Classification Trees with Bivariate Linear Discriminant Node Models

Journal of Computational and Graphical Statistics, 2003, **12**, 512–530

Hyunjoong Kim
Department of Statistics
University of Tennessee
Knoxville, TN 37996
hjkim@utk.edu

Wei-Yin Loh [*]
Department of Statistics
University of Wisconsin
Madison, WI 53706
loh@stat.wisc.edu

Abstract

We introduce a classification tree algorithm that can simultaneously reduce tree size, improve class prediction, and enhance data visualization. We accomplish this by fitting a bivariate linear discriminant model to the data in each node. Standard algorithms can produce fairly large tree structures, because they employ a very simple node model, wherein the entire partition associated with a node is assigned to one class. We reduce the size of our trees by letting the discriminant models share part of the data complexity. Being themselves classifiers, the discriminant models can also help to improve prediction accuracy. Finally, because the discriminant models utilize only two predictor variables at a time, their effects are easily visualized by means of two-dimensional plots.

Our algorithm does not simply fit discriminant models to the terminal nodes of a pruned tree, as this does not reduce the size of the tree. Instead, discriminant modeling is carried out in all phases of tree growth and the misclassification costs of the node models are explicitly used to prune the tree. Our algorithm is also distinct from the "linear combination split" algorithms that partition the data space with arbitrarily oriented hyperplanes. We use

[*]Research partially supported by U.S. Army Research Office grant DAAD19-01-1-0586.

axis-orthogonal splits to preserve the interpretability of the tree structures. An extensive empirical study with real data sets shows that in general our algorithm has better prediction power than many other tree or non-tree algorithms.

Key words and phrases: Decision tree, linear discriminant analysis, tree-structured classifier

1 INTRODUCTION

A major advantage of classification trees is the direct and intuitive way they can be interpreted. Consider, for example, Figure 1 which shows a tree obtained using version 4 of the CART algorithm (Breiman, Friedman, Olshen and Stone, 1984; Steinberg and Colla, 1997). It is based on data from a study on breast cancer at the University of Wisconsin (Wolberg and Mangasarian, 1990). The data consist of measurements taken from 699 patients on 9 predictor variables taking integer values between 1 and 10. The response variable records whether a patient's tumor is `benign` or `malignant`. The tree is straightforward to interpret and shows that five predictor variables may be sufficient to predict the response.

Figures 2 and 3 show trees constructed from the same data using the CRUISE (Kim and Loh, 2001) and QUEST (Loh and Shih, 1997) algorithms. Both are smaller than the CART tree and are thus even easier to interpret. But are they equally good in terms of prediction accuracy? Empirical evidence (Lim, Loh and Shih, 2000; Kim and Loh, 2001) indicates that these algorithms tend to produce trees with comparable accuracy. If these three trees are indeed equally accurate, the QUEST tree may be preferred for its simplicity.

The class compositions are shown beside the terminal nodes of the trees. For example, the extreme left terminal node of the CART tree contains 416 `benign` and 5 `malignant` cases. If a user wishes to see how these 421 cases are distributed in the space of the predictor variables, what is the best way to do this graphically? One could make one-dimensional dot plots of the data for each variable, using a different plot symbol for each class. This will require 9 dot plots. Better still, we could look at two-dimensional plots. But which predictor variable to plot against which? A scatterplot matrix of all pairs of predictors will contain $9^2 = 81$ plots per terminal node. These plots can be tiresome to examine if the number of predictors or the number of terminal nodes is large. Besides, many of the plots will probably be uninteresting. Clearly, it is useful to have a method that can screen through all the plots and show us only the interesting ones. We consider a plot to be "interesting"

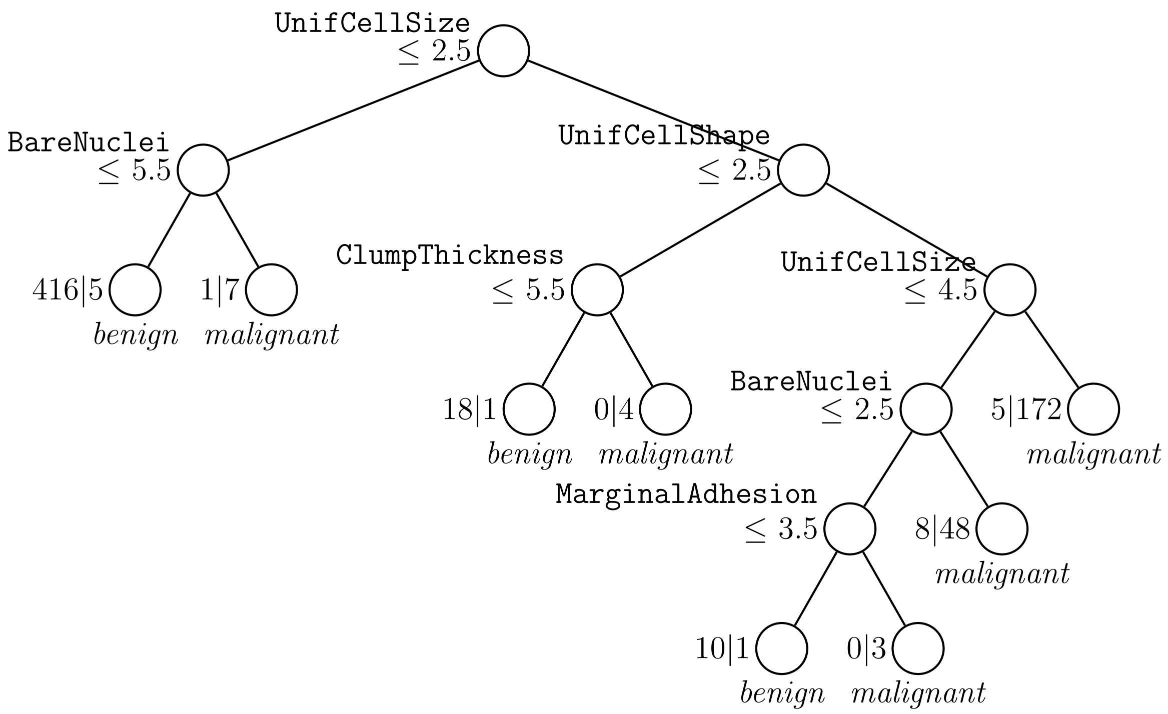


Figure 1: CART tree for breast cancer data. At an intermediate node, a case goes to the left subnode if it satisfies the condition there; otherwise it goes to the right subnode. The pair of numbers on the left of a terminal node gives the numbers of benign and malignant cases at the node.

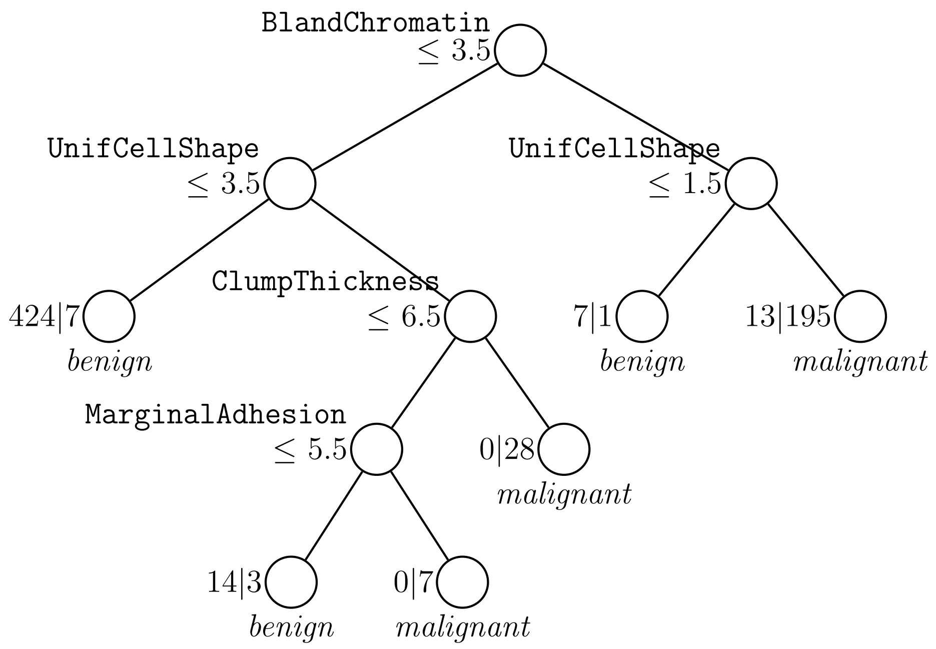


Figure 2: CRUISE tree for breast cancer data. At an intermediate node, a case goes to the left subnode if it satisfies the condition there; otherwise it goes to the right subnode. The pair of numbers on the left of a terminal node gives the numbers of benign and malignant cases at the node.

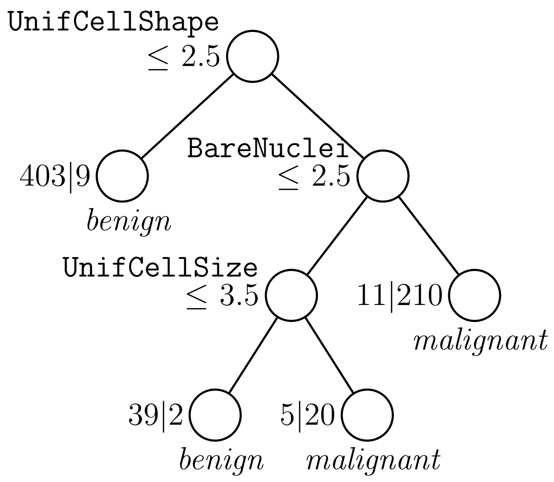


Figure 3: QUEST tree for breast cancer data. At an intermediate node, a case goes to the left subnode if it satisfies the condition there; otherwise it goes to the right subnode. The pair of numbers on the left of a terminal node gives the numbers of benign and malignant cases at the node.

if it shows good separation of the classes.

There is another benefit to graphing the data in each terminal node. The common goal in CART, CRUISE, and QUEST is to obtain a tree such that the learning sample in each terminal node is quite pure. When this cannot be achieved with a small number of univariate (i.e., axis-orthogonal) splits, we will get either a large tree or an extremely simple one (due to over-pruning). One solution is to employ linear combination splits but such splits are usually difficult to interpret if they involve more than two variables. We propose instead to retain univariate splits but fit a linear discriminant model to the best two-variable plot at each node. Because the discriminant models can be used for class prediction, it is not necessary for the terminal nodes to be very pure. Thus we can simplify the tree structure without sacrificing interpretability.

Figure 4 shows the result of applying this idea to the breast cancer data. The tree has only two splits. Plots of the jittered data in the three terminal nodes are given in Figure 5. They show that the two classes are separated quite well in the terminal nodes by the linear discriminant boundaries. Further, the northwest-southeast orientation of the boundaries explain why the CART and CRUISE trees have four or more levels of splits—often several axis-orthogonal splits are needed to approximate a non-orthogonal split.

We will describe the algorithm used to produce this tree in Section 2 and illustrate

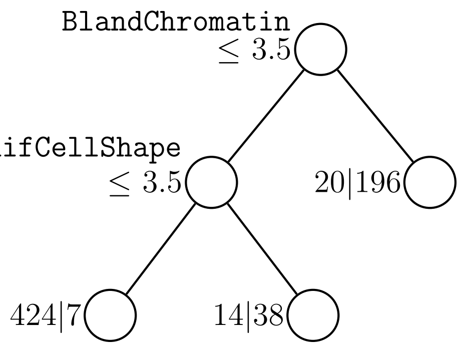


Figure 4: Classification tree for the breast cancer data using the proposed `M` method. At an intermediate node, a case goes to the left subnode if it satisfies the condition there; otherwise it goes to the right subnode. The pair of numbers on the left of a terminal node gives the numbers of benign and malignant cases.

it with an artificial data set. In Section 3, we present the results of an empirical comparative evaluation of the predictive accuracy and training time of our algorithm versus more than 30 other classification algorithms on 32 data sets. In Section 4, we use two real data sets to demonstrate the simplification potential of our approach. We conclude with some remarks in Section 5.

2 THE PROPOSED ALGORITHM

Although our approach is applicable to many split selection algorithms, we will describe its implementation on the CRUISE 2D algorithm. Our choice is influenced by its good prediction accuracy, its negligible bias in the variables selected to split the nodes, and its ability to detect local pairwise interactions between predictor variables (Kim and Loh, 2001).

We describe two ways to select the best pair of variables to fit a linear discriminant node model. The first method fits the model to all pairs of predictors and computes their resubstitution estimates of misclassification cost. The pair with the smallest cost is selected. If there are missing values, only the cases with complete observations in the respective pair of variables are used in the model fitting. The misclassification cost of the fitted model is estimated for all the cases in the node after missing values are imputed with the node class means (for numerical predictors) and modes (for categorical predictors). We call this the “`C`” (for cost) method.

In situations where there are missing values in the learning sample, it is con-

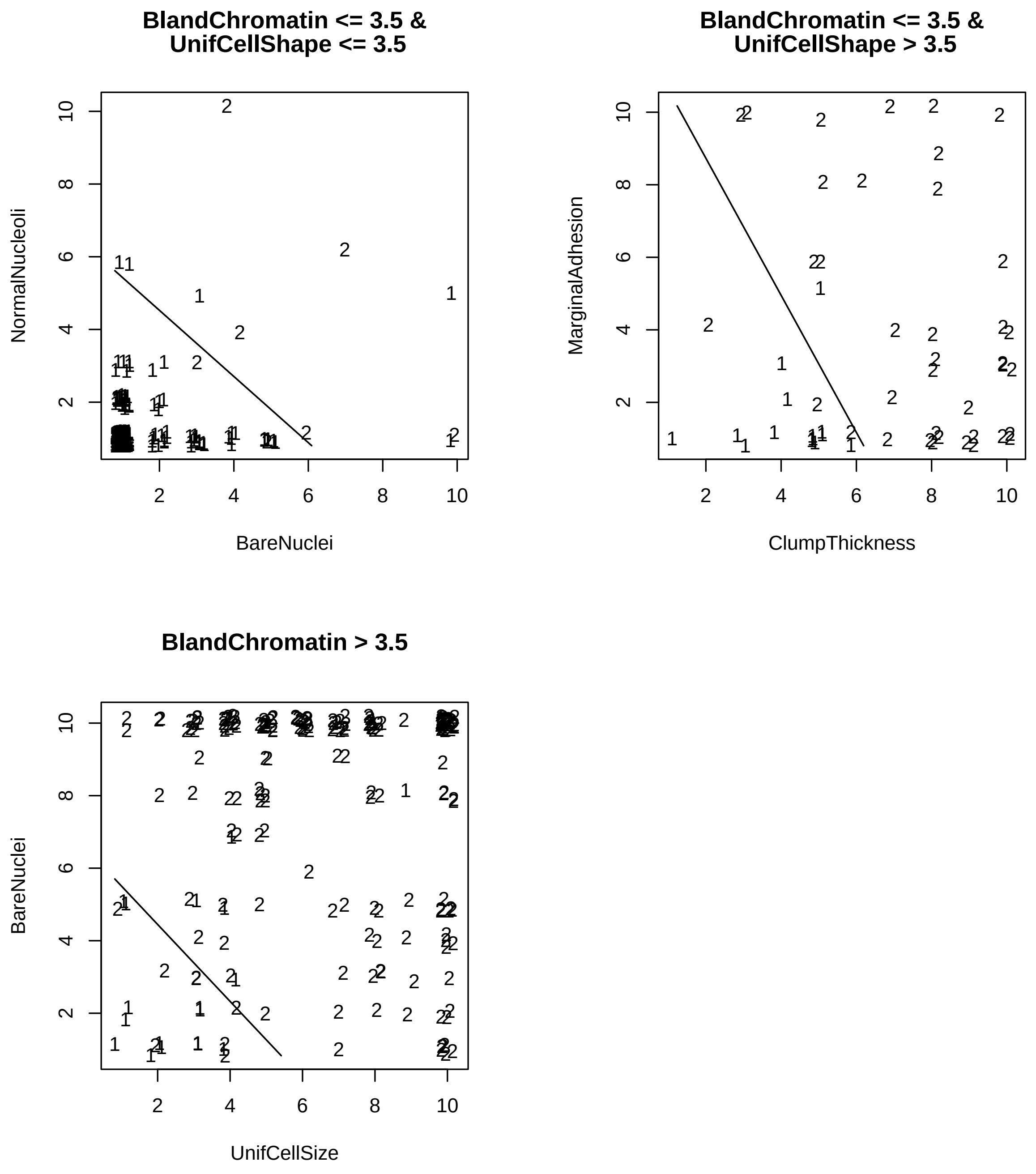


Figure 5: Jittered plots of data and the linear discriminant boundaries in the terminal nodes of the tree in Figure 4. The benign and malignant cases are labeled 1 and 2, respectively.

ceivable that the `C` method may select variables that have a lot of missing values. To avoid this problem, we propose a second method that is based on p-values from multivariate analysis of variance (MANOVA) calculations. Given a node t and a pair of predictor variables $\mathbf{x} = (x_l, x_m)'$, $l \neq m$, let n_j denote the number of class j cases in t with complete observations on $\mathbf{x}$. Let $\mathbf{x}_{ji}$ denote the ith observation from class j, $\bar{\mathbf{x}}_j$ be the sample mean vector for class j computed from the n_j cases, and $\bar{\mathbf{x}}$ be the overall sample mean vector ignoring class. Finally, let $\mathbf{B} = \sum_j n_j(\bar{\mathbf{x}}_j - \bar{\mathbf{x}})(\bar{\mathbf{x}}_j - \bar{\mathbf{x}})'$ and $\mathbf{W} = \sum_j \sum_{i=1}^{n_j}(\mathbf{x}_{ji} - \bar{\mathbf{x}}_j)(\mathbf{x}_{ji} - \bar{\mathbf{x}}_j)'$ denote the between and within group sum of squares matrices. The MANOVA Wilks' Λ statistic for testing equality of the class mean vectors is $\Lambda = |\mathbf{W}|/|\mathbf{B} + \mathbf{W}|$. We compute approximate p-values with Bartlett's (1938) approximation

$$-\{n - 1 - (2 + J_t)/2\} \log \Lambda \sim \chi^2_{2(J_t-1)}$$

where $n = \sum_j n_j$ and J_t is the number of classes present among the learning samples in t. The pair of variables with the minimum p-value is selected. We will refer to this as the "`M`" (for MANOVA) method.

To render the `C` and `M` methods applicable to categorical variables, we first transform each categorical value to a numerical value using a technique in Loh and Vanichsetakul (1988). Specifically, a categorical variable is converted into a 0-1 dummy vector and then projected onto the largest discriminant coordinate.

The rest of our tree construction algorithm proceeds as in CRUISE, except that during pruning, the resubstitution estimate of misclassification cost of the linear discriminant model is used in the cost-complexity function. The whole procedure may be formally stated as follows.

Algorithm 1

1. Modeling: *First, any categorical predictor variables are transformed to their largest discriminant coordinates at each node. Then a bivariate linear discriminant model is fitted to the data there. The pair of predictor variables is selected by one of two methods:*

   `C`: *For each pair of variables, a bivariate linear discriminant model is fitted to the data non-missing in those variables. An estimate of the misclassification cost of the model is obtained from all the cases in the node after imputation of missing values with node class means or modes. The pair of variables having the smallest estimated cost is selected. (If a pair of predictors has a singular within group matrix* $\mathbf{W}$*, a linear discriminant model is not fitted for the pair. Instead, the "constant" model is used,* *which classifies every observation to the class with the highest frequency in the node.)*

`M`: *Wilks' Λ is computed for each pair of predictor variables. Bartlett's approximation is used to obtain the p-values. The pair of variables with the smallest p-value is selected for use in the bivariate linear discriminant model.*

2. Partitioning: *Each node is split into two or more subnodes according to the CRUISE 2D univariate split selection rules described in Kim and Loh (2001).*

3. Stopping: *A node is not partitioned if one or more of the following conditions are met:*

   (a) There are very few cases in the node (default is 5).

   (b) At most one class has more than m cases, where m is user-specified (default is $m = \max\{2, N/200\}$, where N is the number of cases in the learning sample).

   (c) All the cases go down the same branch if the node is split.

4. Pruning: *The tree is pruned according to the cost-complexity pruning algorithm of Breiman et al. (1984). The cost at each node is the estimated misclassification cost of the model fitted in step 1. The estimate is based on the learning sample after missing values are imputed with the node class means (for ordered numerical variables) or modes (for categorical variables).*

This approach is different from the "lazy" one of taking a standard algorithm and adding a step that fits a discriminant model to each terminal node of the pruned tree. The latter approach does not decrease the size of the tree because node model fitting is carried out after the final tree is selected. Besides, it is unlikely to significantly increase classification accuracy since standard algorithms typically produce fairly pure terminal nodes which do not benefit from refinement by discriminant analysis. In our approach, a discriminant model is fitted to each node of the tree during its construction and the misclassification costs of the node models are used to prune the tree.

As a simple illustration of our method, we simulated a set of 10,000 data points uniformly distributed over the two-dimensional rectangle shown in panel (a) of Figure 6. If a point falls in the region above the crooked line, it has probability 0.95 to be in class 1 and probability 0.05 to be in class 2. Similarly, if a point falls in the region below the crooked line, it has probabilities 0.95 and 0.05 to be in class 2 and 1, respectively. The crooked line boundary is defined by the equation

$$y = \begin{cases} 3x/2, & 0 < x \leq 10/3 \\ 5, & 10/3 < x \leq 20/3 \\ 3x/2 - 5, & 20/2 < x \leq 10. \end{cases}$$

The partitions obtained with the CART and CRUISE 2D univariate split algorithms are shown in panels (b) and (c) of Figure 6. Both approximate poorly the two parts of the true class boundary that have positive slope. The corresponding results from the CART and CRUISE linear combination split algorithms are shown in panels (d) and (e). They miss the horizontal piece of the boundary altogether. The result from our `M` method is shown in panel (f).

3 ACCURACY AND SPEED ON REAL DATA

Lim et al. (2000) and Kim and Loh (2001) carried out a large empirical study to compare the prediction accuracy and training time of many classification algorithms. We now add the results for the `C` and `M` methods to theirs for comparison. Table 1 summarizes the 36 competing algorithms. Each algorithm is applied to 32 data sets. For each algorithm-data set pair, a ten-fold cross-validation estimate of the misclassification rate and the total training time are recorded. Details about the competing algorithms and data sets are given in Lim et al. (2000).

Figure 7 plots the median training time versus mean error rate of the algorithms. The training times are measured on a DEC 3000 Alpha Model 300 workstation running the UNIX operating system. In terms of mean error rate, the spline-based polytomous logistic regression `POL` is best. It has the lowest mean error rate of 0.195. Our `M` and `C` methods are ranked 9th and 11th, with mean error rates of 0.213 and 0.217, respectively. To determine if the differences are statistically significant, we fit a mixed effects model to the data, treating the algorithms as fixed effects and the datasets as random effects. A test of the hypothesis of no algorithm effects yields a p-value less than 0.001. Using 90% Tukey simultaneous confidence intervals (Hochberg and Tamhane, 1987, p. 81), we find that a difference in mean error rates less than 0.056 is not statistically significant from zero. Thus the `C` and `M` algorithms are not significantly different from `POL`. The solid vertical line in the top plot of Figure 7 separates the algorithms into two groups: those whose mean error rates do not differ significantly from that of `POL` and those that do. The three horizontal dotted lines in the plot divide the algorithms into four groups according to median training time:

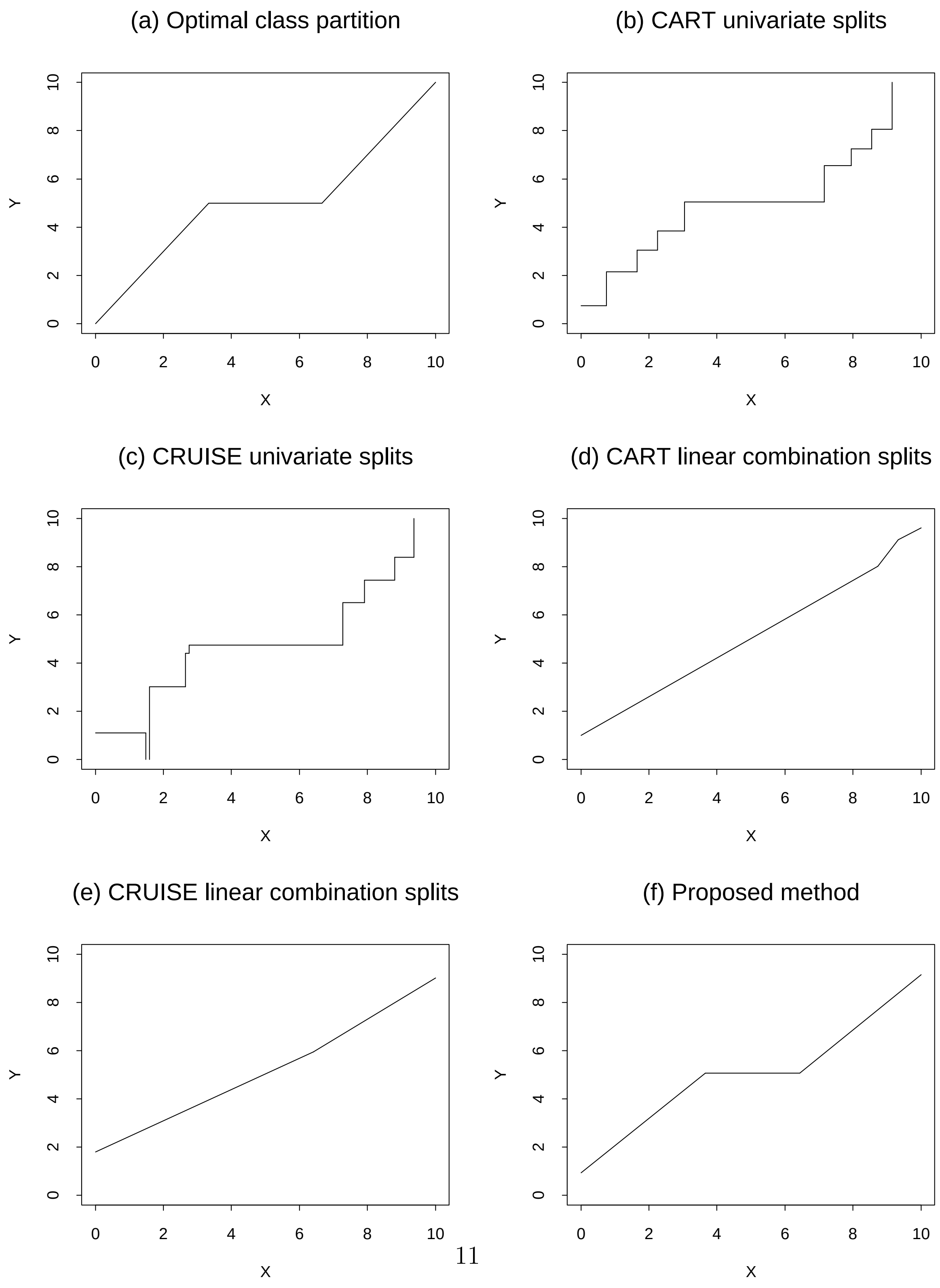


Figure 6: True and estimated class partitions for simulated data example.

Table 1: Classification algorithms in comparative study. The 0-SE tree is used where applicable.

Code	Name of algorithm
C	Proposed algorithm with CR2 split and C discriminant model selection
M	Proposed algorithm with CR2 split and M discriminant model selection
CR1	CRUISE 1D (Kim and Loh, 2001)
CR2	CRUISE 2D (Kim and Loh, 2001)
CRL	CRUISE linear combination splits (Kim and Loh, 2001)
CTU	Salford Systems CART univariate splits (Steinberg and Colla, 1997)
CTL	Salford Systems CART linear combination splits (Steinberg and Colla, 1997)
SPT	Splus-tree univariate splits (Clark and Pregibon, 1993)
QTU	QUEST univariate splits (Loh and Shih, 1997)
QTL	QUEST linear combination splits (Loh and Shih, 1997)
FTU	FACT univariate splits (Loh and Vanichsetakul, 1988)
FTL	FACT linear combination splits (Loh and Vanichsetakul, 1988)
IC	IND CART univariate splits (Buntine and Caruana, 1992)
IB	IND Bayes (Buntine and Caruana, 1992)
IBO	IND Bayes with opt style (Buntine and Caruana, 1992)
IM	IND Bayes with mml style (Buntine and Caruana, 1992)
IMO	IND Bayes with opt and mml styles (Buntine and Caruana, 1992)
C4T	C4.5 decision tree (Quinlan, 1993)
C4R	C4.5 decision rules (Quinlan, 1993)
OCU	OC1 tree, univariate splits (Murthy, Kasif and Salzberg, 1994)
OCL	OC1 with linear combination splits (Murthy et al., 1994)
OCM	OC1 with univariate and linear combination splits (Murthy et al., 1994)
LMT	LMDT linear combination split tree (Brodley and Utgoff, 1995)
CAL	CAL5 decision tree (Müller and Wysotzki, 1997)
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 (Hastie et al., 1994)
PDA	Penalized discriminant analysis (Hastie, Buja and Tibshirani, 1995)
MDA	Mixture discriminant analysis (Hastie and Tibshirani, 1996)
POL	POLYCLASS (Kooperberg, Bose and Stone, 1997)
LVQ	Learning vector quantization neural network (Kohonen, 1995)
RBF	Radial basis function neural network (Sarle, 1994)

less than one minute, one to ten minutes, ten minutes to one hour, and more than one hour. POL has the third highest median training time of 3.2 hours. The median training time of C is 93 seconds and that of M is 44 seconds. A magnified plot of the algorithms that are not statistically significant from POL and that require less than ten minutes of median training time is shown in the lower half of Figure 7. Among the classification tree algorithms that employ univariate splits in this group, the three algorithms with the lowest mean error rates are M, IC (IND CART), and C, in that order.

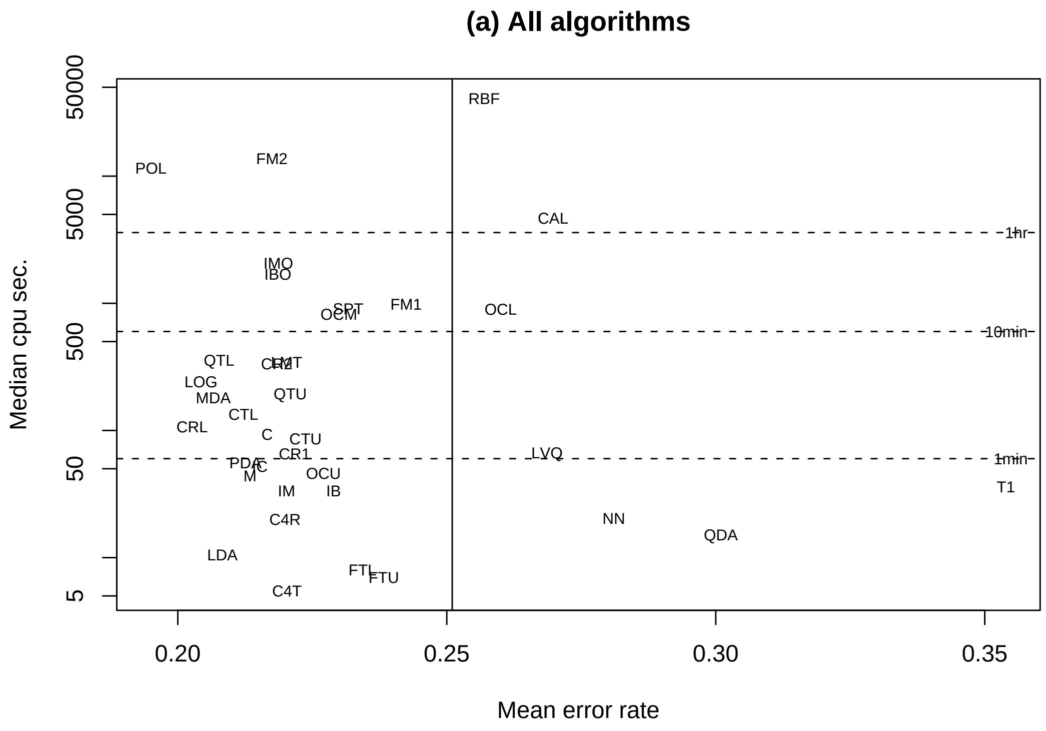
4 MORE EXAMPLES WITH REAL DATA

We now give two examples to show how our method can significantly simplify a tree structure. The first example shows maximum simplification, where only the root node is left after pruning. The second example shows how collinearity among predictors can create difficulties for other classification tree methods.

4.1 Car Data

This data set is from Lock (1993). It contains specifications for 93 new car models for the 1993 year. We use the type of car (small, sporty, compact, midsize, large, and van) as the class variable. The predictor variables are listed in Table 2: nineteen are numerical, three are categorical, and two are binary. The only missing values are for cylin in the rotary engine Mazda RX-7, rearseat for the two-seaters (Corvette and RX-7), and luggage for all the vans and the two-seaters.

The CART, CRUISE and QUEST trees are shown in Figures 8–10. They have 5, 14, and 12 terminal nodes and misclassify 33, 12, and 5 cases, respectively. The C and M methods both yield trivial trees after pruning. The M method selects a linear discriminant model based on weight and passngr; its linear discriminant boundaries are shown in Figure 11 and it misclassifies 18 cases. The C method chooses the variables whlbase and passngr for its linear discriminant model and misclassifies 15 cases. Figure 11 is very easy to interpret. As the partitions indicate, small cars are the lightest and have seating for 4 or 5 passengers. Sporty cars have medium weight but their passenger capacity is 4 or less. Compact cars also have medium weight, but they can seat 5 or 6 passengers. Medium cars are heavy and their passenger capacity ranges from 4–6. The large cars are also heavy but they all seat 6. Finally, vans are heavy but seat 7 or 8. Clearly, it is not possible to draw such simple conclusions from the CRUISE and QUEST trees. (Although the CART tree is easy to interpret,



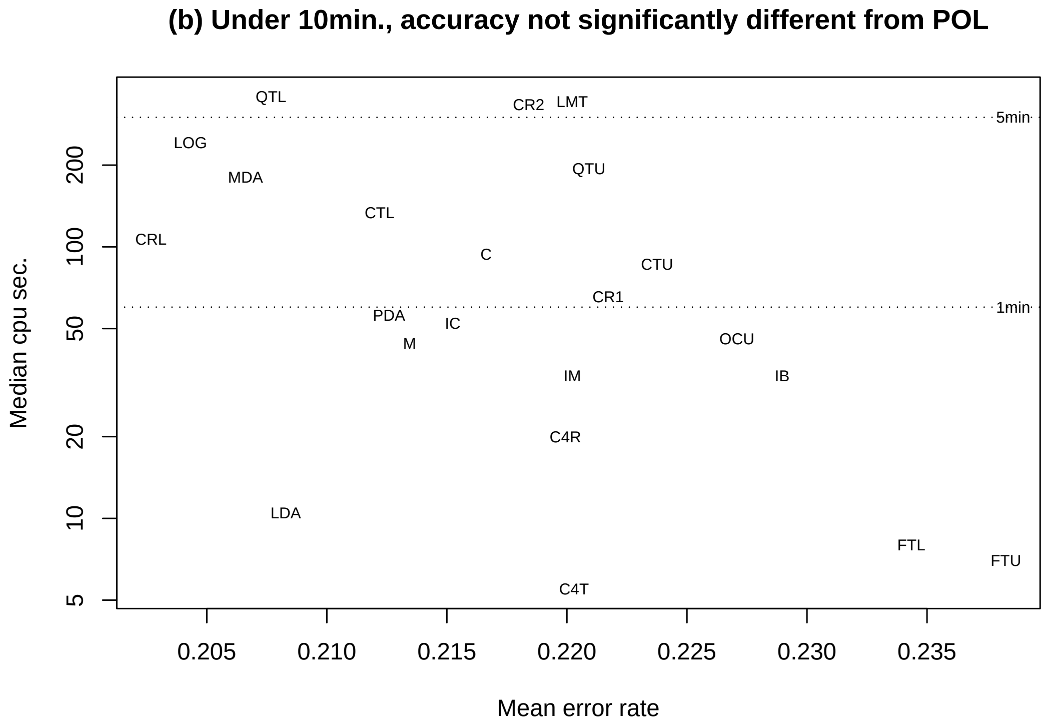


Figure 7: Median training time versus mean error rate. Plot symbols are defined in Table 1. Vertical axes are in log-scale. Algorithms to the left of the solid vertical line in plot (a) have mean error rates that are not statistically significant at the 10% simultaneous level from `POL`. The subset of these that have median training time less than ten minutes is shown in plot (b).

Table 2: Variables for car data

Variable	Definition	Variable	Definition
`manuf`	manufacturer (31 categories)	`rev`	engine revolutions per mile
`minprice`	minimum price in \$1000's	`manual`	manual transmission (yes, no)
`midprice`	midrange price in \$1000's	`fuel`	fuel tank capacity in gallons
`maxprice`	maximum price in \$1000's	`passngr`	passenger capacity
`citympg`	city miles per gallon	`length`	length in inches
`hwympg`	highway miles per gallon	`wheelbase`	wheelbase length in inches
`airbag`	air bags standard (3 categories)	`width`	width in inches
`drtrain`	drive train type (3 categories)	`uturn`	U-turn space in feet
`cylin`	number of cylinders	`rearseat`	rear seat room in inches
`enginsz`	engine size in liters	`luggage`	luggage capacity in cu. ft.
`hp`	horsepower	`weight`	weight in lbs.
`rpm`	revolutions per minute	`domestic`	U.S. or non-U.S. manufacturer

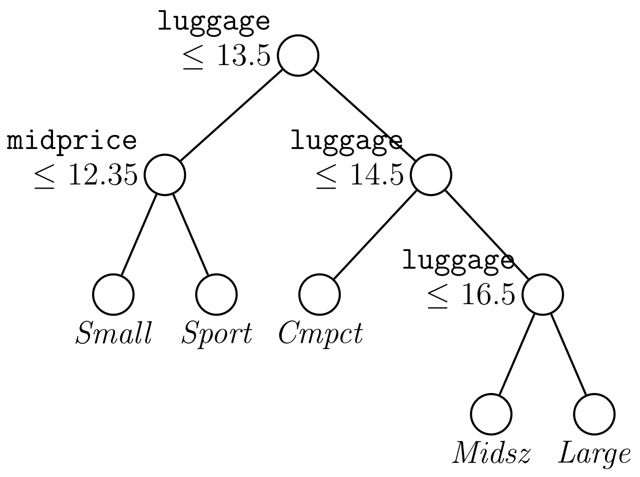


Figure 8: CART tree for car data

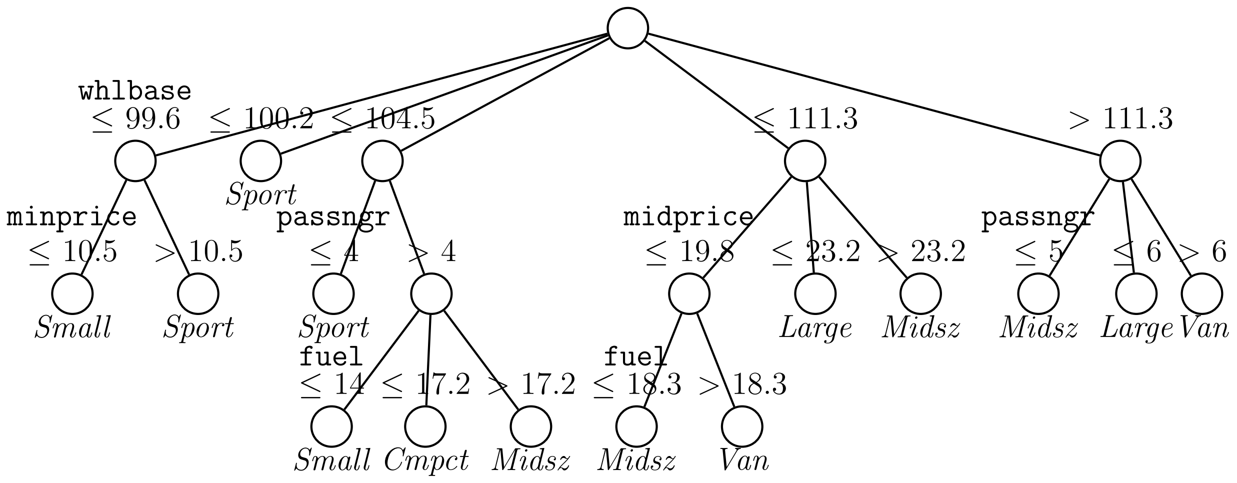


Figure 9: CRUISE tree for car data

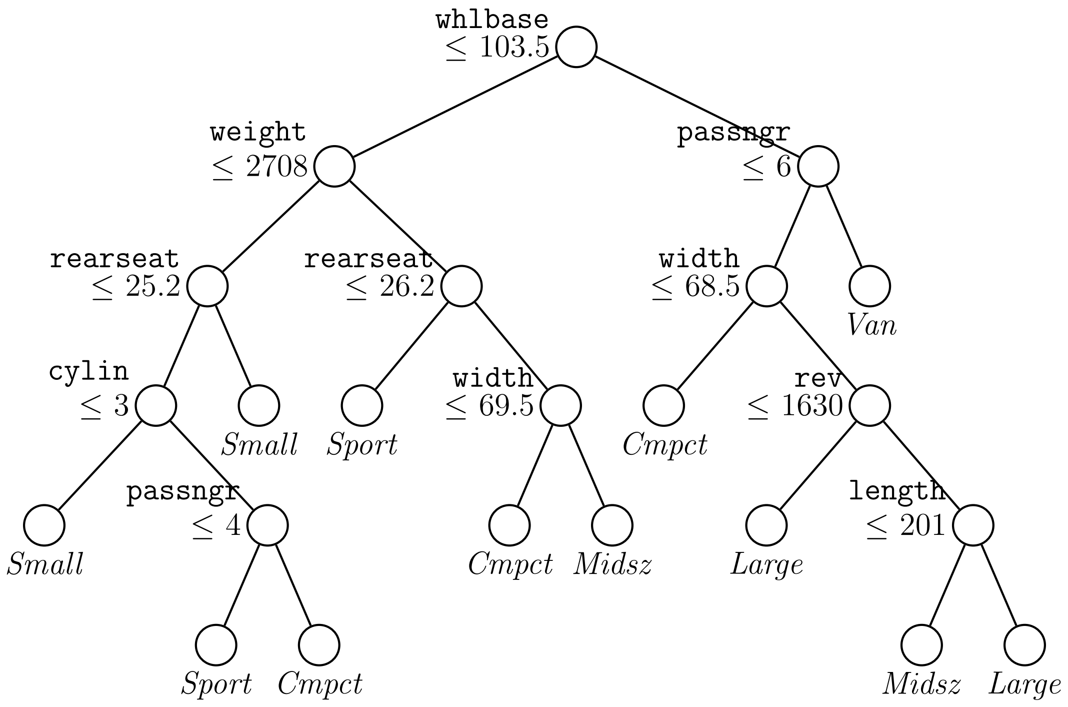


Figure 10: QUEST tree for car data

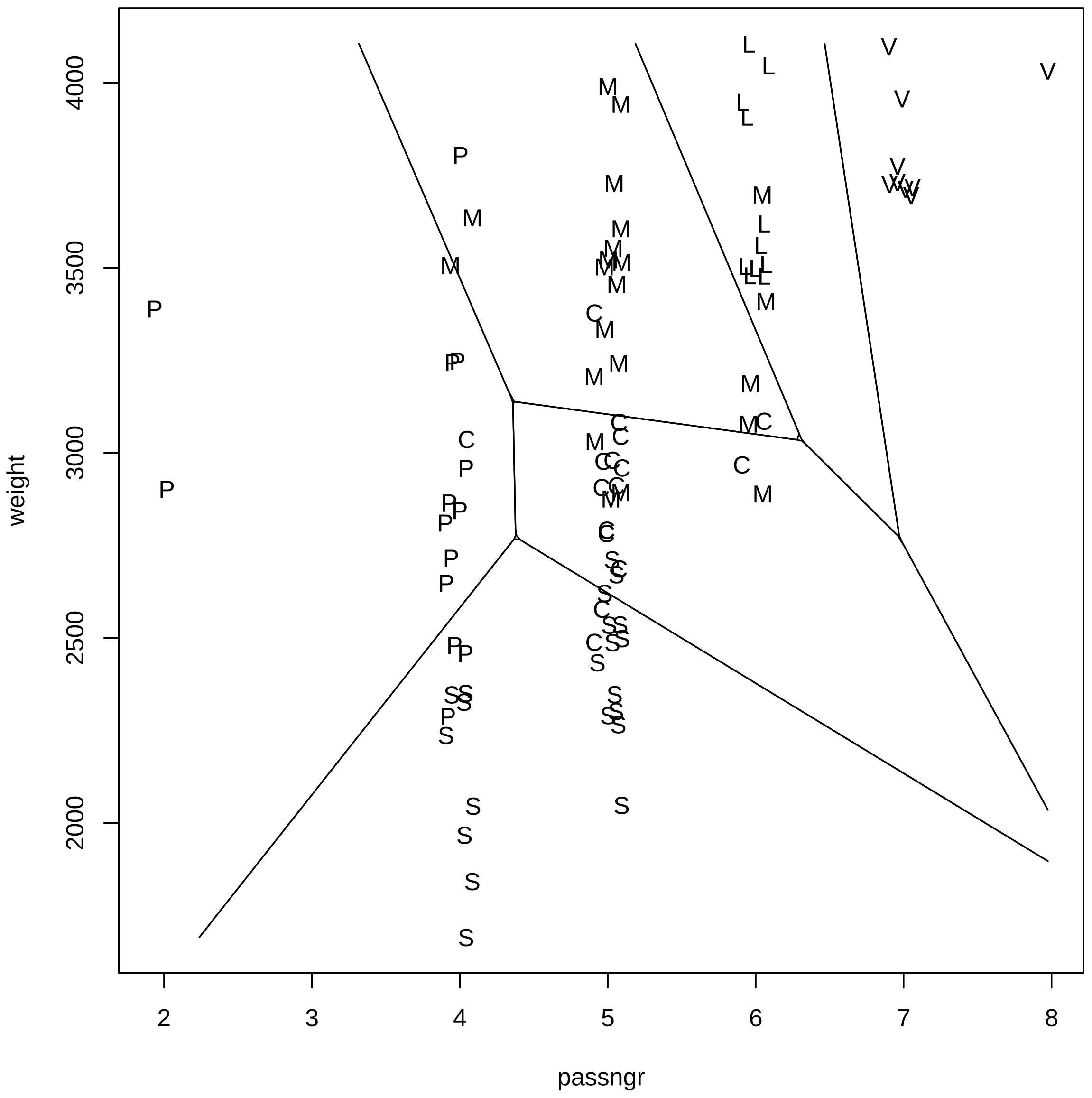


Figure 11: Jittered plot of car data and linear discriminant partitions for the `M` method; S = small, P = sporty, C = compact, M = midsize, L = large, V = van.

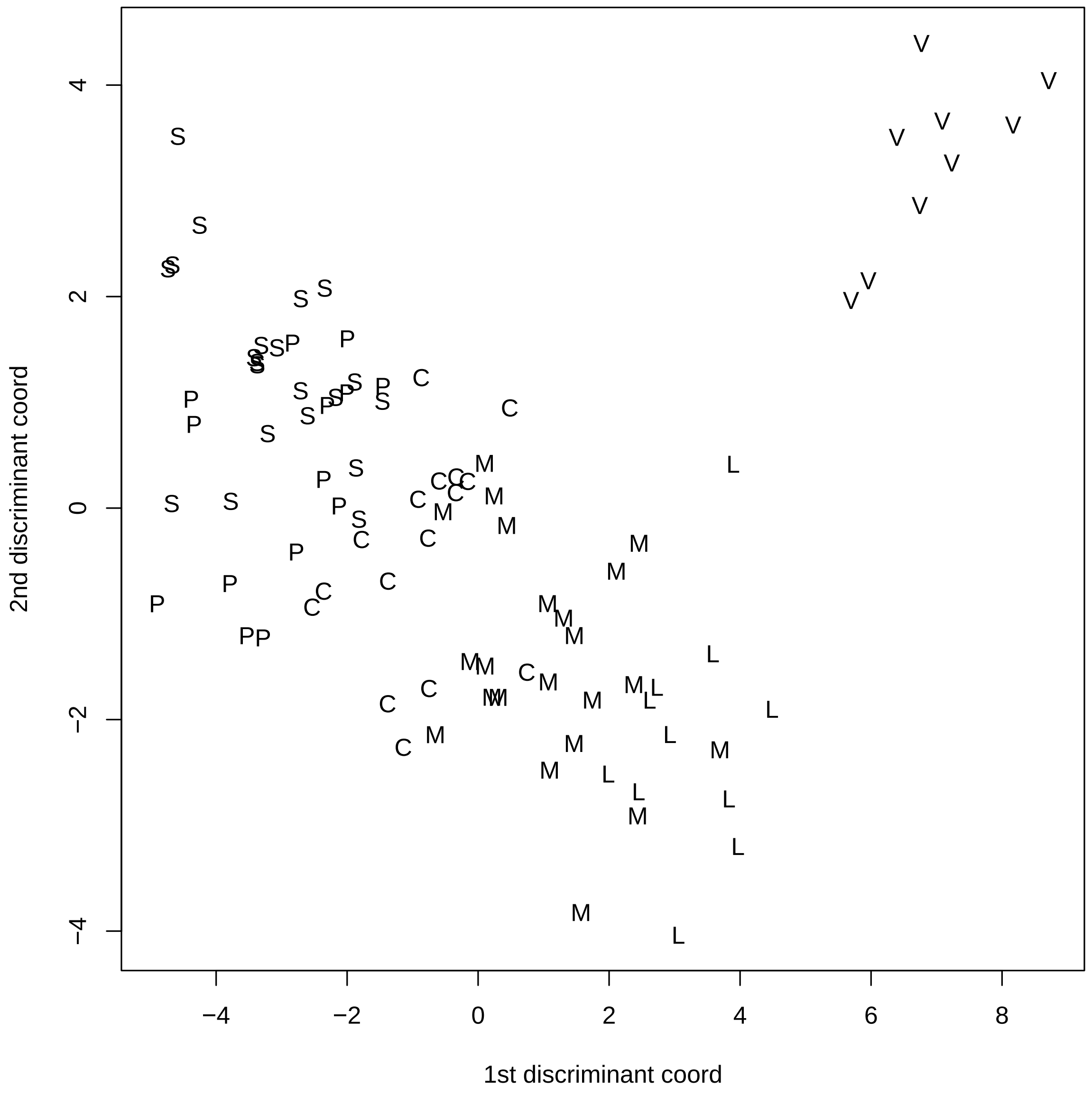


Figure 12: Projection of the car data onto the space of the first two discriminant coordinates of a 20-variable model; S = small, P = sporty, C = compact, M = midsize, L = large, V = van.

Table 3: Predictor variables for fish identification problem

`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

it does not predict `vans` because of missing values in `luggage`—see Kim and Loh (2001) for further discussion of this problem.)

Since the `C` and `M` methods yield single bivariate linear discriminant models here, it is natural to compare them with a multivariate linear discriminant model fitted to all the predictor variables. This task is complicated by the presence of a 31-valued categorical variable (`manuf`) and three variables (`cylin`, `rearseat`, `luggage`) that have missing values. We choose to exclude these four variables in order to keep the sample size constant. After treating the binary predictors as 0-1 variables, this leaves 20 numerical predictors. The resulting 20-variable linear discriminant model misclassifies 9 cases. It thus appears to be more accurate than the `C` and `M` methods but less accurate than `QUEST` (note: the apparent error rate is usually biased low). A weakness of the 20-variate discriminant model is, however, that it cannot be visualized. The best that can be done is to plot the result in the space of the first two discriminant coordinates. Such a projection is given in Figure 12; it obviously does not explain why the model has such a low error rate.

4.2 Fish Data

This data set is from the UC Irvine Repository of databases (Murphy and Aha, 1994). The data consist of observations on 159 fishes caught in a lake in Finland. Seven species of fish are represented in the sample: (1) Bream, (2) Whitefish, (3) Roach, (4) Parkki, (5) Smelt, (6) Pike, and (7) Perch. The predictor variables are defined in Table 3. The `sex` variable is missing from 87 cases and another case does not have a value for the `weight` variable.

The CART, CRUISE, and QUEST trees for predicting species are shown in Figures 13 and 14. They misclassify 21, 24, and 26 cases, respectively. The CART tree

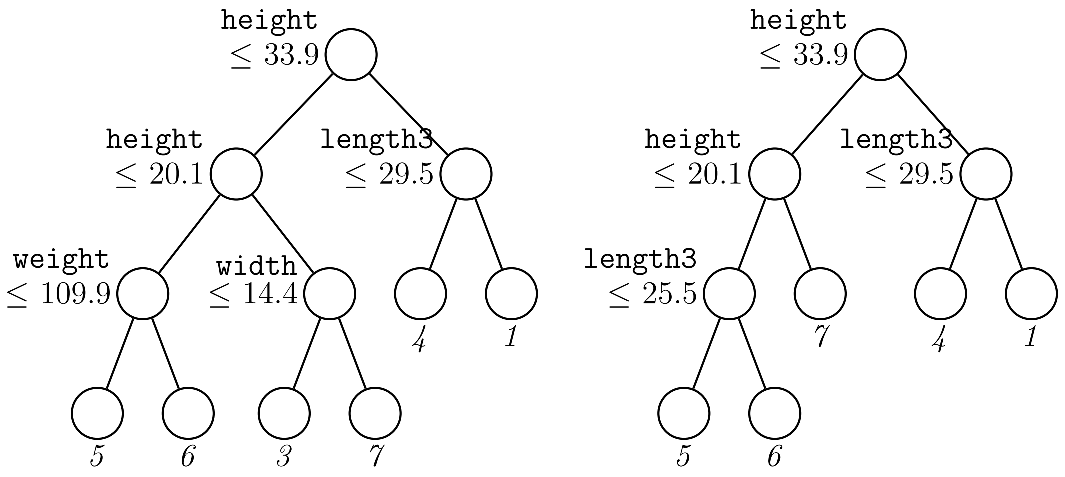


Figure 13: CART (left) and QUEST (right) trees for fish data. The number in italics beneath each terminal node is the predicted class.

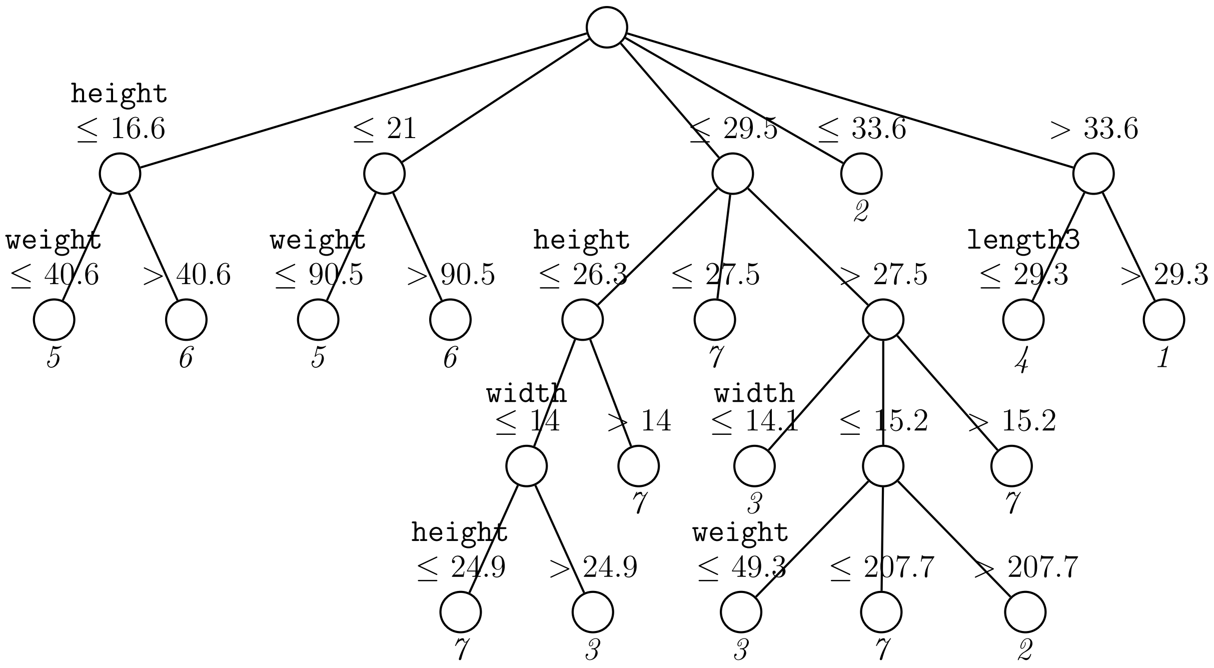


Figure 14: CRUISE 2D tree for fish data

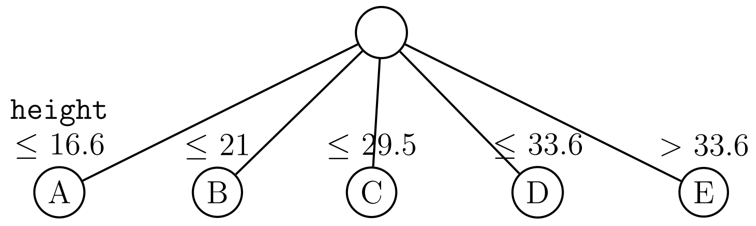


Figure 15: `M` tree for fish data

does not predict class 2 and the QUEST tree does not predict classes 2 and 3.

Figure 15 shows the tree from the `M` method. It splits just once, on `height`, and misclassifies only 3 cases. (The `C` tree has the same structure as the `M` tree, although the variables employed in the linear discriminant node models are slightly different. It also misclassifies 3 cases.) Table 4 displays the class compositions and the predictions in each terminal node of the tree.

The reason for the low error rate of the `M` tree is apparent from Figure 16, which plots the data and the linear discriminant boundaries in each terminal node of the tree. There is a high degree of collinearity among the predictor variables in three of the five terminal nodes. In particular, the collinearity between `length2` and `length3` in node C makes it difficult for any classification tree that employs only univariate splits to achieve a low error rate.

A referee noticed that each terminal node of the `M` tree contains only two or three classes. This is due to the CRUISE split selection algorithm, which seeks to divide them. Since a linear discriminant model is fitted only to the classes present in a node, the model can be relatively simple. As a result, even when the number of classes is large, our approach of fitting a discriminant model at each node does not necessarily add a lot of complexity.

If linear combination splits are allowed, much lower error rates are possible. The CART, CRUISE, and QUEST trees using such splits misclassify 17, 1, and 1 cases, respectively. But since these splits involve more than two variables, they are practically impossible to interpret or visualize. Our method thus strikes a useful compromise between prediction accuracy and interpretability. The tree sizes and resubstitution estimates of error rates of the trees are summarized in Table 5.

Table 4: Predictions in the nodes of Figure 15

	Node A								Node B								Node C						
True	Predicted class							True	Predicted class							True	Predicted class						
Class	1	2	3	4	5	6	7	Class	1	2	3	4	5	6	7	Class	1	2	3	4	5	6	7
1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	2	0	1	3	0	0	0	0
3	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	3	0	0	19	0	0	0	0
4	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0
5	0	0	0	0	5	0	0	5	0	0	0	0	9	0	0	5	0	0	0	0	0	0	0
6	0	0	0	0	0	14	0	6	0	0	0	0	0	3	0	6	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	7	0	0	0	0	0	0	54

	Node D								Node E						
True	Predicted class							True	Predicted class						
Class	1	2	3	4	5	6	7	Class	1	2	3	4	5	6	7
1	0	0	0	0	0	0	0	1	35	0	0	0	0	0	0
2	0	2	0	0	0	0	0	2	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	3	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	4	0	0	0	11	0	0	0
5	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0
7	0	0	0	0	0	0	2	7	0	0	0	0	0	0	0

Table 5: Comparison of methods on fish data

	Univariate splits		Linear splits	
	#Terminal	Resub.	#Terminal	Resub.
Method	nodes	error	nodes	error
CART	6	21/159	6	17/159
CRUISE	16	13/159	16	1/159
QUEST	5	16/159	10	1/159
`C & M`	5	3/159	NA	NA

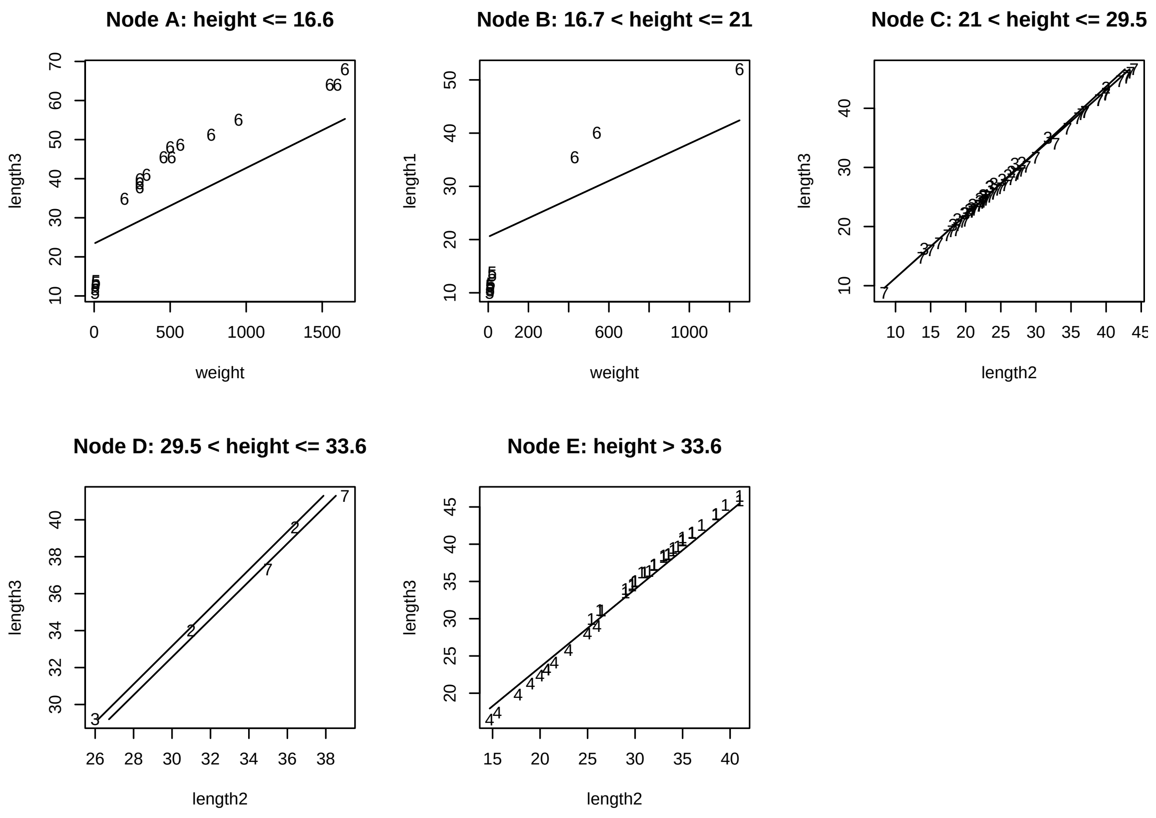


Figure 16: Bivariate linear discriminant partitions of data in the nodes of tree in Figure 15

5 CONCLUDING REMARKS

Classification trees are more intuitive to interpret than other classifiers. Our examples illustrate, however, that the trees are not necessarily easy to interpret. Ease of interpretation decreases rapidly with tree size. On the other hand, tree size typically grows with the amount of information in a data set. Thus the practical reality is that traditional classification trees are easy to interpret only if the data are not too complex. We can simplify a tree structure by pruning it, but over-pruning can degrade its prediction accuracy. (The CART, CRUISE, and QUEST tree shown here are pruned with the "1-SE rule" of Breiman et al. (1984). Thus they probably should not be pruned any further.)

The `C` and `M` methods are our answer to the question: how to reduce tree size without sacrificing prediction accuracy? We do this by taking full advantage of the visual power of two-dimensional graphical displays and the predictive power of linear discriminant analysis. Our methods employ a two-pronged approach to solve the problem. First, they reduce the tree size by absorbing some of the data complexity in the node models. Second, they force each node model to involve only two predictor variables so that the fitted model can be visualized in a two-dimensional plot. The empirical results reported here demonstrate that this approach can be quite effective.

We use linear discriminant analysis to fit the node models because it produces accurate classifiers in our empirical study (Section 3) and because it is computationally efficient. More sophisticated models, such as quadratic discriminant analysis or spline-based models, can also be used but it is unclear if they provide sufficient increase in prediction accuracy to justify the greater computational cost.

Acknowledgments

The authors are grateful to an associate editor and two referees for their helpful comments and suggestions. CART is a registered trademark of California Statistical Software, Inc.

References

Bartlett, M. S. (1938). Further aspects of the theory of multiple regression, *Proceedings of the Cambridge Philosophical Society*, Vol. 34, pp. 33–40.

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

Brodley, C. E. and Utgoff, P. E. (1995). Multivariate decision trees, *Machine Learning* **19**: 45–77.

Buntine, W. and Caruana, R. (1992). *Introduction to IND Version 2.1 and Recursive Partitioning*, NASA Ames Research Center, Moffet Field.

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

Hastie, T. and Tibshirani, R. (1996). Discriminant analysis by Gaussian mixtures, *Journal of the Royal Statistical Society,* **B 58**: 155–176.

Hastie, T., Buja, A. and Tibshirani, R. (1995). Penalized discriminant analysis, *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.

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

Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets, *Machine Learning* **11**: 63–90.

Kim, H. and Loh, W.-Y. (2001). Classification trees with unbiased multiway splits, *Journal of the American Statistical Association* **96**: 598–604.

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

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

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.*

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 discussion), *Journal of the American Statistical Association* **83**: 715–728.

Müller, W. and Wysotzki, F. (1997). The decision tree algorithm CAL5 based on a statistical approach to its splitting algorithm, *in* G. Nakhaeizadeh and C. Taylor (eds), *Machine Learning and Statistics: The Interface*, Wiley, New York, pp. 45–65.

Murphy, P. M. and Aha, D. W. (1994). UCI Repository of machine learning databases, University of California, Irvine, Department of Information and Computer Science. `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

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.

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

Sarle, W. S. (1994). Neural networks and statistical models, *Proceedings of the Nineteenth Annual SAS Users Groups International Conference*, SAS Institute, Inc., Cary, NC, pp. 1538–1550.

Steinberg, D. and Colla, P. (1997). *CART—Classification and Regression Trees: A Supplementary Manual for Windows*, Salford Systems Inc., San Diego.

Wolberg, W. H. and Mangasarian, O. L. (1990). Multisurface method of pattern separation for medical diagnosis applied to breast cytology, *Proceedings of the National Academy of Sciences* **87**: 9193–9196.