Article type: Focus Article

Tree-Structured Classifiers

Wei-Yin Loh

University of Wisconsin, Madison, USA

Keywords

Classification tree, discriminant analysis, missing value, prediction, selection bias

Abstract

A tree-structured classifier is a decision tree for predicting a class variable from one or more predictor variables. THAID [15, 7] was the first such algorithm. This article focuses on the CART® [2], C4.5 [17], and GUIDE [12] methods. The algorithms are briefly reviewed and their similarities and differences compared on a real data set and by simulation.

In a typical classification problem, we have a training sample $\mathcal{L} = \{(\mathbf{X}_1, Y_1), (\mathbf{X}_2, Y_2), \ldots, (\mathbf{X}_N, Y_N)\}$ of N observations, where each $\mathbf{X} = (X_1, \ldots, X_K)$ is a K-dimensional vector of predictor variables and Y is a class variable that takes one of J values. We want to construct a rule for predicting the Y value of a new observation given its value of $\mathbf{X}$. If the predictor variables are all ordered, i.e., non-categorical, some popular classifiers are linear discriminant analysis (LDA), nearest neighbor, and support vector machines. (Categorical predictor variables can be accommodated by transformation to vectors of 0-1 dummy variables.) Although these classifiers often possess good prediction accuracy, they act like black boxes and do not provide much insight into the roles of the predictor variables.

A tree-structured classifier (or classification tree) is an attractive alternative because it is easy to interpret. It is a decision tree obtained by recursive partitioning of the $\mathbf{X}$-space. An observation in a partition is predicted to belong to the class with minimum estimated misclassification cost. Classification trees have been demonstrated to possess high prediction accuracy compared to many other methods; see, e.g., Lim et al. [11], Perlich et al. [16], and Loh [12]. They do not require categorical predictor variables to be transformed. THAID [15, 7] is the first published algorithm. We review here the CART® [2], C4.5 [17], and GUIDE [12] algorithms and illustrate their similarities and differences on a real data set and by simulation.

CART

All classification tree algorithms have to address two common problems: how to split a node t and when to stop splitting it? The first problem is usually solved by means of a node impurity function $i(t)$, with the best split being the one that minimizes a function of the impurities in the subnodes. Let N_j be the number of training samples belonging to class j and let π_j denote the prior probability of class j, which may be known or estimated from the training sample, in which case $\pi_j = N_j/N$ ($j = 1, 2, \ldots, J$). Let $N_j(t)$ be the number of training samples in node t belonging to class j and define $p(j,t) = \pi_j N_j(t)/N_j$, $p(t) = \sum_j p(j,t)$, and $p(j|t) = p(j,t)/p(t)$. If the misclassification cost is equal for all classes, CART uses the Gini impurity function $i(t) = 1 - \sum_j p^2(j|t)$ and splits t into two subnodes t_L and t_R. The best split is the one that minimizes $p(t_L)i(t_L) + p(t_R)i(t_R)$. If X is an ordered variable, the split has the form $X \leq c$, with c being the mid-point of two consecutive order statistics of X. Otherwise, if X is a categorical variable, the split is $X \in S$ with S being a subset of the values of X. Searching for the latter type of split can be computationally expensive if X takes a large number of distinct values. If X has a distinct values, there are $(2^{a-1} - 1)$ splits of the form $X \in S$. When there are missing values, only the observations non-missing in X are used to evaluate the impurity function. CART uses surrogate splits to send observations with missing values through a split. A surrogate split is one on another variable that is most similar to the selected split in its partitioning of the data. Unlike THAID, which uses stopping rules to control splitting, CART first grows a large tree and then uses cross-validation to prune it to a smaller size.

To illustrate, consider a data set on cylinder banding in rotogravure printing from the UCI repository [1]. The data, described in Evans and Fisher [6], consist of 540 observations on 33 predictor variables, of which 14 are categorical. The class variable takes two values: "band" and "noband," with 228 and 312 observations, respectively. Table 1 lists the variable names and the values they take, with numbers of missing values in parentheses. The left side of Figure 1 shows the pruned tree obtained from RPART [20], an R implementation of CART. The first split is on `press` with observations taking values 815, 816, and 821 going to the left node and other values to the right, which is terminal and predicted to be "noband." The left node is split on `speed`, with values less than 2035 predicted to be "band," and otherwise "noband." The sample size in each terminal node is given on its left and the estimated error rate (proportion of training samples misclassified) beneath it. Overall, 163 training samples are misclassified.

C4.5

C4.5 uses the entropy function $i(t) = -\sum_j p(j|t) \log_2 p(j|t)$. Let s denote the split of t on X into subnodes $t_1, t_2, \ldots, t_m$ and let $f(t)$ be the proportion of observations with no missing X-values in t. Let t_0 denote the set of observations missing X; t_0 is empty if $f(t) = 1$. Let $q(t_k) = N(t_k)/N(t)$, for $k = 0, 1, \ldots, m$, and define the weighted sum of entropies $i_s(t) = \sum_{k=0}^{m} q(t_k) i(t_k)$. Then $g_s(t) = f(t)\{i(t) - i_s(t)\}$

Table 1 Variables for banding data, with number of missing values in parentheses.

Categorical	Description	Ordered	Description
grain	grain screened: yes, no (49)	proofcut	proof cut: 0–100 (55)
proof	proof on ctd ink: yes, no (57)	viscosity	viscosity: 0–100 (5)
blademfg	blade manufacturer: Benton, Daetwyler, Uddeholm (60)	caliper	caliper: 0–1.0 (27)
		hardener	hardener: 0–3.0 (7)
paper	paper type: uncoated, coated, super (0)	temp	ink temperature: 5-30 (2)
		inkpct	ink percent: 0–100 (56)
inktype	ink type: uncoated, coated, cover (0)	humid	humidity: 5–120 (1)
		current	current density: 20–50 (7)
direct	direct steam: yes, no (25)	rough	roughness: 0–2 (30)
solvent	solvent type: xylol, lactol, naptha, line, other (55)	pressure	blade pressure: 10-75 (63)
		varnish	varnish percent: 0–100 (56)
type	type on cylinder: yes, no (18)		
presstype	press type: albert, motter70, motter94, woodhoe (0)	speed	press speed: 0–4000 (10)
		anode	anode space ratio: 70–130 (7)
press	press: 821, 802, 813, 824, 815, 816, 827, 828 (0)	chrome	chrome content: 80–120 (3)
unitnum	unit number: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 (0)	roller	roller durometer: 15–120 (55)
size	cylinder size: catalog, spiegel, tabloid (3)	solvpct	solvent percent: 0–100 (56)
location	mill location: north US, south US, Canadian, Scandanavian, mid European (156)	voltage	voltage: 0-16 (57)
		amp	amperage: 0–10 (55)
tank	plating tank: 1910, 1911, other (18)	wax	wax: 0–4.0 (6)

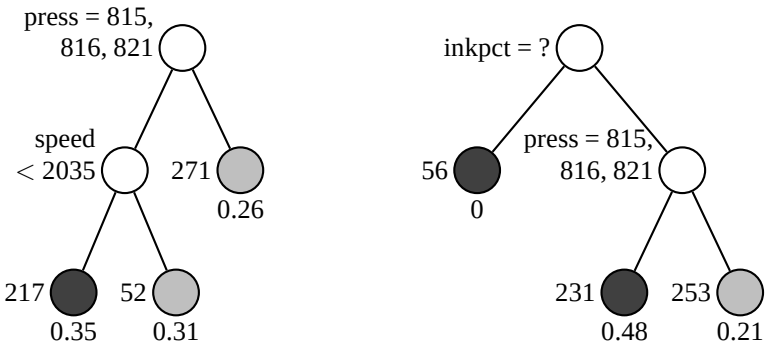

Figure 1: RPART (left) and GUIDE (right) trees with nodes predicted as “band” and “noband” colored dark and light gray, respectively. At each intermediate node, an observation goes to the left branch if and only if the posted condition is satisfied. The sample size and error rate are printed on the left and bottom of each terminal node.

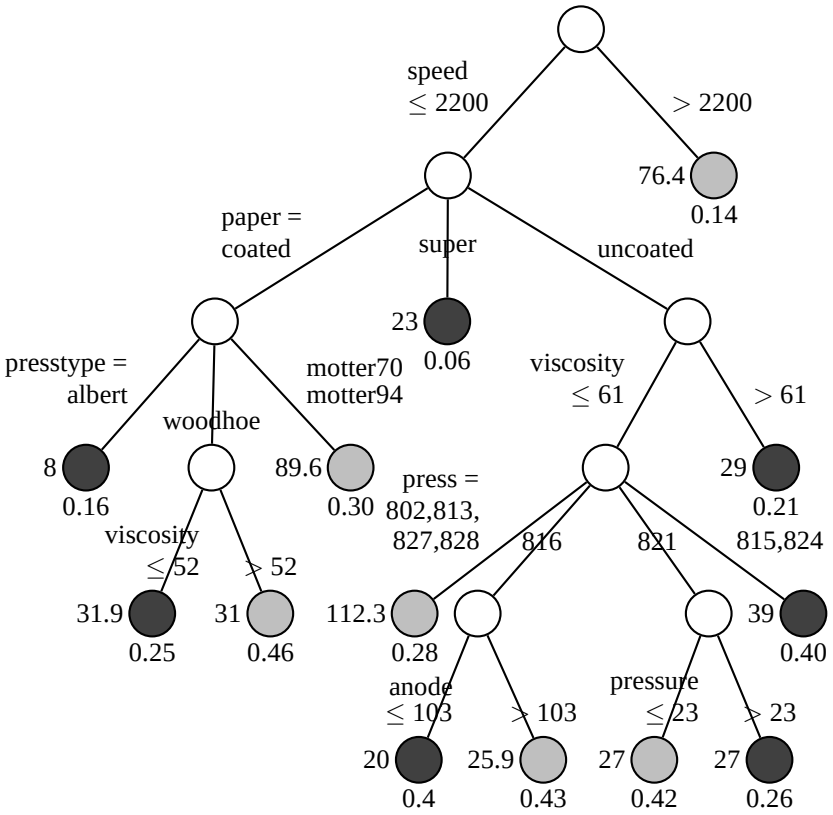

Figure 2: C4.5 tree with nodes predicted as "band" and "noband" colored dark and light gray, respectively. The sample size and error rate are printed on the left and bottom of each terminal node.

is the reduction in entropy attributable to the non-missing observations. The split that maximizes the *gain ratio*, $-g_s(t)/\sum_{k=0}^{m} q(t_k)\log_2 q(t_k)$, is selected. After the split is chosen, t_0 is removed and each observation missing X is sent to every t_k, with weight proportional to $q(t_k)$, ($k = 1, 2, \ldots, m$).

C4.5 uses a binary split of the form $X \leq c$, if X is an ordered variable. But if X is categorical, C4.5 creates one branch for each category. As a result, C4.5 is very fast in splitting on categorical variables. On the other hand, if X has many categories, the branches may be so thinly populated that no further splits are possible. To protect somewhat against this contingency as well as to prune the tree, C4.5 uses a heuristic formula to estimate the error rates of each intermediate node and its branches. A node is pruned if its error rate increases with splitting. Further, it is replaced by its most frequently used branch if the error rate of the branch is smaller than if the node is split.

Figure 2 shows the C4.5 tree for the banding data, after terminal nodes having the same predicted class are combined. The first split is on `speed`, with values > 2200 predicted as "noband." Subsequent splits are on `paper`, `presstype`, `viscosity`,

and `press`. Three nodes are split into three or more branches and the tree misclassifies 116 training samples.

GUIDE

GUIDE is the latest in an evolution that started with FACT [14] and progressed through QUEST [13] and CRUISE [9, 10]. Instead of simultaneously searching for X and the split point c or split set S, FACT breaks the task into two steps: select the X that has the most significant analysis of variance (ANOVA) F-test and apply LDA to the selected X to find c or S. To allow categorical variables, each is first converted to an ordered one by mapping it to a dummy vector and then projecting the latter onto the largest discriminant coordinate. Because LDA is used twice on each categorical variable but only once on each ordered variable, a selection bias toward categorical variables results. QUEST corrects the problem by using chi-square tests for the categorical variables instead of F-tests and it employs quadratic discriminant analysis to find c. CRUISE uses chi-square tests throughout, after discretizing each ordered variable into a categorical one. It tests for main effects as well as interactions between pairs of X variables at each node, and it splits each node into several branches, with one for each class.

GUIDE uses the same chi-square tests as CRUISE, but it controls for the multiplicity of the main effect and interaction tests. Unlike CRUISE and QUEST, both of which use imputation to deal with missing values, GUIDE solves the problem of missing categorical values by assigning a special category for them. A missing category is also created for each ordered variable during computation of the chi-square tests, but after X is selected, its missing values are mapped to negative infinity for split point selection. Thus missing values are always sent to the left branch by the split $X \leq c$, with the choice $c = -\infty$ equivalent to sending only missing values to the left. This is useful for detecting whether a variable is missing at random. An example appears in the GUIDE tree shown on the right side of Figure 1. Variable `inkpct` is chosen to split the root node with $c = -\infty$ because it has 56 missing values all belonging to class "band." The second split on `press` has the same form as the first split of the RPART tree. The GUIDE tree misclassifies 165 training samples.

Selection bias

Which classification tree is best? In terms of error rates, the RPART and GUIDE trees are essentially even while C4.5 misclassifies about one third less. It is unwise, however, to judge classifiers based on error rates that are estimated from the training samples, because the rates can be driven arbitrarily low by over-partitioning the data. Published empirical results indicate that the average error rate of C4.5 is slightly higher than that of GUIDE and slightly lower than that of RPART [12]; see Lim et al. [11], where C4.5 is compared with TREE [18], another implementation of CART.

What can we infer from the trees about the importance of the variables? After all, one supposed advantage of classification trees is their ability to answer such questions. For example, we may be tempted to conclude from the RPART tree that `press` and `speed` are the most important variables. But `speed` does not appear in the GUIDE tree, which selects `inkpct` first. The situation is made muddier by the C4.5 tree, which selects five other variables instead of `inkpct`. Ordering the variables by their importance is a task fraught with difficulties. Chief among them is that the concept of importance is not well defined. CART has a formula for assigning importance scores based on surrogate splits and GUIDE has one based on chi-square statistics. But the orderings usually do not agree completely. Even if the ordering is the same, we cannot expect that the variables will be selected for the splits in that order. This is because only one variable can be selected to split a node. If two variables are equally important and are highly correlated, only one can be selected. Now if their effects are exhausted by the split, the variable that is not selected may never appear in the tree.

Another difficulty with inferring the importance of variables from the trees is the selection bias of CART and C4.5. Doyle [5] was the first to warn that greedy search techniques are biased toward selecting variables with more splits. For example, the variable `press`, which takes 8 categorical values, allows $2^{8-1} - 1 = 127$ splits but the variable `speed`, which takes 83 ordered values, allows 82 splits. Even worse, a binary-valued variable such as `grain` has only one split. Therefore in the hypothetical event that all the predictor variables are independent of the class variable, `press` is more likely than `speed`, which in turn is more likely than `grain`, to reduce the CART node impurity by splitting the node.

An easy way to verify the selection bias is by simulation. Let the class variable take two equally likely values and let there be six predictor variables, with X_1, X_2, and X_3 having 2, 6, and 10, respectively, equi-probable categorical levels, X_4 and X_5 ordered and uniformly distributed on the integers 1–5 and 1–50, respectively, and X_6 uniformly distributed on the unit interval. Thus the numbers of splits allowed by $X_1, \ldots, X_6$ are 1, 31, 511, 4, 49, and 499, respectively. Let all the variables, including class, be mutually independent, so that each variable is selected to split the root node with probability 0.167 if there is no selection bias. Table 2 shows the actual probabilities for the three algorithms, estimated from 100,000 simulation trials for sample size $N = 500$. The results are reported for three situations: no missing values, 90% randomly missing values in X_1 and X_4, and 90% randomly missing values in X_3 and X_6. Clearly, C4.5 is strongly biased toward categorical variables, with the bias increasing with the number of categories. Among ordered variables, C4.5 prefers those with fewer distinct values. RPART is consistently biased toward variables with more splits, regardless of whether they are categorical or ordered. But it, too, has a preference for categorical variables—X_3 is selected more than twice as often as X_6, even though they have about the same numbers of splits.

For C4.5, missing values in X_1 and X_4 (the categorical and ordered variables with the fewest splits) appear to slightly increase their selection probabilities. But if the missing values occur in X_3 and X_6 (the variables with the most splits), C4.5 overwhelmingly selects X_3 with probability 0.965 and doubles its selection probability of X_6 from 0.003 to 0.007. For RPART, missing values in X_1 and X_4 have little effect. But missing

Table 2 Variable selection probabilities for a two-class problem with 500 observations, from 100,000 simulation trials.

Algorithm	X_1	X_2	X_3	X_4	X_5	X_6
			No missing values			
C4.5	0.042	0.275	0.620	0.044	0.016	0.003
RPART	0.009	0.132	0.469	0.033	0.132	0.226
GUIDE	0.168	0.161	0.161	0.170	0.169	0.171
			90% randomly missing X_1 and X_4			
C4.5	0.049	0.274	0.609	0.054	0.015	0.003
RPART	0.009	0.134	0.471	0.029	0.133	0.224
GUIDE	0.163	0.165	0.161	0.165	0.174	0.173
			90% randomly missing X_3 and X_6			
C4.5	0.004	0.013	0.965	0.006	0.004	0.007
RPART	0.012	0.159	0.520	0.042	0.160	0.107
GUIDE	0.167	0.164	0.149	0.179	0.176	0.164

X_1, X_2, and X_3 are categorical with 2, 6, and 10 equi-probable categories. X_4, X_5, and X_6 are ordered, with X_4 and X_5 uniformly distributed on the integers 1–5 and 1–50, respectively, and X_6 is normal. All variables, including class, are mutually independent. An unbiased method selects each variable with probability 0.167.

values in X_3 and X_6 increase the selection probability of the former from 0.469 to 0.520 and decrease that of the latter from 0.226 to 0.107. GUIDE is considerably more robust; its selection probabilities are all around the equi-probable value of 0.167, with or without missing values.

Conclusion

We focused on three tree-structured classifiers to highlight their major features and differences. Two of them are pioneers in the field and the third is representative of a new generation of algorithms designed to guard against selection bias—see Hothorn et al. [8] and Strobl et al. [19] for other approaches. In terms of tree structure, CART and GUIDE always produce binary trees while C4.5 tends to give larger and more sprawling trees, especially when there are categorical variables. For interpretability, the most desirable tree is one that is neither too small (because it provides little information) nor too big (because it can be challenging to follow the logic contained in several levels of splits). In this respect, CART and GUIDE have the advantage.

Selection bias should be a serious concern for anyone wishing to interpret a tree. But if prediction accuracy is all that matters, the bias often does no harm, provided the sample size is large. This is because the damage due to an uninformative split can be repaired by good splits later on. It is possible, however, to get better accuracy by using an ensemble of trees. An increasing

body of literature suggests that the majority vote from a large ensemble of trees often makes better predictions than that from a single member of the ensemble [3, 4]; see Loh [12] for some results on the extent of improvement from ensembles of CART and GUIDE trees.

References

[1] A. Asuncion and D. J. Newman. UCI machine learning repository, 2007. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

[2] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. CRC Press, 1984.

[3] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[4] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[5] P. Doyle. The use of Automatic Interaction Detector and similar search procedures. *Operational Research Quarterly*, 24:465–467, 1973.

[6] B. Evans and D. Fisher. Overcoming process delays with decision tree induction. *IEEE Expert*, 9(1):60–66, 1994.

[7] A. Fielding. Binary segmentation: The automatic detector and related techniques for exploring data structure. In C. A. O'Muircheartaigh and C. Payne, editors, *The Analysis of Survey Data, Volume I, Exploring Data Structures*. Wiley, New York, 1977.

[8] T. Hothorn, K. Hornik, and A. Zeileis. Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical Statistics*, 15(3):651–674, 2006.

[9] H. Kim and W.-Y. Loh. Classification trees with unbiased multiway splits. *Journal of the American Statistical Association*, 96:589–604, 2001.

[10] H. Kim and W.-Y. Loh. Classification trees with bivariate linear discriminant node models. *Journal of Computational and Graphical Statistics*, 12:512–530, 2003.

[11] T.-S. Lim, W.-Y. Loh, and Y.-S. Shih. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine Learning*, 40:203–228, 2000.

[12] W.-Y. Loh. Improving the precision of classification trees. *Annals of Applied Statistics*, 2009.

[13] W.-Y. Loh and Y.-S. Shih. Split selection methods for classification trees. *Statistica Sinica*, 7:815–840, 1997.

[14] W.-Y. Loh and N. Vanichsetakul. Tree-structured classification via generalized discriminant analysis (with discussion). *Journal of the American Statistical Association*, 83:715–728, 1988.

[15] J. N. Morgan and J. A. Sonquist. Problems in the analysis of survey data, and a proposal. *Journal of the American Statistical Association*, 58:415–434, 1963.

[16] C. Perlich, F. Provost, and J. S. Simonoff. Tree induction vs. logistic regression: a learning-curve analysis. *Journal of Machine Learning Research*, 4:211–255, 2003.

[17] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, 1993.

[18] B. Ripley. *tree: Classification and regression trees*, 2007. R package version 1.0-26.

[19] C. Strobl, A.-L. Boulesteix, and T. Augustin. Unbiased split selection for classification trees based on the Gini index. *Computational Statistics and Data Analysis*, 52:483–501, 2007.

[20] T. M. Therneau and B. Atkinson. *rpart: Recursive partitioning*, 2008. R port by B. Ripley. R package version 3.1-41.

Further Reading

H. Ahn, H. Moon, M. J. Fazzari, N. Lim, J. J. Chen, and R. L. Kodell. Classification by ensembles from random partitions of high-dimensional data. *Computational Statistics and Data Analysis*, 51:6166–6179, 2007.

H. A. Chipman, E. I. George, and R. E. McCulloch. Bayesian CART model search (with discussion). *Journal of the American Statistical Association*, 93:935–960, 1998.

A. Ciampi. Generalized regression trees. *Computational Statistics and Data Analysis*, 12:57–78, 1991.

W.-Y. Loh. Regression trees with unbiased variable selection and interaction detection. *Statistica Sinica*, 12:361–386, 2002.

I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques, 2nd ed.* Morgan Kaufmann, San Francisco, 2005.

Notes

CART® is a registered trademark of California Statistical Software, Inc.