Classification Trees With Unbiased Multiway Splits

Hyunjoong KIM and Wei-Yin LOH

Two univariate split methods and one linear combination split method are proposed for the construction of classification trees with multiway splits. Examples are given where the trees are more compact and hence easier to interpret than binary trees. A major strength of the univariate split methods is that they have negligible bias in variable selection, both when the variables differ in the number of splits they offer and when they differ in the number of missing values. This is an advantage because inferences from the tree structures can be adversely affected by selection bias. The new methods are shown to be highly competitive in terms of computational speed and classification accuracy of future observations.

KEY WORDS: Decision tree; Linear discriminant analysis; Missing value; Selection bias.

1. INTRODUCTION

Classification tree algorithms may be divided into two groups—those that yield binary trees and those that yield trees with nonbinary (also called multiway) splits. CART (Breiman, Friedman, Olshen, and Stone 1984) and QUEST (Loh and Shih 1997) are members of the first group. Members of the second group include FACT (Loh and Vanichsetakul 1988), C4.5 (Quinlan 1993), CHAID (Kass 1980), and FIRM (Hawkins 1997). FACT yields one node per class at each split. C4.5 yields a binary split if the selected variable is numerical; if it is categorical, the node is split into C subnodes, where C is the number of categorical values. (We use the adjective *numerical* for a variable that takes values on the real line and *categorical* for one that takes unordered values.) CHAID is similar to C4.5, but employs an additional step to merge some nodes. (This is called *value grouping* by some authors; see, e.g., Fayyad 1991 for other grouping methods.) FIRM extends the CHAID concept to numerical variables by initially dividing the range of each variable into 10 intervals.

There is little discussion in the literature on the merits of binary versus multiway splits. On one hand, a tree with multiway splits always can be redrawn as a binary tree. Thus there may seem to be no advantage in multiway splits. To see that this conclusion is not necessarily true, consider a dataset from Rouncefield (1995) that contains information on six 1990 demographic variables for 97 countries. Table 1 lists the variables and their definitions. The `class` variable takes six values: (i) Eastern Europe (*EE*), (ii) South America and Mexico (*SAM*), (iii) Western Europe, North America, Japan, Australia, and New Zealand (*WAJA*), (iv) Middle East (*MEast*), (v) *Asia*, and (vi) *Africa*.

Figure 1 shows a tree that predicts `class` from the six demographic variables in Table 1. It is obtained by the CRUISE algorithm, which is described later. The root node is split on `birth` into four subnodes. Two subnodes are terminal and two are split on `gnp`. We see that Eastern European (*EE*) and industrialized countries (*WAJA*) have low birth rates and African countries have high birth rates. Further, *WAJA* countries have higher `gnp` values than *EE* countries. An equivalent binary tree representation is given in Figure 2. Owing to its greater depth, more conditions must be considered in tracing a path from the root node to the lowest terminal node. Thus more effort may be needed to fully understand a binary tree than one with multiway splits. (For some ideas on simplifying a tree to enhance its interpretability, see Utgoff, Berkman, and Clouse 1997 and Zhang 1998.)

There are other advantages of multiway splits that are often overlooked. They can be seen by examining the binary CART tree in Figure 3. The figure actually shows two trees—the 0-SE tree, which is the full tree, and the 1-SE tree, which is the subtree with black terminal nodes. Breiman et al. (1984) defined the 0-SE tree as the tree with the smallest cross-validation (CV) estimate of error and the 1-SE tree as the smallest subtree with CV estimate of error within 1 standard error of the minimum. The trees demonstrate two common features when there are many classes. First, the predictions for some classes (namely, *Africa, Asia*, and *SAM*) are spread over two or more terminal nodes. This is harder to interpret than if each class is assigned to as few terminal nodes as possible. Second, the 1-SE tree does not predict the *SAM* class. Therefore, if we want every class to be predicted, we have to settle for the more complicated 0-SE tree.

These difficulties may be traced to the interaction between binary splits, pruning, and J, the number of classes. The larger the value of J, the more terminal nodes are required to ensure that there is at least one for every class. However, because each split produces only two nodes, this requires more splits, which increases the chance that some class predictions are

Table 1. Variables for Demographic Data

Variable	*Definition*
`birth`	Live birth rate per 1,000 of population
`death`	Death rate per 1,000 of population
`infant`	Infant deaths per 1,000 of population under 1 year old
`male`	Life expectancy at birth for males
`female`	Life expectancy at birth for females
`gnp`	Gross national product per capita in U.S. dollars
`class`	Country group

Hyunjoong Kim is Assistant Professor, Department of Mathematical Sciences, Worcester Polytechnic Institute, Worcester, MA 01609-2280 (E-mail: *hkim@wpi.edu*). Wei-Yin Loh is Professor, Department of Statistics, University of Wisconsin, Madison, WI 53706-1685 (E-mail: *loh@stat.wisc.edu*). This work was partially supported by U.S. Army Research Office grants DAAH04-94-G-0042 and DAAG55-98-1-0333. The authors are grateful to two reviewers for their constructive and encouraging comments.

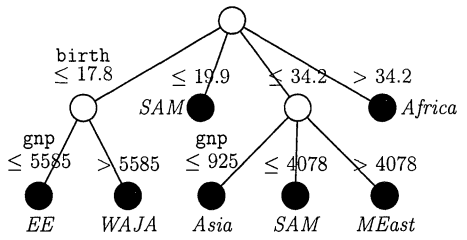

Figure 1. CRUISE 2D Tree for Demographic Data. The cross-validation estimate of the misclassification error is .31. The 0-SE and 1-SE trees are the same.

spread over several terminal nodes. Pruning can alleviate this, but as the 1-SE tree in Figure 3 shows, it can remove so many branches that some classes are not predicted.

All existing algorithms for multiway splits are inadequate in some ways. CHAID is inapplicable to numerical variables. FACT and FIRM do not prune. C4.5 produces multiway splits only for categorical variables and without value grouping. More importantly, all the algorithms have selection bias: if the predictor variables are independent of the class variable, they do not have the same chance to be selected for splitting. FACT, for example, is biased toward categorical variables and FIRM is biased toward numerical variables. Therefore, when a variable appears in a split, it is hard to know if the variable is indeed the most important or if the selection is due to bias. This undermines the inferential ability of the tree.

Doyle (1973), White and Liu (1994), and Loh and Shih (1997) warned about selection bias in greedy search algorithms when variables differ greatly in their numbers of splits. There is, however, another source of bias when variables differ in their proportions of missing values. To illustrate, consider the dataset in Lock (1993) on 93 new cars for the 1993 model year. Table 2 lists the variables: 19 are numerical, 3 are categorical, and 2 are binary. The class variable is type of car: small (*sml*), sporty (*spt*), compact (*cmp*), midsize (*mid*), large (*lrg*), and *van*. Figure 4 shows the CART 0-SE and 1-SE trees. The dominance of `luggage` in the splits is striking, especially because many of the other variables are expected to have similar discriminatory power. It turns out that `luggage` has the most number of missing values by far. We show in Section 5 that CART is biased toward selecting variables with more missing values. This problem is not unique to CART. In the design of an algorithm, care must be taken to consider the effect of missing values on selection bias as well.

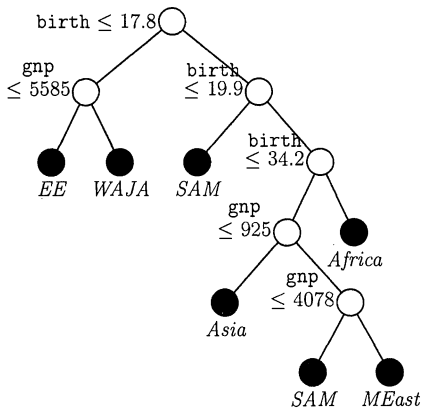

Figure 2. Tree From Figure 1 Reformatted With Binary Splits.

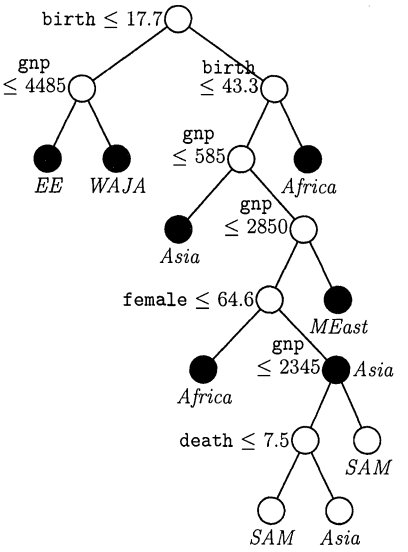

Figure 3. CART 0-SE Tree for Demographic Data. The CV estimate of error is .30. At each split, a case goes down the left branch if the condition is satisfied. Terminal nodes of the 1-SE tree are marked by black circles; it does not predict SAM.

Motivated by the foregoing examples, we propose here a new algorithm called CRUISE (for *classification rule with unbiased interaction selection and estimation*) that splits each node into as many as J subnodes. It borrows and improves upon ideas from many sources, but especially from FACT, QUEST, and GUIDE (Loh 2001) for split selection and CART for pruning. CRUISE has the following desirable properties.

Table 2. Variables for Car Data

Variable	*Definition*
`manuf`	Manufacturer (31 categories)
`minprice`	Minimum price in $1,000s
`midprice`	Midrange price in $1,000s
`maxprice`	Maximum price in $1,000s
`citympg`	City miles per gallon
`hwympg`	Highway miles per gallon
`airbag`	Air bags standard (3 categories)
`drtrain`	Drive train type (3 categories)
`cylin`	Number of cylinders
`enginsz`	Engine size in liters
`hp`	Horsepower
`rpm`	Revolutions per minute
`rev`	Engine revolutions per mile
`manual`	Manual transmission (yes, no)
`fuel`	Fuel tank capacity in gallons
`passngr`	Passenger capacity
`length`	Length in inches
`wheelbase`	Wheelbase length in inches
`width`	Width in inches
`uturn`	U-turn space in feet
`rearseat`	Rear seat room in inches
`luggage`	Luggage capacity in cubic feet
`weight`	Weight in pounds
`domestic`	U.S. or non-U.S. manufacturer

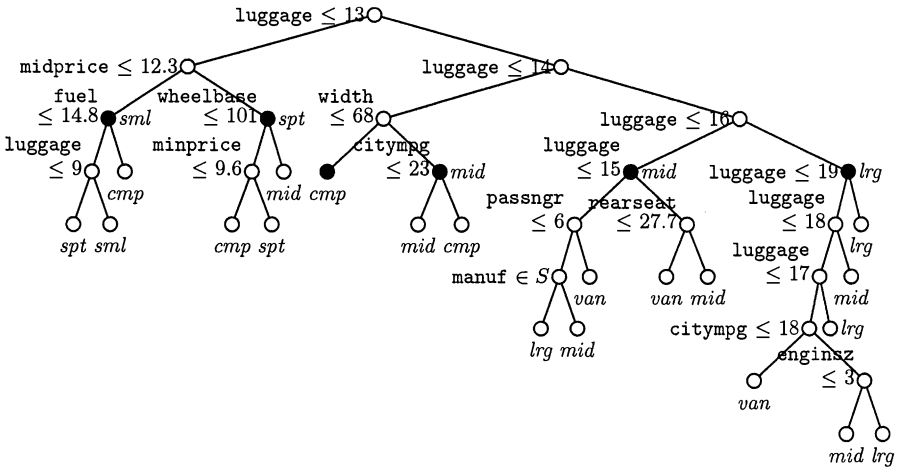

Figure 4. CART 0-SE Tree for Car Data. The CV estimate of error is .45. S contains the manufacturers Chrysler, Eagle, Ford, Geo, Lincoln, Mazda, Mitsubishi, Plymouth, Pontiac, Saturn, Subaru, Suzuki, and Volkswagen. Terminal nodes of the 1-SE tree are marked by black circles; the 1-SE tree does not predict the van class.

1. Its trees often have prediction accuracy at least as high as those of CART and QUEST, two highly accurate algorithms according to a recent study (Lim, Loh, and Shih 2000).
2. It has fast computation speed. Because it employs multiway splits, this precludes the use of greedy search methods.
3. It is practically free of selection bias. QUEST has little bias when the learning sample is complete, but it produces binary splits.
4. It is sensitive to local interactions between variables. This produces more intelligent splits and shorter trees. No previous algorithm is designed to detect local interactions.
5. It has all the preceding properties with or without missing values in the learning sample.

The rest of this article is organized as follows. Section 2 discusses univariate splits, where each split involves only one variable. Two variable selection methods designed to minimize selection bias are presented and simulation results on their effectiveness are reported. Section 3 extends the approach to linear combination splits, which have greater flexibility and prediction accuracy. Section 4 compares the prediction accuracy and computational speed of CRUISE against more than 30 methods on 32 datasets without missing values. The results show that CRUISE has excellent speed and that differences in mean misclassification rates between it and the best method are not statistically significant. Section 5 considers the problems created by missing values. We explain why they cause a bias in CART and how CRUISE deals with them. The algorithms are compared on 13 more datasets that contain missing values. Section 6 concludes the article with some summary remarks. A few algorithmic details are given in the Appendix.

2. UNIVARIATE SPLITS

Loh and Shih (1997) showed that the key to avoiding selection bias is separation of variable selection from split point selection. That is, to find a binary split of the form $X \in S$, first choose X and then search for the set S. This differs from the greedy search approach of simultaneously finding X and S to minimize some node impurity criterion. The latter results in selection bias when some X variables permit few splits, while others allow many. We, therefore, first deal with the problem of how to select X in an unbiased manner.

2.1 Selection of Split Variable

We propose two methods of variable selection. The first method (called 1D) is borrowed from QUEST. The idea is to compute p values from analysis of variance (ANOVA) F tests for numerical variables and contingency table χ^2 tests for categorical variables, and to select the variable with the smallest p value. In the event that none of the tests is significant, a Bonferroni-corrected Levene (1960) test for unequal variance among the numerical variables is carried out. The procedure is approximately unbiased in the sense that if the predictor variables and the class variable are mutually independent, each variable has approximately the same probability of being selected. Algorithm 1 in the Appendix describes the method in detail.

A weakness of this method is that it is designed to detect unequal class means and variances in the numerical variables. If the class distributions differ in other respects, it can be ineffective. Two examples are given in Figure 5. The left panel shows the distributions of two classes along one predictor variable. One distribution is normal and the other is exponential, but their means and variances are the same. The ANOVA and Levene tests will not find this variable significant. The right panel shows another two-class problem where there is a checkerboard pattern in the space of two variables. One class is uniformly distributed on the white and the other on the gray squares. The ideal solution is to split on one variable followed by splits on the other. Unfortunately, because the ANOVA and Levene tests do not look ahead, they most likely would select a third variable for splitting.

Loh (2001) suggests a way to detect pairwise interactions among the variables in regression trees. We extend it to the

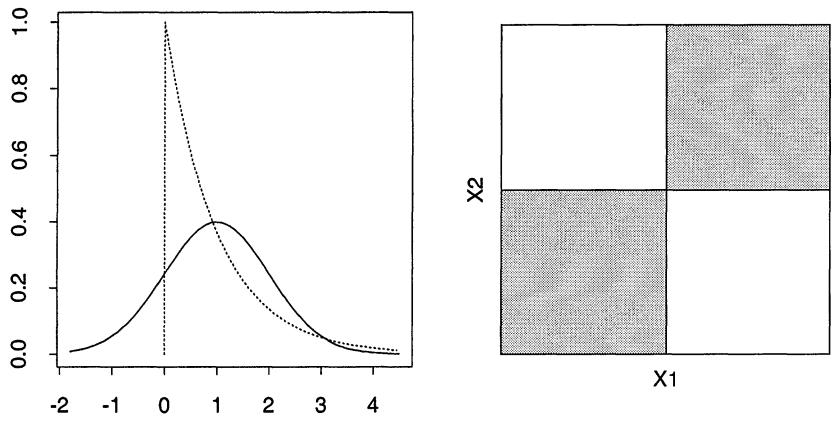

Figure 5. Two Examples Where the ANOVA and Levene Tests Fail. The left panel shows two univariate class densities with the same means and variances. The right panel shows a bivariate domain where one class is uniformly distributed over the white and the other over the gray squares.

classification problem here. The idea is to divide the space spanned by a pair of variables into regions and cross-tabulate the data using the regions as columns and the class labels as rows. In the right panel of Figure 5, for example, we can divide the (X_1, X_2) space into four regions at the sample medians and form a 2×4 contingency table for the data. The Pearson χ^2 test of independence will be highly significant. If both X_1 and X_2 are categorical variables, their category value pairs may be used to form the columns. If one variable is numerical and the other is categorical, the former can be converted into a two-category variable by grouping its values according to whether they are larger or smaller than their sample median. To detect marginal effects such as that in the left panel of Figure 5, we apply the same idea to each variable singly. If the variable is categorical, its categories form the columns of the table. If the variable is numerical, the columns are formed by dividing the values at the sample quartiles. Thus a set of marginal tables and a set of pairwise tables are obtained. The table with the most significant p value is selected. If it is a marginal table, the associated variable is selected to split the node. Otherwise, if it is a pairwise table, we can choose the variable in the pair that has the smaller marginal p value.

Loh (2001) showed that for regression trees, this approach is slightly biased toward categorical variables, especially if some take many values. He used a bootstrap method to correct the selection bias. To avoid overcorrecting the bias when it is small, he increased the bias before correction by selecting the categorical variable if the most significant p value was due to a pairwise table that involved one numerical and one categorical variable. We follow a similar approach here and call it the 2D variable selection method. A full description of the procedure is given in Algorithms 2 and 3 in the Appendix.

2.2 Selection of Split Points

Once X is selected, we need to find the sets of X values that define the split. If X is a numerical variable, FACT applies linear discriminant analysis (LDA) to the X values to find the split points. Because LDA is most effective when the data are normally distributed with the same covariance matrix, CRUISE performs a Box–Cox transformation on the X values before application of LDA. (See Qu and Loh 1992 for some theoretical support for Box–Cox transformations in classification problems.) If X is a categorical variable, it is first converted to a 0–1 vector. That is, if X takes values in the set $\{c_1, c_2, \ldots, c_m\}$, we define a vector $D = (D_1, D_2, \ldots, D_m)$ such that $D_i = 1$ if $X = c_i$ and $D_i = 0$ if $X \neq c_i$. The D vectors are then projected onto the largest discriminant coordinate (crimcoord). Finally, the Box–Cox transformation is applied to the crimcoord values. Because the Box–Cox transformation requires the data to be positive valued, we add $2x_{(2)} - x_{(1)}$ to the X values if $x_{(1)} \leq 0$. Here $x_{(i)}$ denotes the ith order statistic of the X or crimcoord values. The details are given in Algorithm 4 in the Appendix. After the split points are computed, they are back-transformed to the original scale.

The preceding description is for the 2D method. Selection of split points in the 1D method is the same, except that the Box–Cox transformation is not carried out if the variable is selected by Levene's test. Instead, the FACT method is used, that is, the partitions are found by applying LDA to the absolute deviations about the sample grand mean at the node.

Owing to its parametric nature, LDA sometimes can yield one or more intervals that contain no data points. When this occurs, we divide each empty interval into two halves and combine each half with its neighbor. A rarer situation occurs when large differences among the class priors cause LDA to predict the same class in all the intervals. In this event, the LDA partitions are ignored and the split points are simply taken to be the midpoints between successive class means.

2.3 Comparison of Selection Bias

A simulation experiment was carried out to compare the selection bias of the 1D and 2D methods with that of CART. The experiment is restricted to the two-class problem to avoid the long computation times of greedy search when there are more than two classes and some categorical variables take many values. Tables 3 and 4 define the simulation models. The learning sample size is 1,000 and class priors are equal.

First we consider selection bias at the root node in the null case with three numerical and two categorical variables, each being independent of the class variable. Table 5 shows the results when the predictor variables are mutually independent. The CART selection probability for the categorical variable X_5 grows steadily as k, its number of categories, increases. When

Table 3. Distributions Used in Simulation Study of Selection Bias

Variable	*Definition*
Z	Standard normal variable
E	Exponential variable with unit mean
B	Beta variable with density proportional to $x^4(1-x)^2$, $0 < x < 1$
C_k	Categorical-valued variable uniformly distributed on the integers $1, 2, \ldots, k$
D_k	Numerical variable uniformly distributed on the integers $1, 2, \ldots, k$
U	Uniform variable on the unit interval
R	Independent copy of U

$k = 5$, the probability is about .1, half the unbiased value of .2. When $k = 10$, the probability is close to .5 and when $k = 20$, it is .9. On the other hand, the probabilities for the 1D and 2D methods all lie within two simulation standard errors of .2.

To examine the effect of dependence among the variables, another experiment was conducted with correlated variables. The precise form of dependence is given in the first column of Table 6. Variables X_1 and X_2 are correlated, with their correlation controlled by a parameter δ such that $\text{corr}(X_1, X_2) = \delta/\sqrt{1+\delta^2}$. As δ varies from 0 to 10, the correlation increases from 0 to .995. The joint distribution of X_4 and X_5 is given in Table 7. The results in Table 6 show that CART is still heavily biased toward X_5. The 1D and 2D methods are again relatively unbiased, although only 2D has all its probabilities within 2 standard errors of .2.

Selection bias in the null case is harmless if pruning or a direct stopping rule yields a trivial tree. A trivial tree is worthless, however, in nonnull situations. To observe how often this occurs when the 1-SE tree is used, a third simulation experiment was carried out with mutually independent variables. Misclassification rates are estimated with independent test samples. For the shift and asymmetric models defined in Table 4, the selection probability for X_1 should be high, because it is the only variable associated with the class variable. For the interaction model, either X_1 or X_2 should be selected with high probability. Tables 8, 9, and 10 give the results as k, the number of categories in X_5, increases. They show that the following statements hold:

1. The selection bias of CART does not disappear with pruning even in the null case. Table 8 shows that about 40% of the CART trees have at least one split.
2. For CART, the probability of a nontrivial tree decreases slowly as k increases, but the conditional probability that the noise variable X_5 is selected increases quickly with k. This holds for the null and nonnull models. Hence, large values of k tend to produce no splits, but when a split does occur, it is likely to be on the wrong variable.
3. There is no evidence of selection bias in the 1D and 2D methods for the null model, either unconditionally or conditionally on the event of a nontrivial tree.
4. Table 9 shows that the 1D method has more power than the 2D method in selecting X_1 in the shift model, but 1D is worse than 2D in the asymmetric model and, as expected, in the interaction model (Table 10).
5. Only the 2D method selects the right variables with high probability in the interaction model. The other methods could not detect the interaction between X_1 and X_2.
6. The average sizes of the nontrivial trees are fairly similar among the methods.
7. The misclassification rates are also fairly similar, except at the interaction model where the 2D method is slightly more accurate.

Table 4. Models for Simulation Experiment on the Effect of Pruning

Model	*Class*	*Distributions of X_1*
Null	1	Z
	2	Z
Shift	1	$Z+.2$
	2	Z
Asymmetric	1	$(Z-1)I(U > .5)+(1.5Z+1)I(U \le .5)$
	2	$1.5Z$
Interaction	1	$ZI(X_2 > .5)+(Z+.5)I(X_2 \le .5)$
	2	$ZI(X_2 \le .5)+(Z+.5)I(X_2 > .5)$

NOTE: Variables are mutually independent with $X_2 \sim R$, $X_3 \sim E$, $X_4 \sim B$, and $X_5 \sim C_k$, as defined in Table 3.

3. LINEAR COMBINATION SPLITS

Trees with linear combination splits usually have better prediction accuracy because of their greater generality. They also tend to have fewer terminal nodes, although this does not translate to improved interpretation, because linear combination splits are much harder to comprehend. To extend the CRUISE approach to linear combination splits, we follow the FACT method, but add several enhancements. First, each categorical variable is transformed to a dummy vector and then projected onto the largest discriminant coordinate. This maps each categorical variable into a numerical one. After all categorical variables are transformed, a principal component analysis of the correlation matrix of the variables is carried out. Principal components with small eigenvalues are dropped to reduce the influence of noise variables. Finally, LDA is applied to the remaining principal components to find the split. Unlike the linear combination split methods of CART and QUEST, which divide the space in each node with a hyperplane, this method divides it into polygons, with each polygon being a node associated with a linear discriminant function. As in the case of univariate splits, the class assigned to a terminal node is the one that minimizes the misclassification cost, estimated from the learning sample. Whereas FACT breaks ties randomly, we choose among the tied classes those that have not been assigned to any sibling nodes.

Another departure from the FACT algorithm occurs when a split assigns the same class to all its nodes. Suppose, for example, that there are J classes, labeled $1, 2, \ldots, J$. Let $d_1, d_2, \ldots, d_J$ be the J linear discriminant functions induced

Table 5. Estimated Probabilities of Variable Selection for the Two-Class Null Case Where the Variables Are Mutually Independent

	CART				1D				2D			
k	5	10	15	20	5	10	15	20	5	10	15	20
$X_1 \sim Z$	.41	.25	.12	.05	.20	.20	.22	.20	.20	.19	.21	.20
$X_2 \sim E$	.42	.26	.12	.05	.23	.20	.21	.21	.21	.20	.20	.19
$X_3 \sim D_4$	.04	.02	.01	.00	.21	.21	.20	.20	.20	.20	.19	.20
$X_4 \sim C_2$	.02	.01	.01	.00	.19	.19	.18	.21	.17	.21	.19	.21
$X_5 \sim C_k$	.11	.46	.74	.90	.18	.20	.20	.19	.21	.21	.21	.20

NOTE: X_1, X_2, and X_3 are numerical, and X_4 and X_5 are categorical variables. Estimates are based on 1,000 Monte Carlo iterations and 1,000 samples in each iteration. Simulation standard errors are about .015. A method is unbiased if it selects each variable with probability .2.

Table 6. Estimated Probabilities of Variable Selection for the Two-Class Null Case With Varying Degrees of Dependence Between X_1 and X_2

	CART			1D			2D		
δ	0	1	10	0	1	10	0	1	10
$X_1 \sim Z$	.27	.25	.24	.19	.18	.14	.20	.21	.20
$X_2 \sim E + \delta Z$	.29	.25	.20	.21	.19	.12	.20	.20	.21
$X_3 \sim D_4$	.02	.03	.04	.25	.19	.26	.22	.20	.20
$X_4 \sim \lfloor UC_{10}/5 \rfloor + 1$	.01	.01	.01	.18	.23	.25	.18	.18	.19
$X_5 \sim C_{10}$	.41	.46	.52	.17	.21	.23	.19	.21	.21

NOTE: The joint distribution of X_4 and X_5 is given in Table 7. Estimates are based on 1,000 Monte Carlo iterations and 1,000 samples in each iteration. A method is unbiased if it selects each variable with probability .2. Simulation standard errors are about .015. Only the 2D method has all its entries within two simulation standard errors of .2.

Table 7. Joint Distribution of Categorical Variables X_4 and X_5 in Tables 6 and 13

	X_5									
X_4	1	2	3	4	5	6	7	8	9	10
1	1/10	1/10	1/10	1/10	1/10	5/60	5/70	5/80	5/90	1/20
2	0	0	0	0	0	1/60	2/70	3/80	4/90	1/20

Table 8. Probabilities of Variable Selection at the Root Node for the Null Model Before and After Pruning

				Conditional on $A = \{\lvert\tilde{T}\rvert > 1\}$				
Method	k	$P(X_1)$	$P(A)$	$P(X_1)$	$P(X_5)$	$E\lvert\tilde{T}\rvert$	Misclass. rate	Time (s.)
CART	10	.17	.44	.16	.34	4.3	.49	6392
	15	.09	.42	.09	.62	3.8	.49	5457
	20	.04	.39	.03	.89	3.1	.49	5028
1D	10	.18	.38	.19	.20	4.0	.49	327
	15	.20	.37	.18	.20	3.9	.49	419
	20	.19	.37	.21	.20	4.0	.49	532
2D	10	.18	.34	.19	.18	4.0	.49	1378
	15	.22	.35	.22	.17	4.2	.49	1839
	20	.18	.37	.20	.18	4.4	.49	2301

NOTE: k denotes the number of categories in X_5. $P(X_i)$ is the probability that X_i is selected to split the node. $\lvert\tilde{T}\rvert$ is the number of terminal nodes and $E\lvert\tilde{T}\rvert$ is its expected value. $P(A)$ is the probability that $\lvert\tilde{T}\rvert > 1$. Results are based on 1,000 Monte Carlo iterations with 1,000 learning samples in each iteration. Misclassification rates are estimated from independent test samples of size 500. Times are measured on a DEC Alpha Model 500a UNIX workstation.

Table 9. Probabilities of Variable Selection at the Root Node for the Shift and Asymmetric Models Before and After Pruning

Method	k	$P(X_1)$	$P(A)$	Conditional on $A = \{\|\tilde{T}\| > 1\}$ $P(X_1)$	$P(X_5)$	$E\|\tilde{T}\|$	Misclass. rate	Time (s.)
				Shift model				
CART	10	.82	.54	.88	.05	2.9	.48	5433
	15	.68	.49	.76	.18	3.0	.48	4881
	20	.52	.46	.59	.40	3.1	.48	4512
1D	10	.93	.58	.95	.01	2.6	.47	293
	15	.91	.58	.96	.01	2.7	.47	350
	20	.93	.61	.96	.02	2.7	.47	414
2D	10	.81	.52	.90	.01	2.7	.48	1361
	15	.79	.52	.89	.02	2.7	.48	1788
	20	.81	.54	.90	.02	2.8	.48	2219
				Asymmetric model				
CART	10	.71	.66	.74	.13	3.8	.48	5828
	15	.58	.61	.62	.30	3.8	.48	5185
	20	.37	.57	.38	.58	3.7	.49	4832
1D	10	.35	.50	.44	.13	3.8	.49	316
	15	.36	.50	.46	.17	4.1	.48	404
	20	.34	.49	.41	.15	3.7	.49	496
2D	10	.65	.53	.71	.06	4.4	.48	1375
	15	.64	.53	.74	.07	4.5	.48	1807
	20	.65	.53	.73	.05	4.3	.48	2243

NOTE: Simulation standard errors are about .03. Misclassification rates are estimated from independent test samples of size 500. Times are measured on a DEC Alpha Model 500a UNIX workstation.

by a split. Denote their values taken at the ith case by $d_1(i), d_2(i), \ldots, d_J(i)$. Suppose that there is a class j' such that $d_{j'}(i) \geq d_j(i)$ for all i and j. Then all the nodes are assigned to class j'. This event can occur if class priors or misclassification costs are sufficiently unbalanced. Because such a split is not useful, we force a split between class j' and the class with the next largest average discriminant score. Let $\bar{d}_j$ be the average value of $d_j(i)$ in the node. Then $\bar{d}_{j'} \geq \bar{d}_j$ for all j. Let j'' be the class with the second largest value of $\bar{d}_j$ and define $c = \bar{d}_{j'} - \bar{d}_{j''}$. Now split the node with the discriminant functions $d_1, d_2, \ldots, d_J$ except that $d_{j'}$ is replaced with $d_{j'} - c$. This usually will produce two nodes that contain most of the observations, one for class j' and one for class j''. The nodes for the other classes will contain few observations. Whereas this procedure is likely to be unnecessary in nodes far down the tree (because they may be pruned later), it is carried out only if the number of cases in the node exceeds 10% of the total sample size.

4. PREDICTION ACCURACY AND TRAINING TIME

Lim et al. (2000) compared a large number of algorithms on 32 datasets in terms of misclassification error and training time. They found that POLYCLASS (Kooperberg, Bose, and Stone 1997), a spline-based logistic regression algorithm, has the lowest average error rate, although it is not statistically significant from that of many other methods. On the other hand, there are great differences in the training times, which range from seconds to days. That study includes two implementations of the CART univariate split algorithm—IND (Buntine 1992) and Splus (Clark and Pregibon 1993). In this section, we add CRUISE and Salford Systems' CART (Steinberg and

Table 10. Probabilities of Variable Selection at the Root Node for the Interaction Model for Nontrivial Pruned Trees

Method	k	$P(A)$	Conditional on $A = \{\|\tilde{T}\| > 1\}$ $P(X_1)$	$P(X_2)$	$P(X_5)$	$E\|\tilde{T}\|$	Misclass. rate	Time (s.)
CART	10	.59	.21	.20	.28	5.0	.48	5989
	15	.50	.11	.09	.64	4.2	.49	5407
	20	.45	.04	.04	.80	3.8	.49	5010
1D	10	.61	.27	.27	.13	4.6	.47	303
	15	.61	.27	.24	.15	4.7	.47	376
	20	.60	.27	.24	.13	4.8	.47	456
2D	10	.90	.48	.52	.00	5.3	.44	1288
	15	.89	.49	.51	.00	5.3	.44	1672
	20	.91	.51	.49	.00	5.3	.44	2051

NOTE: Simulation standard errors are about .03. Misclassification rates are estimated from independent test samples of size 500. Times are measured on a DEC Alpha Model 500a UNIX workstation.

Table 11. Classification Algorithms in Comparative Study; 0-SE Tree Used Where Applicable

Name	*Description*
CR1	CRUISE 1D
CR2	CRUISE 2D
CRL	CRUISE linear combination splits
CTU	Salford Systems CART univariate splits (Steinberg and Colla 1992)
CTL	Salford Systems CART linear combination splits
SPT	Splus-tree univariate splits (Clark and Pregibon 1993)
QTU	QUEST univariate splits (Loh and Shih 1997)
QTL	QUEST linear combination splits
FTU	FACT univariate splits (Loh and Vanichsetakul 1988)
FTL	FACT linear combination splits
IC	IND CART univariate splits (Buntine 1992)
IB	IND Bayes
IBO	IND Bayes with `opt` style
IM	IND Bayes with `mml` style
IMO	IND Bayes with `opt` and `mml` styles
C4T	C4.5 decision tree (Quinlan 1993)
C4R	C4.5 decision rules
OCU	OC1 tree, univariate splits (Murthy, Kasif, and Salzberg 1994)
OCL	OC1 with linear combination splits
OCM	OC1 with univariate and linear combination splits
LMT	LMDT linear combination split tree (Brodley and Utgoff 1995)
CAL	CAL5 decision tree (Müller and Wysotzki 1994)
T1	One-split tree (Holte 1993)
LDA	Linear discriminant analysis
QDA	Quadratic discriminant analysis
NN	Nearest neighbor
LOG	Polytomous logistic regression
FM1	FDA-MARS, additive model (Hastie, Tibshirani, and Buja 1994)
FM2	FDA-MARS, interaction model
PDA	Penalized discriminant analysis (Hastie, Buja, and Tibshirani 1995)
MDA	Mixture discriminant analysis (Hastie and Tibshirani 1996)
POL	POLYCLASS (Kooperberg et al. 1997)
LVQ	Learning vector quantization neural network (Kohonen 1995)
RBF	Radial basis function neural network (Sarle 1994)

Colla 1992), which allows linear combination splits, to the comparison. The list of algorithms and their acronyms are given in Table 11. The reader is referred to Lim et al. (2000) for details on the other algorithms and the datasets.

A plot of median training time versus mean error rate for each algorithm is given in the upper half of Figure 6. The training times are measured on a DEC 3000 Alpha model 300 workstation running the UNIX operating system. POLYCLASS (abbreviated as POL in the plot) still has the lowest mean error rate. As in Lim et al. (2000), we fit a mixed effects model with interactions to determine if the differences in mean error rates are statistically significant. The algorithms are treated as fixed effects and the datasets as random effects. This yields a p value less than .001 for a test of the hypothesis of equal algorithm effects. Using 90% Tukey simultaneous confidence intervals (Hochberg and Tamhane 1987, p. 81), we find that a difference in mean error rates less than .056 is not statistically significant from zero. This is indicated in the plot by a solid vertical line, which separates the algorithms into two groups: those that have mean error rates that do not differ statistically significantly from that of POL and those that do. All except seven algorithms fall on the left of the line. The dotted horizontal lines divide the algorithms into four groups according to median training time: less than 1 min,

Table 12. Probabilities of Variable Selection Where the Class Variable Is Independent of Five Mutually Independent Variables

	CART Percent missing X_1				*1D* Percent missing X_1				*2D* Percent missing X_1			
Distribution	*20*	*40*	*60*	*80*	*20*	*40*	*60*	*80*	*20*	*40*	*60*	*80*
$X_1 \sim Z$	.42	.55	.67	.78	.22	.20	.23	.21	.23	.18	.21	.23
$X_2 \sim E$	.20	.12	.11	.07	.21	.20	.20	.21	.17	.21	.20	.20
$X_3 \sim D_4$	.03	.01	.01	.01	.19	.20	.19	.20	.19	.19	.17	.17
$X_4 \sim C_2$	.01	.00	.01	.00	.20	.21	.21	.19	.19	.19	.20	.18
$X_5 \sim C_{10}$	.35	.31	.20	.14	.18	.20	.19	.20	.23	.23	.22	.22

NOTE: Notations are defined in Table 3. Only X_1 has missing values. Estimates are based on 1,000 Monte Carlo iterations and 1,000 samples in each iteration. A method is unbiased if it selects each variable with probability .2. Simulation standard errors are about .015.

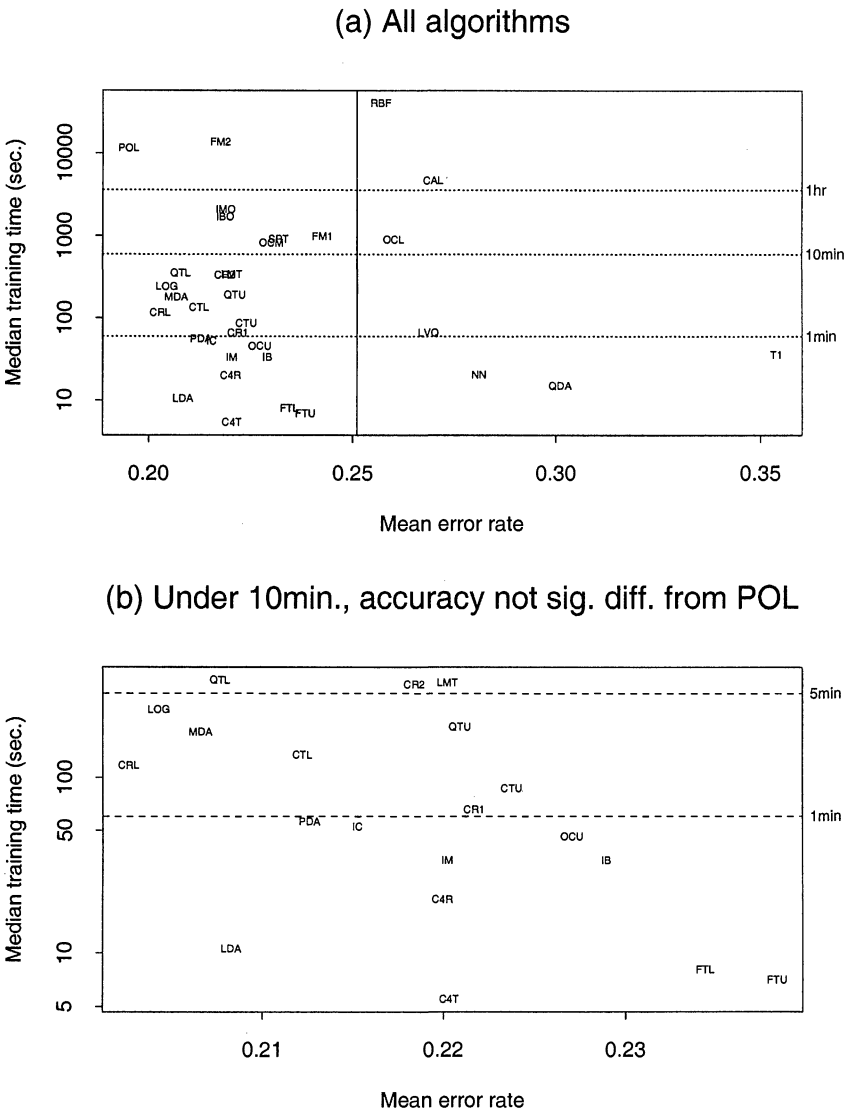

Figure 6. Median Training Time Versus Mean Error Rate. The vertical axis is in log scale. Algorithms to the left of the solid vertical line in plot (a) have mean error rates that are not statistically significant at the 10% simultaneous level from POL. *The subset of these algorithms that have median training time less than 10 min is shown in plot (b).*

1–10 min, 10 min to 1 h, and more than 1 h. POL has the third highest median training time. The CRUISE linear combination split algorithm (CRL) has the second lowest mean error rate, but takes substantially less time. The mean error rates of the 1D and 2D algorithms (CR1 and CR2) and Salford Systems CART (CTU and CTL) are also not statistically significant from POL. A magnified plot of the algorithms that are not statistically significant from POL and that require less than 10 min of median training time is shown in the lower half of Figure 6. The best algorithm in this group is CRL, followed closely by logistic regression.

5. MISSING VALUES

The discussion so far has assumed that there are no missing values in the data. We now extend the CRUISE method to allow missing values in the learning sample as well as in future cases to be classified. One popular solution for univariate split selection uses only the cases that are nonmissing in the variable under consideration. We call this the "available case" strategy. It is used in CART and QUEST. Another solution, used by FACT and QUEST, imputes missing values in the learning sample at each node and then treats all the data as complete.

After a split is selected, there is the problem of how to send a case with missing values through it. CART uses a system of "surrogate splits," which are splits on alternate variables. Others use imputation or send the case through every branch of the split. Quinlan (1989) compared these and other techniques on a nonpruning version of the C4.5 algorithm.

In this section, we show that missing values can contribute two additional sources of bias to the CART algorithm: in the selection of the main split and in the selection of the surrogates. We also consider some new unbiased missing value strategies and compare them with CART, QUEST, and C4.5 on some real datasets.

5.1 Bias of CART Split Selection

When CART evaluates a split of the form $X \in S$, it first restricts the learning sample to the set A of cases that are non-

Table 13. Probabilities of Variable Selection Where the Class Variable Is Independent of Five Dependent Variables

	CART				1D				2D			
Percent missing X_1	20	40	60	80	20	40	60	80	20	40	60	80
$X_1 \sim Z_1$	.40	.51	.65	.79	.24	.24	.24	.25	.22	.22	.23	.22
$X_2 \sim Z_2$	.12	.11	.06	.04	.16	.13	.14	.15	.18	.18	.19	.20
$X_3 \sim E + 10Z_2$	.11	.09	.09	.04	.13	.14	.15	.14	.19	.19	.20	.20
$X_4 \sim \lfloor UC_{10}/5 \rfloor + 1$	.01	.01	.00	.00	.23	.23	.25	.23	.19	.19	.18	.18
$X_5 \sim C_{10}$	.36	.28	.20	.13	.24	.27	.23	.22	.22	.23	.20	.21

NOTE: Z_1 and Z_2 are independent standard normals. The correlation between X_2 and X_3 is .995. The joint distribution of X_4 and X_5 is given in Table 7. Only X_1 has missing values. A method is unbiased if it selects each variable with probability .2. Estimates are based on 1,000 Monte Carlo iterations and 1,000 samples in each iteration. Simulation standard errors are about .015.

missing in X. Then, using a node impurity measure that is a function of the class proportions in A, it searches over all sets S to minimize the total impurity in the nodes. One problem with basing the impurity measure on proportions instead of sample sizes is that this creates a selection bias toward variables that possess larger numbers of missing values.

As an extreme example, consider a two-class problem where there is an X variable that is missing in all but two cases, so that A has only two members. Suppose that the cases take distinct values of X and they belong to different classes. Then any split on X that separates these two cases into different nodes will yield zero total impurity in the nodes. Since this is the smallest possible impurity, the split is selected unless there are ties.

To appreciate the extent of the selection bias in less extreme situations, we report the results of a simulation experiment with two classes and five variables that are independent of the class. The class variable has a Bernoulli distribution with probability of success .5. X_1 has randomly missing values; the other variables are complete. The relative frequency with which each variable is selected is recorded. If a method is unbiased, the probabilities should be .2. Two scenarios are considered, with one having mutually independent variables and another having dependent ones. The results are given in Tables 12 and 13, respectively. The dependence structure in Table 13 is the same as that in Table 6. Clearly, the selection bias of CART toward X_1 grows with the proportion of missing values.

Example 1. We saw in the car example in Figure 4 that the `luggage` variable is repeatedly selected to split the nodes. It turns out that `luggage` has the most missing values—11 out of 93. Only two other variables have missing values, namely, `cylin` and `rearseat`, with one and two missing, respectively. In view of the preceding results, it is likely that the selection of `luggage` is partly due to the bias of CART toward variables with more missing values. This conjecture is supported by one additional fact: all the vans in the dataset are missing the `luggage` variable, probably because the variable is not applicable to vans because they do not have trunks. To send the vans through the root node, the CART algorithm uses a surrogate split on `wheelbase`. However, because vans have similarly large `wheelbase` values, all of them are sent to the right node. This increases the proportion of missing values for `luggage` in the right node (from 11:93 to 11:57) and hence its chance of selection there too. It is interesting to note that CRUISE selects `wheelbase` to split the root node.

Example 2. Another example of the effect of missing values on selection bias is provided by the hepatitis dataset from the University of California, Irvine (UCI), Repository of Machine Learning Databases (Merz and Murphy 1996). There are 19 measurements on 155 cases of hepatitis, of which 32 cases are deaths. Six variables take more than two values on a numerical scale; the rest are binary. The variables and their number of missing values are given in Table 14. `Protime` has the highest percentage (43%) of missing values.

The small proportion of deaths makes it hard to beat the naive classifier that classifies every case as "live" (see, e.g., Diaconis and Efron 1983 and Cestnik, Konenenko, and Bratko 1987). In fact, the 1-SE trees from the CART, QUEST, and CRUISE 1D methods are trivial with no splits. To make the problem more interesting, we employ a 2:1 cost ratio, making the cost of misclassifying a "die" patient as "live" twice that of the reverse. [C4.5 does not allow unequal misclassi-

Table 14. Variables and Their Number of Missing Values in Hepatitis Data

Name	Type	Number missing
Person variables		
`Age`	Numerical	0
`Sex`	Binary	0
Medical test variables		
`Bilirubin`	Numerical	6
`Alk phosphate`	Numerical	29
`Sgot`	Numerical	4
`Albumin`	Numerical	16
`Protime`	Numerical	67
Symptom variables		
`Histology`	Binary	0
`Steroid`	Binary	1
`Antivirals`	Binary	0
`Fatigue`	Binary	1
`Malaise`	Binary	1
`Anorexia`	Binary	1
`Big liver`	Binary	10
`Firm liver`	Binary	11
`Spleen palpable`	Binary	5
`Spiders`	Binary	5
`Ascites`	Binary	5
`Varices`	Binary	5

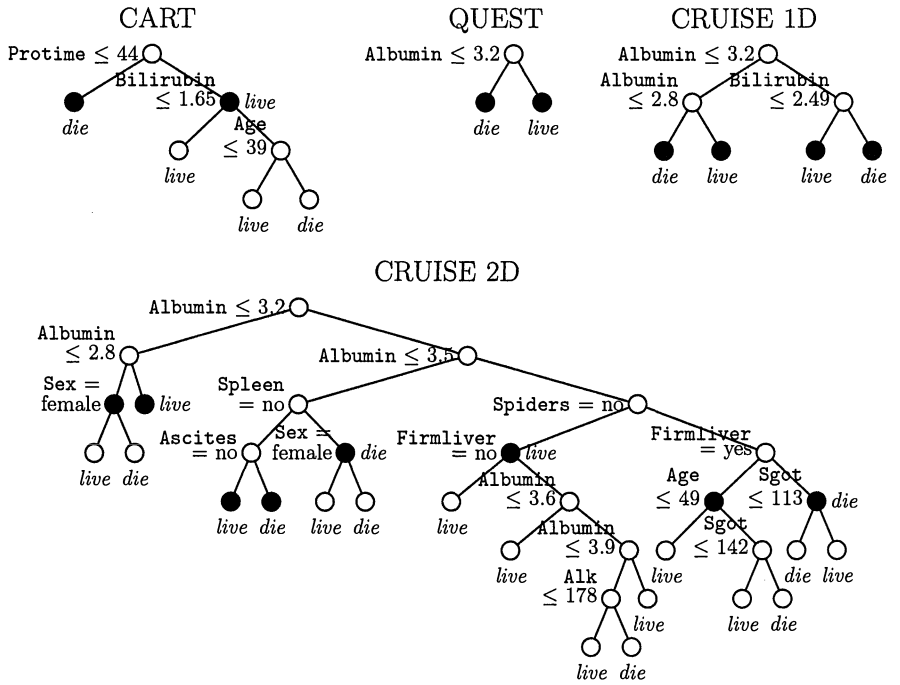

Figure 7. CART, QUEST, and CRUISE 1D and 2D 0-SE Trees for Hepatitis Data Based on 2:1 Misclassification Costs. Tenfold CV estimates of misclassification costs are .30, .30, .30, and .29, respectively. Terminal nodes of 1-SE trees are marked in black.

fication costs. CRUISE employs unequal costs in split point selection via LDA (see, e.g., Seber 1984, p. 285) and during cost-complexity pruning (Breiman et al. 1984).] Figure 7 shows the results. CART splits first on `Protime`. QUEST and CRUISE do not split on `Protime` at all. Instead they split first on `Albumin`. Figure 8 shows how the data are partitioned by the CART and CRUISE-1D 1-SE trees. Although the partitions appear to do a reasonable job of separating the classes, they can be misleading because cases with missing values are invisible. For example, only about half of the observations appear in the CART plot.

Because `Protime` has so many missing values, it is impossible to determine how much of its prominence in the CART tree is due to its predictive ability. On the other hand, the methods appear to be equally good in classifying future observations—10-fold cross-validation estimates of misclassification costs for the CART, QUEST, and CRUISE 1D and 2D methods are .30, .30, .30, and .29, respectively.

Breiman et al. (1984) gave a formula that ranks the overall "importance" of the variables based on the surrogate splits. According to their formula, the top three predictors are `Protime`, `Bilirubin`, and `Albumin`, in that order. We will see

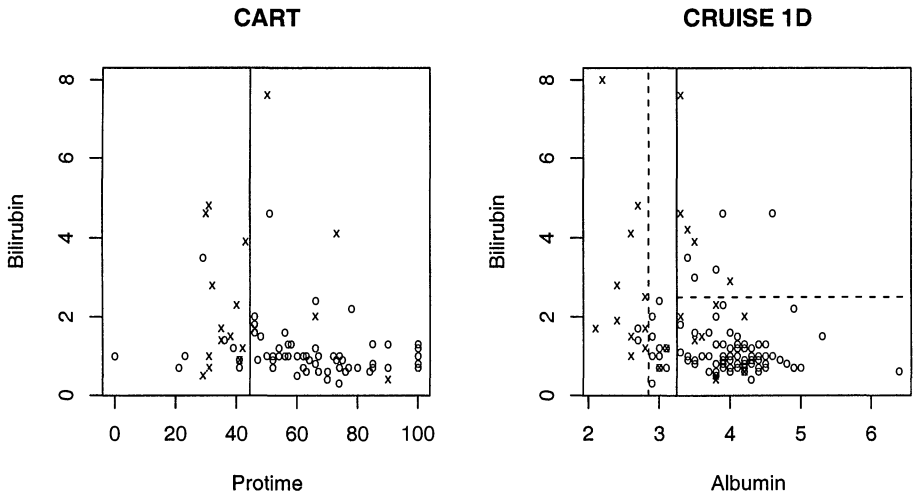

Figure 8. Partitions of the Hepatitis Data Produced by the 1-SE Trees From the CART and CRUISE 1D Methods. Circles and crosses denote the "live" and "die" cases, respectively. The different numbers of points in the plots are due to unequal numbers of missing values in `Protime` *and* `Albumin`.

in the next section that the ranking may be unreliable, because the surrogate splits have their own selection bias.

5.2 Bias of CART Surrogate Split Selection

The CART surrogate split technique is very intuitive. If a split s requires a value that is missing from an observation, it uses a split s' on another variable to usher it through. The surrogate split s' is found by searching over all splits to find the one that best predicts s, in terms of the number of cases nonmissing in the variables required in s and s' (Breiman et al., 1984, pp. 140–141). This creates another kind of selection bias. Suppose s and s' are based on variables X and X', respectively. Let n' be the number of cases with nonmissing values in both X and X' that are sent to the same subnodes by s and s'. If X' has many missing values, n' will be small, and, therefore, the desirability of s' as a surrogate for s will be low. As a result, variables with many missing values are penalized. Although it makes sense to exact some penalty on a variable for missing observations (CRUISE does this through the degrees of freedom in p-value calculations), the CART method overdoes it—all other things being equal, the more missing values a variable has, the lower its probability of selection as a surrogate variable.

To demonstrate this, we simulate data with variables $X_0, X_1, \ldots, X_5$ and a Bernoulli class variable Y with .5 success probability. Variable X_0 has a standard normal distribution if Y is 0 and a normal distribution with mean .3 and variance 1 if Y is 1. The other X variables are mutually independent standard normal and are independent of X_0 and Y. Only X_1 has missing values, which are randomly assigned according to a fixed percentage. We find the best split on X_0 and then observe how often surrogate splits on $X_1, X_2, \ldots, X_5$ are selected. Table 15 gives the results for sample size 200 (the CRUISE method uses an "alternate variable" strategy described in the next section). The proportions are based on 1,000 Monte Carlo iterations. The selection bias of CART begins to show when X_1 has 2% missing values. With 25% missing values, X_1 has almost no chance of being selected in a surrogate split.

Selection bias in surrogate splits is not a serious problem by itself. As long as the predictive accuracy of the tree is unaffected, the bias probably can be ignored. In the case of CART, however, the surrogate splits are used to rank the importance of the variables. This makes the ranking biased too.

5.3 CRUISE Missing Value Methods

We evaluated many different methods of handling missing values. Owing to space limitations, only the best are reported here.

5.3.1 Univariate Splits. If there are values missing in the learning sample, we use the "available case" solution, where each variable is evaluated using only the cases nonmissing in that variable at the node. The procedure for the 1D and 2D methods is as follows:

1. For method 1D, compute the p value of each X in Algorithm 1 from the nonmissing cases in X.
2. For method 2D, compute the p value of each pair of variables in Algorithm 2 from the nonmissing cases in the pair.
3. If X^* is the selected split variable, use the cases with nonmissing X^* values to find the split points.
4. If X^* is a numerical variable, use the node sample class mean to impute missing values in X^*. Otherwise, if X^* is categorical, use the class mode.
5. Pass the imputed sample through the split.
6. Delete the imputed values and restore their missing status.

To process a future case for which the selected variable is missing at a node t, we split on an alternate variable. The idea is similar in concept to the CART surrogate splits, but it is faster and appears to be unbiased. Let X be the most significant variable according to the variable selection algorithm and let s be the associated split. Let X' and s' be the second most significant variable and associated split.

1. If X' is nonmissing in the case, use s' to predict its class. Then impute the missing X value with the learning sample mean (if X is numerical) or mode (if X is categorical) of the nonmissing X values for the predicted class in t.
2. If X' is missing in the case, impute the missing X value with the grand mean or mode in t, ignoring the class.

After the case is sent to a subnode, its imputed value is deleted. We call this the *alternate variable strategy*. The simulation results on the right side of Table 15 show that this method has negligible bias.

Table 15. Estimated Probabilities of Surrogate–Alternate Variable Selection for the Null Case Where the Variables $X_1, X_2, \ldots, X_5$ Are Independent of the Main Split Variable X_0.

	CART Percent missing X_1					CRUISE Percent missing X_1				
Variable	1	2	3	4	25	1	2	3	4	25
X_1	.18	.12	.09	.05	.00	.19	.20	.18	.20	.18
X_2	.25	.25	.26	.24	.30	.18	.22	.18	.19	.19
X_3	.21	.23	.26	.27	.25	.22	.19	.20	.21	.19
X_4	.20	.23	.20	.23	.23	.22	.19	.22	.22	.21
X_5	.17	.17	.19	.21	.22	.20	.20	.22	.18	.23

NOTE: The variable X_1 has missing values, but others do not. Estimates are based on 1,000 Monte Carlo iterations and 200 samples in each iteration. Simulation standard errors are about .015. A method is unbiased if it selects each variable with probability .2.

Table 16. Datasets With Missing Values

Code	Description	Source	N	J	Variables #num	Variables #cat	m_1	m_2
bio	Biomedical data	UCI	209	2	5	0	7	1.4
bnd	Cylinder bands	UCI	540	2	19	18	49	5.0
car	1993 cars	Lock (1993)	93	6	19	5	12	0.6
crx	Credit approval	UCI	690	2	6	9	5	0.6
dem	Demography	Rouncefield (1995)	97	6	6	0	6	1.0
ech	Echo cardiogram	UCI	132	2	13	0	8	2.2
fsh	Fish catch	UCI	159	7	6	1	55	7.9
hco	Horse colic	UCI	366	3	9	17	98	20.3
hea	Heart disease	UCI	303	2	5	8	2	0.2
hep	Hepatitis	UCI	155	2	6	13	11	5.7
hin	Head injury	Hawkins (1997)	1000	3	0	6	41	9.8
imp	Auto imports	UCI	205	6	13	9	4	1.0
usn	College data	StatLib	1302	3	26	1	88	20.2

NOTE: The column labeled m_1 gives the percent of cases with one or more missing values, the column labeled m_2 gives the percent of missing values in the dataset.

5.3.2 Linear Combination Splits. It is often unwise to restrict the search for splits to the cases with nonmissing values in a linear combination split. In our experience, the best solution is imputation of missing values with the node mean or mode. This is the same strategy used in FACT and QUEST. The specific steps during tree construction are as follows:

1. Impute each missing value with the node class mean (numerical variable) or mode (categorical variable).
2. Split the node with the imputed sample.
3. Delete the imputed values and restore their missing status.

This procedure is inapplicable for directing a future case that contains missing values through the split because its class is unknown. Instead, we use a univariate split as an alternative to the selected linear combination split. Let X and s be the selected variable and split obtained with the 1D method at a node t.

1. If X is nonmissing in the case, use s to predict its class. Then impute all the missing values in the case with the means and modes of the numerical and categorical variables, respectively, for the predicted class.
2. If X is missing in the case, impute all missing values with the grand means or modes in t, ignoring the class.

After the case is sent to a subnode, its imputed values are deleted and their missing status is restored.

5.4 Comparison of Methods on Real Data With Missing Values

Thirteen real datasets are used to compare the missing value methods. They are listed in Table 16 with brief descriptions. Many are from UCI. Two (car and dem) were discussed in the Introduction. The hin data are from Hawkins (1997) and the usn data are from StatLib (http://lib.stat.cmu.edu). The percentage of cases with one or more missing values in the datasets, range from 2 to 98. (Note: Unit misclassification costs are employed in all except the hep dataset, where 2:1 costs are used. As mentioned earlier, C4.5 does not allow unequal costs. For the hep data, we calculate the misclassification cost of C4.5 by multiplying the misclassification errors with the appropriate costs.)

Tenfold cross-validation is used to estimate the misclassification costs. That is, each dataset is randomly divided into 10 roughly equal-sized subsets. One subset is held out and a classification tree is constructed from the other nine subsets. The holdout set is then applied to the tree to estimate its misclassification cost. This procedure is repeated 10 times by using a different holdout set each time. The average of the 10 cost estimates is reported in Table 17. The last two columns of the table give the estimated misclassification cost and the number of terminal nodes for each method, averaged across the datasets. (Note: The CART program failed on the imp dataset when the arcing option was selected. The average misclassification cost for this method is therefore based on the other 12 datasets.)

The following conclusions are apparent from the results:

1. The univariate split methods have nearly the same average misclassification costs.
2. The misclassification costs of the CRUISE linear combination split method are on average about 12% lower than the univariate split methods. Surprisingly, the CART linear method has higher average misclassification cost than the univariate methods. This is opposite to the results for nonmissing data observed in Section 4.
3. CART trees tend to have fewer terminal nodes than CRUISE, with QUEST in between. C4.5 trees have, on average, twice as many terminal nodes as CART. This is consistent with the results of Lim et al. (2000), who studied datasets without missing values.
4. Except for CART, trees with linear combination splits tend to have substantially fewer terminal nodes than their univariate counterparts. The CART trees with linear combination splits are, on average, about the same size as its univariate trees.
5. The last line of Table 17 gives the results for CART univariate splits with the arcing option. Instead of one tree, an ensemble of 50 trees is constructed from random perturbations of the learning sample. It has been observed elsewhere in the literature (Breiman 1998) that arcing tends to decrease the average misclassification cost of CART univariate trees. The method does

Table 17. Tenfold Cross-Validation Estimates of Misclassification Costs ($|\widetilde{T}|$ denotes number of terminal nodes)

	Datasets													Mean	
Method	*bio*	*bnd*	*car*	*crx*	*dem*	*ech*	*fsh*	*hco*	*hea*	*hep*	*hin*	*imp*	*usn*	*Cost*	$\lvert\widetilde{T}\rvert$
							Univariate splits								
1D	.15	.27	.20	.14	.33	.34	.17	.33	.22	.30	.28	.19	.29	.25	16.6
2D	.16	.28	.27	.15	.31	.27	.16	.37	.23	.29	.30	.22	.30	.25	16.7
QUEST	.13	.25	.16	.15	.35	.31	.17	.34	.26	.30	.27	.29	.30	.25	14.4
CART	.16	.22	.45	.15	.30	.36	.20	.29	.22	.30	.31	.20	.28	.26	12.0
C4.5	.14	.33	.24	.15	.29	.37	.21	.30	.28	.31	.29	.20	.29	.26	25.6
							Linear combination splits								
CRUISE	.11	.20	.31	.14	.31	.24	.01	.30	.16	.22	.27	.29	.33	.22	5.6
QUEST	.09	.21	.41	.15	.38	.26	.05	.30	.16	.22	.26	.37	.31	.24	6.1
CART	.14	.23	.25	.16	.33	.38	.17	.32	.26	.38	.31	.27	.32	.27	11.6
							Univariate splits with arcing								
CART	.14	.20	.28	.15	.32	.37	.16	.29	.21	.26	.30	—	.30	.24	NA

not appear to be more accurate than the QUEST and CRUISE linear combination split methods here.

Table 18 reports the training time (summed over the 10 cross-validation trials) for each method. Despite the great variability of times between methods and datasets, some patterns can be discerned:

1. Consistent with the results for nonmissing data in Section 4, C4.5 is the fastest.
2. The CRUISE 2D method is often the slowest.
3. The speed of the CRUISE 1D method is comparable to that of CART and QUEST.
4. Among linear combination split methods, CART is fastest on six datasets and slowest on four datasets. The CRUISE linear combination split method is fastest on seven datasets and never the slowest.
5. The CART arcing option is slower than all the linear combination split methods on eight datasets. It is slower than the CRUISE linear method on all but one dataset.

6. CONCLUDING REMARKS

There are two non-mutually-exclusive reasons for using a classification tree. One is to infer qualitative information about the learning sample from the splits and another is to classify future observations. The former is unique to tree methods and is what makes them so appealing. On the other hand, owing to dependencies among variables, there is typically more than one correct way to describe a dataset with a tree structure. Thus it is advantageous to compare trees generated by different algorithms.

To provide useful information, the tree structure must be easy to understand and there must not be biases in the selection of the splits. CRUISE uses two techniques to improve the interpretability of its trees. First, it splits each node into multiple subnodes, with one for each class. This reduces tree depth. Second, it selects variables based on one-factor and two-factor effects. Therefore, where other methods would fail, CRUISE can immediately identify a variable with a significant two-factor interaction even when it does not have a significant one-factor effect.

More important than tree depth is absence of selection bias, because the latter can undermine our confidence in the interpretation of a tree. We saw that some algorithms can be severely biased if variables have unequal numbers of splits or possess different proportions of missing values. CRUISE solves this problem with a two-step approach. First, it uses the

Table 18. Training Times (in seconds) on a DEC 3000 Alpha 300 Workstation

Method	*bio*	*bnd*	*car*	*crx*	*dem*	*ech*	*fsh*	*hco*	*hea*	*hep*	*hin*	*imp*	*usn*	*Median*
							Univariate splits							
1D	11	2711	16	61	11	11	11	208	33	16	143	44	220	33
2D	16	20399	385	682	16	22	44	2458	143	88	330	737	2596	330
QUEST	9	1254	55	101	16	10	27	220	33	18	269	53	357	53
CART	30	176	58	63	33	28	38	95	44	48	70	58	149	58
C4.5	2	49	27	5	16	2	27	33	3	2	33	4	44	16
							Linear combination splits							
CRUISE	11	6638	33	253	5	11	5	245	66	27	225	71	533	66
QUEST	13	3877	55	219	16	13	27	335	59	28	610	74	720	59
CART	36	194	65	122	36	32	45	96	59	50	71	80	202	65
							Univariate splits with arcing							
CART	75	1508	134	307	73	79	41	457	140	124	369	164	630	140

p values from significance tests to select variables. This avoids the bias of the greedy search approach caused by variables with unequal numbers of splits. It also automatically accounts for unequal numbers of missing values through the degrees of freedom. Then CRUISE uses a bootstrap bias correction to further reduce the bias due to differences between numerical and categorical variables. The bootstrap correction is critical because the amount of bias is dependent on many aspects of the data, such as sample size, number and type of variables, missing value pattern, and configuration of the data points.

With regard to classification of future observations, there exist many tree and nontree methods with excellent computational speed and classification accuracy. Our results show that CRUISE is among the best.

The CRUISE computer program may be obtained from `http://www.wpi.edu/~hkim/cruise/` or `http://www.stat.wisc.edu/~loh/cruise.html`.

APPENDIX: ALGORITHMIC DETAILS

Algorithm 1 (1D)

Let α be a selected significance level (default is .05). Suppose $X_1, \ldots, X_{K_1}$ are numerical and $X_{K_1+1}, \ldots, X_K$ are categorical variables.

1. Carry out an ANOVA analysis on each numerical variable and compute its p value. Suppose X_{k_1} has the smallest p value $\hat{\alpha}_1$.
2. For each categorical variable, form a contingency table with the categorical values as rows and class values as columns, and find its χ^2 p value. Let the smallest p value be $\hat{\alpha}_2$ and let the associated variable be X_{k_2}.
3. Define
$$k' = \begin{cases} k_1, & \hat{\alpha}_1 \le \hat{\alpha}_2, \\ k_2, & \hat{\alpha}_1 > \hat{\alpha}_2. \end{cases}$$
4. If $\min(\hat{\alpha}_1, \hat{\alpha}_2) < \alpha/K$ (first Bonferroni correction), choose $X_{k'}$ as the split variable.
5. Otherwise, find the p value for Levene's F test on absolute deviations about the class mean for each numerical variable. Suppose $X_{k''}$ has smallest p value $\tilde{\alpha}$.
   (a) If $\tilde{\alpha} < \alpha/(K+K_1)$, choose $X_{k''}$ (second Bonferroni correction).
   (b) Otherwise, choose $X_{k'}$.

Algorithm 2 (2D)

Suppose $X_1, \ldots, X_{K_1}$ are numerical and $X_{K_1+1}, \ldots, X_K$ are categorical variables. Let J_t be the number of classes represented at node t.

1. Marginal test for each numerical variable X:
   (a) Divide the data into four groups at the sample quartiles of X.
   (b) Construct a $J_t \times 4$ contingency table with classes as rows and groups as columns.
   (c) Compute the Pearson χ^2 statistic with $\nu = 3(J_t - 1)$ degrees of freedom.
   (d) Convert χ^2 to an approximate standard normal value with the Peizer–Pratt transformation
$$z = \begin{cases} |W|^{-1}(W - 1/3)\sqrt{(\nu-1)\log\{(\nu-1)/\chi^2\} + W}, & \nu > 1, \\ \sqrt{\chi^2}, & \nu = 1, \end{cases} \quad \text{(A.1)}$$
   where $W = \chi^2 - \nu + 1$.

   Let z_n denote the largest among the K_1 z values.
2. Marginal test for each categorical variable X: Let C denote the number of categories of X.
   (a) Construct a $J_t \times C$ contingency table with classes as rows and the C categories as columns.
   (b) Compute the Pearson χ^2 statistic with $(J_t - 1)(C - 1)$ degrees of freedom.
   (c) Use the Peizer–Pratt transformation (A.1) to convert it to a z value.

   Let z_c denote the largest among the $(K - K_1)$ z values.
3. Interaction test for each pair of numerical variables $(X_k, X_{k'})$:
   (a) Divide the $(X_k, X_{k'})$ space into four quadrants at the sample medians.
   (b) Construct a $J_t \times 4$ contingency table with classes as rows and the quadrants as columns.
   (c) Compute the Pearson χ^2 statistic with $3(J_t - 1)$ degrees of freedom.
   (d) Use the Peizer–Pratt transformation (A.1) to convert it to a z value.

   Let z_{nn} denote the largest among the $K_1(K_1 - 1)/2$ z values.
4. Interaction test for each pair of categorical variables: Use pairs of categorical values to form the groups in the table. If the pair of variables takes C_1 and C_2 categorical values, a $J_t \times C_1C_2$ table is obtained. Let z_{cc} denote the largest among the $(K - K_1)(K - K_1 - 1)/2$ z values.
5. Interaction tests for pairs $(X_k, X_{k'})$, where X_k is numerical and $X_{k'}$ is categorical: If $X_{k'}$ takes C values, obtain a $J_t \times 2C$ table. Let z_{nc} denote the largest among the $K_1(K - K_1)$ z values.

Let f^* be the bootstrap value from Algorithm 3 and define $z^* = \max\{f^* z_n, z_c, f^* z_{nn}, z_{cc}, z_{nc}\}$.

1. If $f^* z_n = z^*$, select the numerical variable with the largest z.
2. If $z_c = z^*$, select a categorical variable with the largest z.
3. If $f^* z_{nn} = z^*$, select the numerical variable in the pair with the larger z.
4. If $z_{cc} = z^*$, select the categorical variable in the pair with the larger z.
5. If $z_{nc} = z^*$, select the categorical variable in the interacting pair.

Algorithm 3 (Bootstrap Bias Correction)

1. Create a bootstrap learning sample by copying the values of the variables and bootstrapping the Y column so that the response variable is independent of the predictors.
2. Apply Steps 1–5 in Algorithm 2 to the bootstrap sample to get five sets of z values.
3. Given $f > 1$, select a numerical variable if $f \max\{z_n, z_{nn}\} \ge \max\{z_c, z_{cc}, z_{nc}\}$. Otherwise, select a categorical variable.
4. Repeat Steps 1–3 many times with several values of f. Let $\pi(f)$ be the proportion of times that a numerical variable is selected.
5. Linearly interpolate if necessary to find f^* such that $\pi(f^*)$ equals the proportion of numerical variables in the data.

Algorithm 4 (Box-Cox Transformation)

Suppose X is the selected variable. If X is categorical, its values are first transformed to crimcoord values.

1. Let $x_{(i)}$ denote the ith order statistic. Define $\theta = 0$ if $x_{(1)} > 0$ and $\theta = 2x_{(1)} - x_{(2)}$ otherwise.

2. Given λ, define

$$x^{(\lambda)} = \begin{cases} [(x-\theta)^{\lambda} - 1]/\lambda, & \text{if } \lambda \neq 0, \\ \log(x-\theta), & \text{if } \lambda = 0. \end{cases}$$

3. Let $\hat{\lambda}$ be the minimizer of

$$\sum_j \sum_i \left[x_{ji}^{(\lambda)} - \bar{x}_j^{(\lambda)}\right]^2 \exp\left\{-2n^{-1}\lambda\left[\sum_j \sum_i \log x_{ji}\right]\right\},$$

where x_{ji} is the ith value of X in class j and $\bar{x}_j^{(\lambda)}$ is the sample class mean of their transformed values.

4. Transform each x value to $x^{(\hat{\lambda})}$.

[Received September 1999. Revised July 2000.]

REFERENCES

Breiman, L. (1998), "Arcing Classifiers" (with discussion), *The Annals of Statistics*, 26, 801–849.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984), *Classification and Regression Trees*, New York: Chapman and Hall.

Brodley, C. E., and Utgoff, P. E. (1995), "Multivariate Decision Trees," *Machine Learning*, 19, 45–77.

Buntine, W. (1992), "Learning Classification Trees," *Statistics and Computing*, 2, 63–73.

Cestnik, G., Konenenko, I., and Bratko, I. (1987), "Assistant-86: A Knowledge-Elicitation Tool for Sophisticated Users," in *Progress in Machine Learning*, (eds). I. Bratko and N. Lavrac, Manhasset, NY: Sigma Press.

Clark, L. A., and Pregibon, D. (1993), "Tree-Based Models," in *Statistical Models in S*, eds. J. M. Chambers and T. J. Hastie, New York: Chapman and Hall, pp. 377–419.

Diaconis, P., and Efron, B. (1983), "Computer-Intensive Methods in Statistics," *Scientific American*, 248, 116–130.

Doyle, P. (1973), "The Use of Automatic Interaction Detector and Similar Search Procedures," *Operational Research Quarterly*, 24, 465–467.

Fayyad, U. M. (1991), "On the Induction of Decision Trees for Multiple Concept Learning," Ph.D. thesis, University of Michigan, Dept. of EECS.

Hastie, T., and Tibshirani, R. (1996), "Discriminant Analysis by Gaussian Mixtures," Journal of the Royal Statistical Society, Ser. B, 58, 155–176.

Hastie, T., Buja, A., and Tibshirani, R. (1995), "Penalized Discriminant Analysis," *The Annals of Statistics*, 23, 73–102.

Hastie, T., Tibshirani, R., and Buja, A. (1994), "Flexible Discriminant Analysis by Optimal Scoring," *Journal of the American Statistical Association,* 89, 1255–1270.

Hawkins, D. M. (1997), "FIRM: Formal Inference-based Recursive Modeling, PC Version, Release 2.1," Technical Report 546, University of Minnesota, School of Statistics.

Hochberg, Y., and Tamhane, A. C. (1987), *Multiple Comparison Procedures*, New York: Wiley.

Holte, R. C. (1993), "Very Simple Classification Rules Perform Well on Most Commonly Used Datasets," *Machine Learning*, 11, 63–90.

Kass, G. V. (1980), "An Exploratory Technique for Investigating Large Quantities of Categorical Data," *Applied Statistics*, 29, 119–127.

Kohonen, T. (1995), *Self-Organizing Maps*, Heidelberg: Springer-Verlag.

Kooperberg, C., Bose, S., and Stone, C. J. (1997), "Polychotomous Regression," *Journal of the American Statistical Association*, 92, 117–127.

Levene, H. (1960), "Robust Tests for Equality of Variances," in Contributions to Probability and Statistics, eds. I. Olkin, S. G. Ghurye, W. Hoeffding, W. G. Madow, and H. B. Mann, Palo Alto, CA: Stanford University Press, pp. 278–292.

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," *Machine Learning*, 40, 203–228.

Lock, R. H. (1993), "1993 New Car Data," *Journal of Statistics Education*, 1(1).

Loh, W.-Y. (2001), "Regression Trees With Unbiased Variable Selection and Interaction Detection," *Statistica Sinica*, to appear.

Loh, W.-Y., and Shih, Y.-S. (1997), "Split Selection Methods for Classification Trees," *Statistica Sinica*, 7, 815–840.

Loh, W.-Y., and Vanichsetakul, N. (1988), "Tree-Structured Classification via Generalized Discriminant Analysis" (with comments), *Journal of the American Statistical Association*, 83, 715–728.

Merz, C. J., and Murphy, P. M. (1996), "UCI Repository of Machine Learning Databases," University of California, Irvine, Dept. of Information and Computer Science, (`http://www.ics.uci.edu/mlearn/MLRepository.html`).

Müller, W., and Wysotzki, F. (1994), "Automatic Construction of Decision Trees for Classification," *Annals of Operations Research*, 52, 231–247.

Murthy, S. K., Kasif, S., and Salzberg, S. (1994), "A System for Induction of Oblique Decision Trees," *Journal of Artificial Intelligence Research*, 2, 1–33.

Qu, P., and Loh, W.-Y. (1992), "Application of Box–Cox Transformations to Discrimination for the Two-Class Problem," *Communications in Statistics (Theory and Methods)*, 21, 2757–2774.

Quinlan, J. R. (1989), "Unknown Attribute Values in Induction," in *Proceedings of the Sixth International Machine Learning Workshop*, pp. 164–168.

——— (1993), *C4.5: Programs for Machine Learning*, San Mateo, CA: Morgan Kaufmann.

Rouncefield, M. (1995), "The Statistics of Poverty and Inequality," *Journal of Statistics Education*, 3(3).

Sarle, W. S. (1994), "Neural Networks and Statistical Models," in *Proceedings of the Nineteenth Annual SAS Users Groups International Conference*, Cary, NC: SAS Institute, Inc., pp. 1538–1550.

Seber, G. A. F. (1984), *Multivariate Observations*, New York: Wiley.

Steinberg, D., and Colla, P. (1992), *CART: A Supplementary Module for SYSTAT* Evanston, IL: SYSTAT Inc.

Utgoff, P. E., Berkman, N. C., and Clouse, J. A. (1997), "Decision Tree Induction Based on Efficient Tree Restructuring," *Machine Learning*, 29, 5–44.

White, A. P., and Liu, W. Z. (1994), "Bias in Information-Based Measures in Decision Tree Induction," *Machine Learning*, 15, 321–329.

Zhang, H. (1998), Comment on "Bayesian CART model search," *Journal of the American Statistical Association*, 93, 948–950.