `N` and `K` algorithms, missing values in a categorical variable are assigned to their own separate "missing" category.

TABLE 6
Algorithms and plot symbols for the comparison experiment

`C45`	C4.5
`C2d`	CRUISE with interaction detection and simple node models
`C2v`	CRUISE with interaction detection and linear discriminant node models
`Qu`	QUEST with univariate splits
`Ql`	QUEST with linear splits
`Rp`	RPART
`Ct`	CTree
`S`	Proposed method with simple node models (Algorithm 3.1)
`K`	Proposed method with kernel node models (Algorithm 5.1)
`N`	Proposed method with nearest-neighbor node models (Algorithm 5.2)

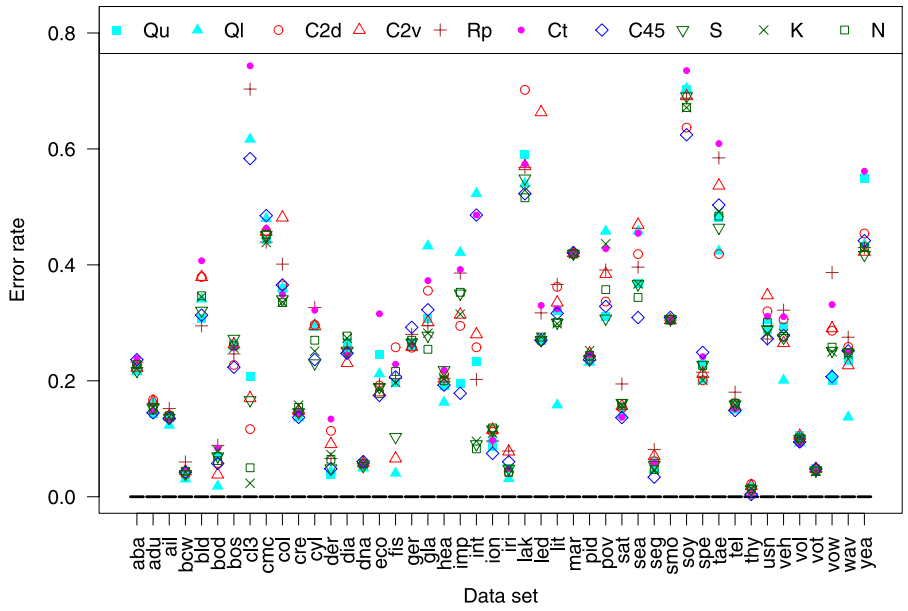

FIG. 11. *Error rates of algorithms by data set.*

Observations with missing values in a noncategorical split variable are always sent to the left node. If there are enough such cases, our algorithms will consider a split on missingness as one of the candidate splits. Bandwidths are computed from cases with nonmissing values in the selected variables. Kernel and nearest-neighbor model classifications are applied only to observations with nonmissing values in the selected variables. Observations with missing values in the selected variables are classified with the majority rule.

Figure 11 graphs the errors rates of the ten algorithms for each data set. Despite the large range of the error rates (from near 0 to about 0.7), the algorithms have very similar accuracy for about half of the data sets. The most obvious exceptions are the artificial data sets `int` and `cl3`, where our `K`, `N` and `S` algorithms have a superior edge; algorithms not designed to detect interactions pay a steep price here. Two other examples showing interesting patterns are the fish (`fis`) data used in Section 3 and the `bod` data set [13], where body measurements are used to classify gender. For these two data sets, algorithms that employ LDA techniques (`C2v`, `Ql` and `S`) are more accurate. `Ql` seems to be either best or worst for a majority of the data sets.

Figure 12 shows the corresponding results for the sizes of the trees in terms of their numbers of leaf nodes. `C45` tends to produce the largest trees, while `C2v` and `Rp` often give the shortest.

Figure 13 shows arithmetic means (over the 46 data sets) of the error rates and numbers of leaf nodes for each algorithm, with corresponding 95% Tukey simul-

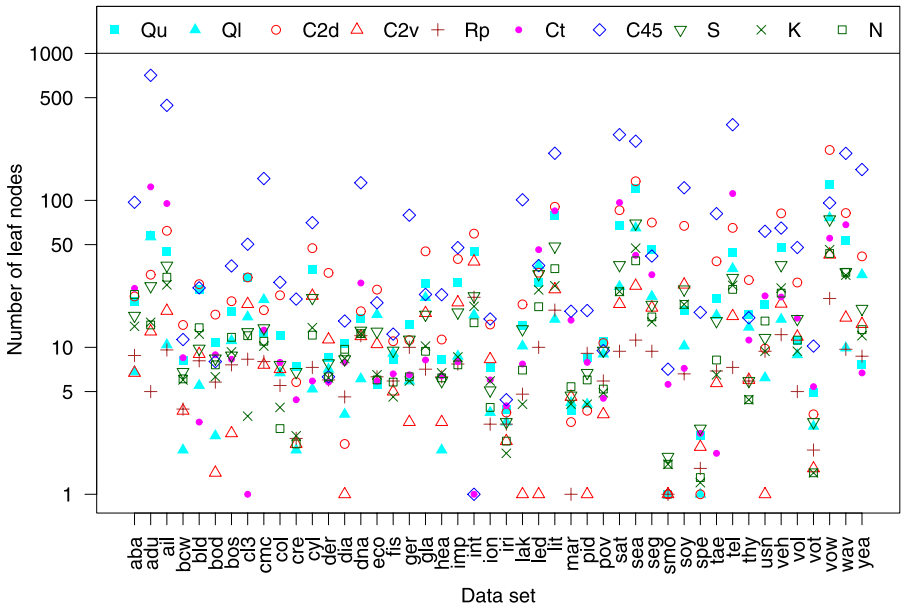

FIG. 12. *Numbers of leaf nodes by data set.*

taneous confidence intervals. The latter are obtained by fitting two-factor mixed models to the error rates and number of leaf nodes separately, using algorithm as fixed factor, data set as random factor, and their interaction as error term. S and

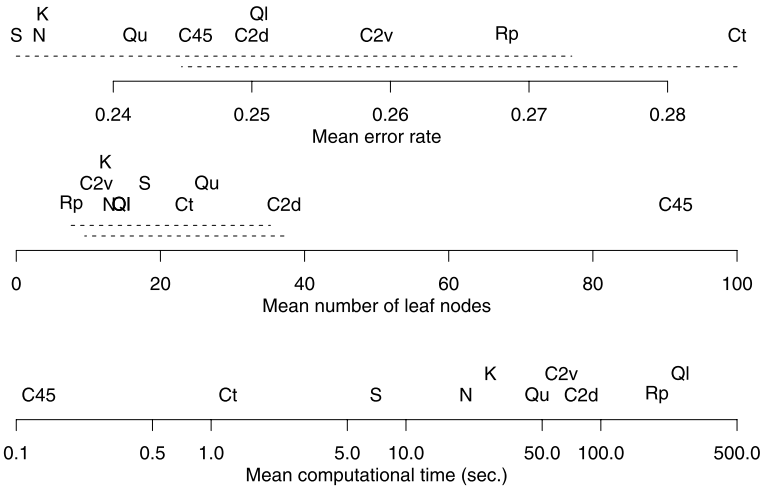

FIG. 13. *Means of error rates, numbers of leaf nodes and computational times; for the top two plots, dashed lines join algorithms that do not differ significantly at the 95% simultaneous level of confidence.*

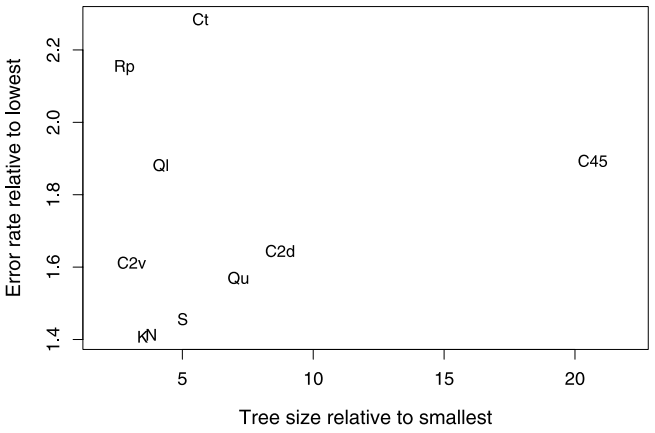

FIG. 14. *Mean of error rates relative to the lowest (for each data set) versus mean of tree size relative to that of the smallest tree; tree size is measured by the number of leaf nodes.*

`Ct` have the smallest and largest, respectively, mean error rates. The confidence intervals show that the mean error rate of `Ct` differs significantly from those of `S`, `N`, `K` and `Qu`; other differences are not significant. As for mean number of leaf nodes, `C45` is significantly different from the others, as is `Rp` from `C2d`.

Figure 13 also shows the mean computational times, on a 2.66Ghz quad-core Linux PC with 8Gb memory. Because the algorithms are implemented in different computer languages (`Rp` and `Ct` in R, `C45` in C and the others in Fortran), the results compare execution times rather than number of computer operations. Further, the mean time for `Rp` is dominated by two data sets each having six classes and categorical variables with many categorical levels. It is well known that the computational time of CART, upon which `Rp` is based, increases exponentially with the number of categorical levels when the number of classes exceeds two.

Another way to account for differences among data sets is to compare the ratio of the error rate of each algorithm to the lowest error rate among the ten algorithms within each data set. We call this ratio the "error rate relative to the lowest" for the particular algorithm and data set. The mean of these ratios for an algorithm over the data sets yields an overall measure of its relative inaccuracy. Applying the same procedure to the number of leaf nodes gives an overall measure of relative tree size for each algorithm. Figure 14 gives a plot of these two measures. The best algorithms are those in the bottom left corner of the plot: `K`, `N`, `S` and `C2v`. They are relatively most accurate and they yield relatively small trees.

7. Tree ensembles. A tree ensemble classifier uses the majority vote from a collection of tree models to predict the class of an observation. *Bagging* [3] creates the ensemble by using bootstrap samples of the training data to construct the trees. Random Forest (`RF`), which is based on CART and employs 500 trees,

goes beyond bagging by splitting each node on a random subset of $\sqrt{K}$ variables (K being the total number of variables) and not pruning the trees. Because it is practically impossible to interpret so many trees, ensemble classifiers are typically used for prediction only.

To see how much an ensemble can affect the predictive accuracy of the single-tree models proposed here, we use the bagging and RF ideas to create two new ensemble classifiers. The first, called *bagged GUIDE* (BG), is a collection of 100 pruned trees, each constructed using the S method from bootstrap samples. The second ensemble classifier, called *GUIDE forest* (GF), consists of 500 unpruned trees constructed by the S method without interaction and linear splits. As in RF, GF uses a random subset of $\sqrt{K}$ variables to split each node.

Figure 15 shows the error rates of BG, GF and RF compared with those of the S and K methods for each data set. The R package of Liaw and Wiener [21], which we use for RF, is not applicable if the data set has predictor variables with more than 32 categorical levels or if the test sample contains class labels that are not present among the training samples. The first condition occurs in the adu and lak data sets, which have categorical variables with 41 and 35 levels, respectively, and the second condition occurs with the eco data set, which has 8 classes and a total sample size of 336. Table 7 gives the mean error rates with these three data sets excluded. All three ensemble methods have lower means than the single-tree methods, but the differences are fairly small on average. Figure 15 shows that although RF is best for many data sets, it performs particularly poorly compared

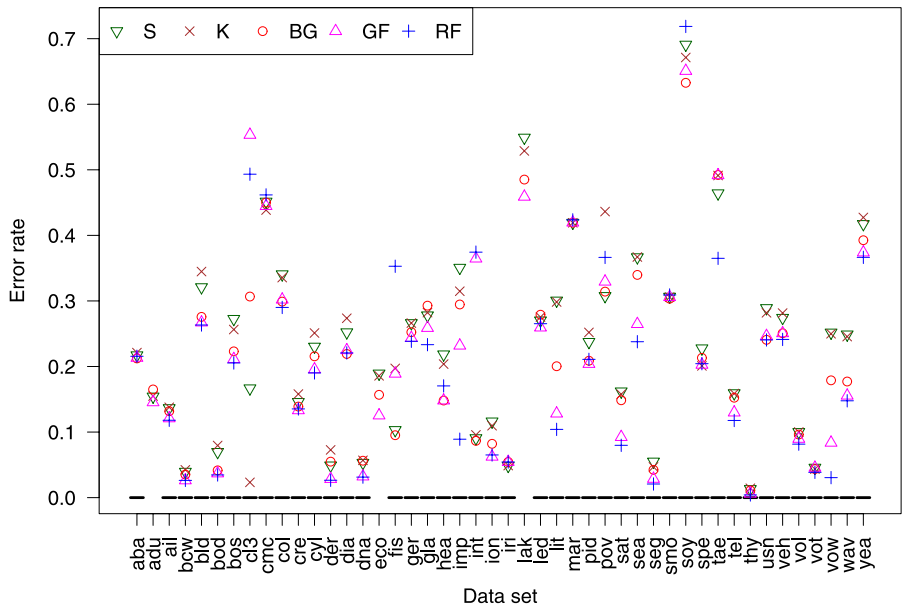

FIG. 15. *Single tree models* (S *and* K) *versus ensemble models* (BG, GF *and* RF).

TABLE 7
Mean error rates over the 46 *data sets where* RF *is applicable. Differences are not statistically significant*

Algorithm	**S**	**K**	**BG**	**GF**	**RF**
Error rate	0.228	0.231	0.212	0.212	0.206

to S and K for three data sets: cl3 (Figure 10), fis (fish data in Section 3) and int (Figure 5)—data sets that have strong linear or interaction effects. GF shares the same difficulties with RF, but BG does not, because the latter allows linear and interaction splits.

8. Conclusion. Improving the prediction accuracy of a tree and the precision of its splits is a balancing act. On the one hand, we must refrain from searching too greedily for splits, as the resulting selection bias can cause irrelevant variables to be selected. On the other hand, we should search hard enough to avoid overlooking good splits hidden behind interactions or linear relationships. We solve this problem by using three groups of significance tests, with increasing complexity of effects and decreasing order of priority. The first group of tests for main effects is always carried out. The second group, which tests for interactions, is performed only if no main effect test is significant. The third group, which tests for linear structure, is performed only if no test in the first two groups is significant. A Bonferroni correction controls the significance level of each group. In addition, if an interaction test is significant, the split is found by a two-level search on the pair of interacting variables.

If greater reduction in the size of the tree structure is desired, we can fit a kernel or a nearest-neighbor model to each node. Owing to the flexibility of these models, we dispense with linear splits in such situations. We showed by an example that when there are weak main effects but strong two-factor interaction effects, classification trees constructed with these models can achieve substantial gains in accuracy and tree compactness. They require more computation than trees with simple node models, but the empirical evidence indicates that their prediction accuracy remains high even for ordinary data sets.

We also investigated the effect of tree ensembles on predictive accuracy. Although Random Forest quite often yields higher accuracy than the single-tree models S and K, the average increase is only about 10% for the 43 data sets in the study. Much depends on the complexity of the data. If there are strong interaction or linear effects, the single-tree algorithms proposed here can be substantially more accurate than Random Forest. Further, the choice of the single-tree algorithm used in the ensemble matters.

All the proposed algorithms are implemented in version 8 of GUIDE, which may be obtained from www.stat.wisc.edu/~loh/guide.html for the Linux, Macintosh and Windows operating systems.

APPENDIX

PROCEDURE A.1. Split set selection for a categorical variable X. Let $\{a_1, a_2, \ldots, a_n\}$ be the set of distinct values of X in node t:

1. If $J = 2$ or $n \leq 11$, search all subsets S to find a split of the form $t_L = \{X \in S\}$ that minimizes (2.2).
2. Else if $J \leq 11$ and $n > 20$, for each $i = 1, 2, \ldots, n$, let j_i be the class that minimizes the misclassification cost for the observations in $t \cap \{X = a_i\}$. Define the new categorical variable $X' = \sum_i j_i I(X = a_i)$ and search all subsets S' to find a split of the form $\{X' \in S'\}$ that minimizes (2.2). Re-express the selected split in terms of X as $t_L = \{X \in S\}$.
3. Else use LDA as follows:
   (a) Convert X into a vector of dummy variables $(u_1, u_2, \ldots, u_n)$, where $u_i = 1$ if $X = a_i$ and $u_i = 0$ otherwise.
   (b) Obtain the covariance matrix of the u-vectors and find the eigenvectors associated with the positive eigenvalues. Project the u-vectors onto the space spanned by these eigenvectors.
   (c) Apply LDA to the projected u-vectors to find the largest discriminant coordinate $v = \sum_i c_i u_i$.
   (d) Let $v_{(1)} < v_{(2)} < \cdots$ denote the (at most n) sorted v-values.
   (e) Find the split $t_L = \{v \leq v_{(m)}\}$ that minimizes (2.2).
   (f) Re-express the split as $t_L = \{X \in S\}$.

PROCEDURE A.2. Split selection between two noncategorical variables X_1 and X_2. Let S_k $(k = 1, 2)$ be defined as in (2.4):

1. Split t first on X_1 and then on X_2 as follows. Given numbers c, d and e, let $t_L = \{X_1 \leq c\}$, $t_R = \{X_1 > c\}$, $t_{LL} = t_L \cap \{X_2 \leq d\}$, $t_{LR} = t_L \cap \{X_2 > d\}$, $t_{RL} = t_R \cap \{X_2 \leq e\}$, and $t_{RR} = t_R \cap \{X_2 > e\}$. Search over all $c \in S_1$ and $d, e \in S_2$ to find the best $c = c_1$ that minimizes (2.3).
2. Exchange the roles of X_1 and X_2 in the preceding step to find the best split $\{X_2 \leq c_2\}$ with $c_2 \in S_2$.
3. If the minimum value of (2.3) from the split $\{X_1 \leq c_1\}$ is less than that from $\{X_2 \leq c_2\}$, select the former. Otherwise, select the latter.

PROCEDURE A.3. Split selection between noncategorical X_1 and categorical X_2:

1. First find a split of t on X_1 as follows. Let $t_L = \{X_1 \leq c\}$ and $t_R = \{X_1 > c\}$, where $c \in S_1$ and S_1 is defined in (2.4):
   (a) Consider the observations in t_L. If $J > 2$, form two superclasses, with one containing the class that minimizes the misclassification cost in t_L and the other containing the rest. Use Theorem 2.1 to obtain an ordering $a_1 \prec a_2 \prec \cdots$ of the values of X_2. Define $A_i = \{a_1, a_2, \ldots, a_i\}$, $t_{LL} = t_L \cap \{X_2 \in A_i\}$, and $t_{LR} = t_L \cap \{X_2 \notin A_i\}$ for $i = 1, 2, \ldots$.

(b) Repeat the preceding step on the observations in t_R to obtain an ordering $b_1 \prec b_2 \prec \cdots$ of the values of X_2. Define $B_j = \{b_1, b_2, \ldots, b_j\}$, $t_{RL} = t_R \cap \{X_2 \in B_j\}$, and $t_{RR} = t_R \cap \{X_2 \notin B_j\}$ for $j = 1, 2, \ldots$.

(c) Let δ_1 be the minimum value of (2.3) over $\{A_i\}$, $\{B_j\}$, and $c \in S_1$. Let c^* be the minimizing value of c.

2. Find the following two splits of t on X_2:

(a) Order the values of X_2 in the set $\{X_1 \le c^*\}$ according to step 1 above and use them to generate splits of the form $t_L = \{X_2 \in U_i\}$ and $t_R = \{X_2 \notin U_i\}$, $i = 1, 2, \ldots$. For each i, find the best splits of t_L and t_R on X_1 into t_{LL}, t_{LR} and t_{RL}, t_{RR}, respectively. Let δ_2 be the minimum value of (2.3), attained at $t_L = \{X_2 \in U\}$, say.

(b) Repeat the preceding step on the observations in $\{X_1 > c^*\}$ to get the split $t_L = \{X_2 \in V\}$, say, with minimizing value δ_3.

3. If $\delta_1 \le \min(\delta_2, \delta_3)$, split t with $\{X_1 \le c^*\}$. Otherwise, select $\{X_2 \in U\}$ if $\delta_2 \le \delta_3$, and $\{X_2 \in V\}$ if $\delta_2 > \delta_3$.

PROCEDURE A.4. Split selection between categorical variables X_1 and X_2:

1. Given U, V and W, let $t_L = \{X_1 \in U\}$, $t_R = \{X_1 \notin U\}$, $t_{LL} = t_L \cap \{X_2 \in V\}$, $t_{LR} = t_L \cap \{X_2 \notin V\}$, $t_{RL} = t_R \cap \{X_2 \in W\}$, and $t_{RR} = t_R \cap \{X_2 \notin W\}$. Let k_1 be the number of distinct values of X_1:

(a) If $J = 2$, search over all sets U, V and W.

(b) If $J > 2$ and $k_1 \le 5$, search over all sets U but restrict the sets V and W as follows. Let j_0 be the class that minimizes the misclassification cost in t and create two superclasses, with one containing j_0 and the other containing the rest. Use Theorem 2.1 on the two superclasses to induce an ordering of the X_2-values and then search for V and W using the ordered values.

(c) If $J > 2$ and $k_1 > 5$, use the method in the preceding step to restrict the search on U, V and W.

Let δ_1 be the smallest searched value of (2.3), and let it be attained at $U = U'$.

2. Repeat step 1 with the roles of X_1 and X_2 reversed and let δ_2 be the minimum value of (2.3), attained with $t_L = \{X_2 \in V'\}$.

3. Split t with $\{X_1 \in U'\}$ if $\delta_1 \le \delta_2$; otherwise, split with $\{X_2 \in V'\}$.

Acknowledgments. The author is grateful to editor M. Stein, an associate editor and a referee for their helpful comments and suggestions. He also thanks T. Hothorn for assistance with the PARTY R package.

REFERENCES

[1] AMASYALI, M. F. and ERSOY, O. (2008). CLINE: A new decision-tree family. *IEEE Transactions on Neural Networks* **19** 356–363.

[2] ATKINSON, E. J. and THERNEAU, T. M. (2000). An introduction to recursive partitioning using the RPART routines. Technical report 61, Biostatistic Section, Mayo Clinic, Rochester, NY.

[3] BREIMAN, L. (1996). Bagging predictors. *Mach. Learn.* **24** 123–140. MR1425957
[4] BREIMAN, L. (2001). Random forests. *Mach. Learn.* **45** 5–32.
[5] BREIMAN, L., FRIEDMAN, J. H., OLSHEN, R. A. and STONE, C. J. (1984). *Classification and Regression Trees*. Wadsworth, Belmont. MR0726392
[6] BUTTREY, S. E. and KARO, C. (2002). Using k-nearest-neighbor classification in the leaves of a tree. *Comput. Statist. Data Anal.* **40** 27–37. MR1930465
[7] CANTU-PAZ, E. and KAMATH, C. (2003). Inducing oblique decision trees with evolutionary algorithms. *IEEE Transactions on Evolutionary Computation* **7** 54–68.
[8] CLARK, V. (2004). *SAS/STAT 9.1 User's Guide*. SAS Publishing, Cary, NC.
[9] DOYLE, P. (1973). The use of Automatic Interaction Detector and similar search procedures. *Operational Research Quarterly* **24** 465–467. MR0324682
[10] FAN, G. (2008). Kernel-induced classification trees and random forests. Manuscript.
[11] GAMA, J. (2004). Functional trees. *Mach. Learn.* **55** 219–250.
[12] GHOSH, A. K., CHAUDHURI, P. and SENGUPTA, D. (2006). Classification using kernel density estimates: Multiscale analysis and visualization. *Technometrics* **48** 120–132. MR2236534
[13] HEINZ, G., PETERSON, L. J., JOHNSON, R. W. and KERK, C. J. (2003). Exploring relationships in body dimensions. *Journal of Statistics Education* **11**. Available at www.amstat.org/publications/jse/v11n2/datasets.heinz.html.
[14] HOSMER, D. W. and LEMESHOW, S. (2000). *Applied Logistic Regression*, 2nd ed. Wiley, New York.
[15] HOTHORN, T., HORNIK, K. and ZEILEIS, A. (2006). Unbiased recursive partitioning: A conditional inference framework. *J. Comput. Graph. Statist.* **15** 651–674. MR2291267
[16] KIM, H. and LOH, W.-Y. (2001). Classification trees with unbiased multiway splits. *J. Amer. Statist. Assoc.* **96** 589–604. MR1946427
[17] KIM, H. and LOH, W.-Y. (2003). Classification trees with bivariate linear discriminant node models. *J. Comput. Graph. Statist.* **12** 512–530. MR2002633
[18] LEE, T.-H. and SHIH, Y.-S. (2006). Unbiased variable selection for classification trees with multivariate responses. *Comput. Statist. Data Anal.* **51** 659–667. MR2297477
[19] LI, X. B., SWEIGART, J. R., TENG, J. T. C., DONOHUE, J. M., THOMBS, L. A. and WANG, S. M. (2003). Multivariate decision trees using linear discriminants and tabu search. *IEEE Transactions on Systems Man and Cybernetics Part A–Systems and Humans* **33** 194–205.
[20] LI, Y. H., DONG, M. and KOTHARI, R. (2005). Classifiability-based omnivariate decision trees. *IEEE Transactions on Neural Networks* **16** 1547–1560.
[21] LIAW, A. and WIENER, M. (2002). Classification and regression by randomforest. *R News* **2** 18–22. Available at http://CRAN.R-project.org/doc/Rnews/.
[22] LIM, T.-S., LOH, W.-Y. and SHIH, Y.-S. (2000). A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Mach. Learn. J.* **40** 203–228.
[23] LOH, W.-Y. (2002). Regression trees with unbiased variable selection and interaction detection. *Statist. Sinica* **12** 361–386. MR1902715
[24] LOH, W.-Y. and SHIH, Y.-S. (1997). Split selection methods for classification trees. *Statist. Sinica* **7** 815–840. MR1488644
[25] LOH, W.-Y. and VANICHSETAKUL, N. (1988). Tree-structured classification via generalized discriminant analysis (with discussion). *J. Amer. Statist. Assoc.* **83** 715–728. MR0963799
[26] MCCULLAGH, P. and NELDER, J. A. (1989). *Generalized Linear Models*, 2nd ed. Chapman and Hall, London. MR0727836
[27] MORGAN, J. N. and MESSENGER, R. C. (1973). THAID: A sequential analysis program for the analysis of nominal scale dependent variables. Technical report, Institute for Social Research, Univ. Michigan, Ann Arbor.

[28] MORGAN, J. N. and SONQUIST, J. A. (1963). Problems in the analysis of survey data, and a proposal. *J. Amer. Statist. Assoc.* **58** 415–434.

[29] NOH, H. G., SONG, M. S. and PARK, S. H. (2004). An unbiased method for constructing multilabel classification trees. *Comput. Statist. Data Anal.* **47** 149–164. MR2087934

[30] PERLICH, C., PROVOST, F. and SIMONOFF, J. S. (2003). Tree induction vs. logistic regression: A learning-curve analysis. *J. Mach. Learn. Res.* **4** 211–255. MR2050883

[31] QUINLAN, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo.

[32] STATACORP. (2003). *Stata Statistical Software: Release 8.0*. Stata Corporation, College Station, TX.

[33] STROBL, C., BOULESTEIX, A.-L. and AUGUSTIN, T. (2007). Unbiased split selection for classification trees based on the Gini index. *Comput. Statist. Data Anal.* **52** 483–501. MR2409997

[34] WILSON, E. B. and HILFERTY, M. M. (1931). The distribution of chi-square. *Proc. Nat. Acad. Sci. USA* **17** 684–688.

[35] WITTEN, I. H. and FRANK, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. Morgan Kaufmann, San Fransico, CA.

[36] YILDLZ, O. T. and ALPAYDIN, E. (2005). Linear discriminant trees. *International Journal of Pattern Recognition and Artificial Intelligence* **19** 323–353.

DEPARTMENT OF STATISTICS
UNIVERSITY OF WISCONSIN
1300 UNIVERSITY AVENUE
MADISON, WISCONSIN 53706
USA
E-MAIL: loh@stat.wisc.edu