CENTER FOR
PARALLEL OPTIMIZATION

DECISION TREE CONSTRUCTION VIA LINEAR PROGRAMMING

by

Kristin P. Bennett

Computer Sciences Technical Report #1067

January 1992

Decision Tree Construction via Linear Programming *

Kristin P. Bennett

Computer Sciences Department
University of Wisconsin
1210 West Dayton Street
Madison, Wisconsin 53706
Phone (608)262-6619, Fax (608) 262-9777
Email bennett@cs.wisc.edu

January 1992

Abstract

Adding linear-combination splits to two-class decision trees allows multivariate relations to be expressed more accurately and succinctly than univariate splits alone. We propose the use of linear programming for determining linear combination splits within two-class decision trees. The problem of determining an optimal linear combination split to distinguish two sets can be formulated as a single linear program. Fast and powerful techniques such as simplex and interior point methods exist for quickly solving such problems. The linear programming approach eliminates the problems of stopping criteria and local minima that plague gradient and perceptron approaches. Computational comparison of the linear programming tree algorithm and classical univariate split algorithms indicates that the linear programming approach produces smaller trees that generalize well.

Keywords: Decision trees, Linear programming, Empirical evaluation

*The work described in this paper has been partially supported by Air Force Laboratory Graduate Fellowship SSAN 531-56-2969, Air Force Office of Scientific Research Grant AFOSR-89-0410 and National Science Foundation Grant CCR-9101801.

1 INTRODUCTION

Tree-structured classification algorithms such as CART [3] and ID3 [18] have proven to be powerful and effective methods for handling classification problems. Typically, decision trees are limited to univariate splits, that is splits based on a single variable. While such splits do make the tree easier to interpret logically, complex trees may be required to express multivariate relations. Geometrically, each leaf of the tree corresponds to a polyhedral region in the original observation space. Trees based on univariate splits limit the faces of these polyhedral regions to the planes orthogonal to the univariate axes and may require many more regions to capture a particular relation. The CART package, perceptron trees [6], and neural tree networks [22] all utilize linear-combination splits. However, there are potential difficulties with these splitting algorithms. These difficulties are discussed in Section 2. Finding the best linear combination split can be posed as a linear program (LP) that minimizes a weighted sum of the misclassification errors. This is an attractive approach because it avoids local minima and there are powerful algorithms for solving LPs efficiently.

The paper is organized as follows. Section 2 discusses the linear programming formulation. A brief description of algorithms that can be used to solve the problem is given. Comparisons of the LP approach with other linear-combination splitting methods are made. Section 3 describes an implementation of the LP decision tree approach. Section 4 contains results of experiments comparing the LP approach with CART and C4.5 [18, 20]. Section 5 concludes with a summary and directions for future work.

2 LINEAR PROGRAM APPROACH

The problem of finding an optimal linear-combination split is equivalent to finding a separating plane that minimizes some measure of misclassification error. In this section, an LP formulation [2] of such a problem is proposed that has the following properties:

(i) If the two classes are linearly separable, i.e. points of each class lie on the opposite sides of some plane, such a "strictly separating" plane is found.

(ii) If the two classes are not linearly separable, a plane is always obtained that minimizes some measure of misclassified points.

(iii) No extraneous constraints are imposed on the linear program which rule out any problem from consideration.

We note that not all linear programming approaches possess these properties [24, 10, 9].

We first describe the notation that we shall employ. For a vector x in the n-dimensional real space R^n, x_+ will denote the vector in R^n with components $(x_+)_i := \max\{x_i, 0\}$, $i = 1, \ldots, n$ (the *plus* function). The notation $A \in R^{m \times n}$ will signify a real $m \times n$ matrix. For such a matrix, A' will denote the transpose, while A_i will denote its ith row. The 1-norm of x, $\sum_{i=1}^{n} |x_i|$, will be denoted by $\|x\|_1$. A vector of ones in a real space of arbitrary dimension will be denoted by e.

2.1 LINEAR PROGRAM FORMULATION

Let the two classes be represented by the two point-sets $\mathcal{A}$ and $\mathcal{B}$ in the n-dimensional real space R^n. Each training example in $\mathcal{A}$ and $\mathcal{B}$ is represented by a row of the $m \times n$ matrix A and the $k \times n$ matrix B respectively. When the sets $\mathcal{A}$ and $\mathcal{B}$ are linearly **separable** the goal is to find a "strictly separating plane" by solving the inequalities:

$$Aw > e\gamma, \quad e\gamma > Bw,$$

where w is an n-dimensional "weight" vector representing the normal to an optimal "separating" plane, and γ is a real number which is a "threshold" that locates the strictly separating plane $wx = \gamma$. If a point A_i in $\mathcal{A}$ is correctly classified, then $-A_iw + \gamma < 0$ and consequently $(-A_iw + \gamma)_+ = 0$. If A_i is incorrectly classified then $(-A_iw + \gamma)_+ \geq 0$. Similarly, $(B_iw - \gamma)_+$ provides a measure of misclassification for a point B_i in $\mathcal{B}$.

When the sets are linearly **inseparable**, an optimal separating plane is defined as a plane that minimizes a weighted sum of the misclassifications. Such an optimal

plane can by obtained by solving the following minimization problem:

$$\min_{w\neq 0,\gamma} \quad \frac{1}{m}\|(-Aw+e\gamma)_+\|_1+\frac{1}{k}\|(Bw-e\gamma)_+\|_1 \tag{2.1.1}$$

The constraint $w \neq 0$ is essential. Without it the point, $w = 0$, $\gamma = 0$, is an optimal solution and no error-minimizing plane is obtained. However, it is this constraint that makes it difficult to formulate an LP with the desired properties (i)-(iii) above. In order to overcome this difficulty we modify (2.1.1) as follows [2]:

$$\min_{w,\gamma} \frac{1}{m}\|(-Aw+e\gamma+e)_+\|_1+\frac{1}{k}\|(Bw-e\gamma+e)_+\|_1 \tag{2.1.2}$$

Problem (2.1.2) will always generate a *strictly* separating plane $wx = \gamma$ for linearly separable sets $\mathcal{A}$ and $\mathcal{B}$. The added term e in the error measure ensures that no points of either class will be directly on the separating plane for the linearly separable case. For linearly inseparable sets $\mathcal{A}$ and $\mathcal{B}$, (2.1.2) will generate an optimal separating plane $wx = \gamma$, with $w \neq 0$, that minimizes the average violations

$$\frac{1}{m}\sum_{i=1}^{m}(-A_iw+\gamma+1)_+ + \frac{1}{k}\sum_{i=1}^{k}(B_iw-\gamma+1)_+.$$

Points of $\mathcal{A}$ which lie on the wrong side of the plane $wx = \gamma + 1$, that is in $\{x|wx < \gamma + 1\}$, and points of $\mathcal{B}$ which lie on the wrong side of the plane $wx = \gamma - 1$, that is in $\{x|wx > \gamma - 1\}$, are the only points that contribute to the violations. Figure 1 depicts an actual error-minimizing plane $wx = \gamma$ obtained by minimizing (2.1.2).

Problem (2.1.2) can be easily solved when it is transformed [2] to the following equivalent linear program.

$$\min_{w,\gamma,y,z} \{\frac{1}{m}ey+\frac{1}{k}ez | y \geq -Aw+e\gamma+e,\ z \geq Bw-e\gamma+e,\ y \geq 0,\ z \geq 0\} \tag{2.1.3}$$

We briefly mention the desirable properties of LP (2.1.3) and recommend that the reader consult [2] for a complete discussion and proofs. LP (2.1.3) satisfies all of the properties (i)-(iii) listed above. The planes $wx = \gamma + 1$ and $wx = \gamma - 1$ used to calculate the misclassification error are arbitrary in the sense that they can be replaced by $wx = \gamma + \zeta$ and $wx = \gamma - \zeta$ for any positive ζ. The linear program

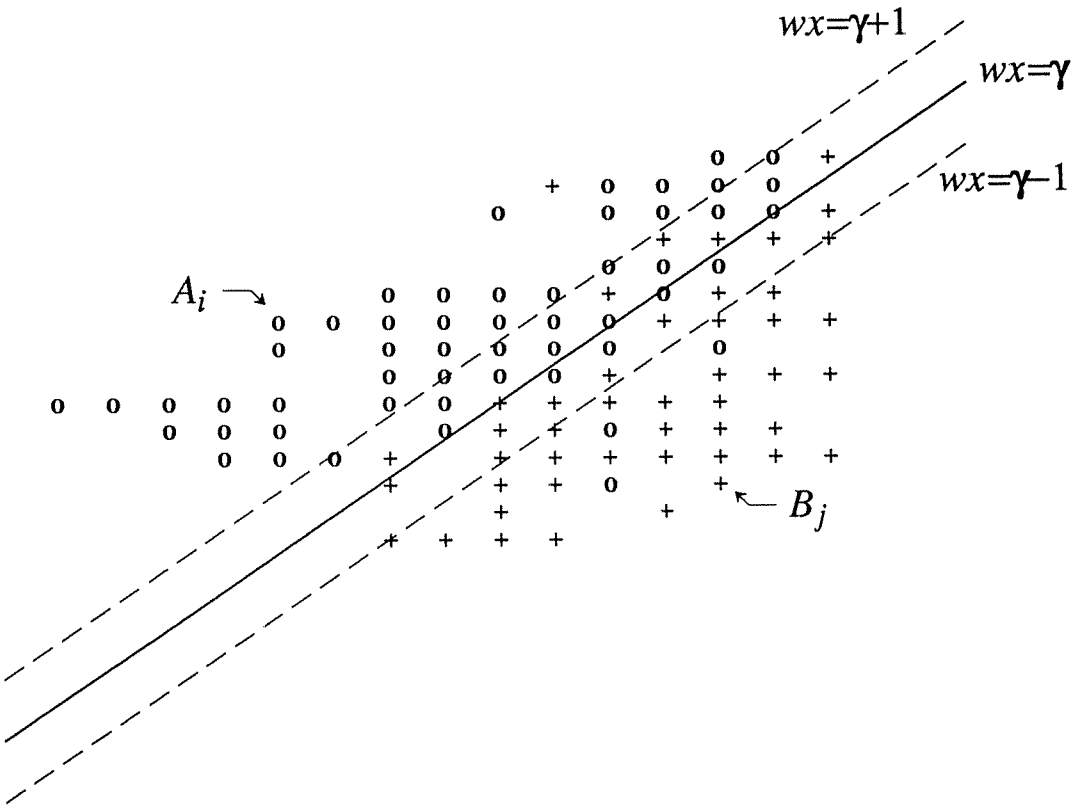

Figure 1: Linear-Combination Split by Linear Programming

(2.1.3) will generate the same error-minimizing solution $wx = \gamma$ for any $\zeta > 0$. The weights of $\frac{1}{m}$ and $\frac{1}{k}$ on the sums ensure that a nontrivial w is always generated without imposing any extraneous constraints. Computational results on real-world problems show that the LP (2.1.3) is preferable to other [24, 10, 9] LP-based approaches for linearly inseparable sets.

2.2 LINEAR PROGRAMMING SOLUTION

LP (2.1.3) can be solved by taking advantage of various powerful computational algorithms. The most widely used method is the simplex method originally proposed by George Dantzig [4] in the 1940s. It works as follows. The feasible region of a linear program is a convex polyhedron. If an LP has an optimal solution, it has an optimal solution which is a vertex of this polyhedron. The simplex method traverses vertices of the polyhedron until an optimal vertex is reached. The algorithm is guaranteed to find an optimal vertex or detect that no optimal solution exists. In the worst case the method is exponential. However in practice it works very fast. Other methods have been developed that work as well or better on large problems (millions of variables). Interior point methods [11, 12] find an optimal solution in polynomial time. They traverse the interior of the feasible region of the problem directly to an optimum point, thus the name. Both algorithms have well-defined, easy-to-check stopping criteria for global optimality.

2.3 OTHER LINEAR-SPLITTING METHODS

Other decision-tree algorithms have used variants of the back propagation algorithm [21] altered to fit the 1-norm error measure, variants of the perceptron algorithm, and heuristic searches. CART uses a heuristic search algorithm which is computationally costly and is prone to local minima [3]. Utgoff [6] proposes using the perceptron algorithm with modifications to address the cycling problem [16, 8]. Basically, since the perceptron algorithm fails to converge for the linearly *inseparable* case, stopping conditions are more difficult to determine and there is no guarantee that an optimal solution will be found. Sankar and Mamonne's neural tree network [22] uses back propagation [15] modified to use the sum of the absolute value of the errors to train each unit. It suffers from the usual difficulties of back propagation: choice of parame-

ters, local minima, and stopping conditions. The advantage of the LP approach used with the simplex method is that there are no parameters, no problems with local minima or convergence, and it has well-defined, easy-to-check stopping conditions.

3 THE LP TREE ALGORITHM

We call the LP-based tree algorithm multisurface method - tree (MSMT) because it uses multiple surfaces to construct the tree and is related to the multisurface method of pattern recognition [13, 14]. For each node in the tree, the best split of the points reaching that node is found by solving LP (2.1.3) using the simplex method. The node is split into two branches, and the same procedure is applied until there are mostly points of one class at the node or there are too few points at the node. In practice, we split the most impure nodes first, as measured by the information function popularized by ID3, and limit the tree to at most 10 linear-combination splits. The leaf nodes are assigned the classification of the majority of points at that node. Pruning strategies improve generalization in decision trees [17]. Thus we adopted the pessimistic pruning strategy used in C4.5 proposed by Quinlan [19, 20]. This strategy, which does not require any additional data or cross-validation, works by pruning a node of the completed subtree if the misclassification cost of a subtree is greater than the misclassification cost of its root. This misclassification cost is based on the re-substitution estimate of the misclassification cost statistically modified to account for the fact that the estimate is overly optimistic.

4 COMPUTATIONAL RESULTS

In this section we give computational comparisons on several real-world databases: the Wisconsin Breast Cancer Database [14, 25], the Cleveland Heart Disease Database [7], and the Bank Failure Database [1]. We use MSMT, CART, and C4.5 (the new and improved ID3). The original experimental design was to use the linear-combination feature of CART. Unfortunately, our commercial CART package crashes after extensive computational time whenever the linear-combination feature is invoked. Thus CART used univariate splits in conjunction with a cost-complexity pruning procedure which uses cross-validation. C4.5 used univariate splits with pessimistic pruning. The

windowing feature of C4.5 was disabled because windowing did not seem to improve the results significantly and it could be used to improve any of the three algorithms.

Table 1 summarizes the results on the three databases. The Wisconsin Breast Cancer Database consists of 681 points of which 442 are benign and 239 are malignant, all in a 9-dimensional real space. The Cleveland Heart Disease Database [1] consists of 197 points in a 13-dimensional real space, of which 137 are negative and 60 are positive. Categorical features within this database were converted to ordered integers for MSMT but not for C4.5 and CART. The Bank Failure Database consists of 4751 points in a 9-dimensional real-space with 4311 successful banks and 441 failed banks. This previously unpublished data set, collected by Richard S. Barr of Southern Methodist University and Thomas F. Siems of the Federal Reserve Bank of Dallas, has 9 numeric features which range from 0 to 1. The Bank Failure Database exceeded the space limitations for the CART program so there are no results for CART.

Ten-fold cross validation was used to measure generalization. The data was partitioned into 10 roughly-equal parts. For each part, a decision tree was created using the remaining nine parts and tested on the part. The cross-validation error is the total number of points misclassified on all 10 parts divided by the total number of points in the database. The times reported are the CPU time on a DECStation 5000/125 required to construct and prune one tree averaged over the ten folds. The CART program performs additional computations and was executed on a different machine. Thus no times are reported for the CART algorithm. The percent training set error and the number of leaf nodes reported are the results from using the entire dataset one time.

On the whole, MSMT quickly produced trees with fewer nodes with equivalent or better generalization than the other two methods. Specifically, MSMT produced trees with less cross-validation error than C4.5 on all three databases, and was better than CART on all but the Heart Disease database. Also, MSMT produced smaller trees in terms of leaf nodes than did C4.5 and CART. Dramatic reduction in tree size makes the tree easier to interpret and thus compensates for the slightly more complex linear-combination splits. CART also had smaller trees than C4.5 probably because of its better but more expensive pruning algorithm. MSMT and C4.5 were very fast on the Breast Cancer data and the Heart Disease data. C4.5 is slightly faster especially on

[1] Available via anonymous ftp from ics.uci.edu courtesy of the University of California-Irvine.

COMPUTATIONAL RESULTS

WISCONSIN BREAST CANCER

METHOD	TRAIN ERROR	CV ERROR	LEAF NODES	TIME (secs)
MSMT	2.4%	3.0%	2	6.8
C4.5	2.8%	3.8%	11	3.7
CART	5.3%	5.3%	3	-

CLEVELAND HEART DISEASE

METHOD	TRAIN ERROR	CV ERROR	LEAF NODES	TIME (secs)
MSMT	19.1%	22.6%	2	10.0
C4.5	9.4%	25.9%	28	1.0
CART	16.8%	20.5%	6	-

BANK FAILURE

METHOD	TRAIN ERROR	CV ERROR	LEAF NODES	TIME (secs)
MSMT	6.4%	6.5%	3	156.3
C4.5	5.0%	7.2%	67	261.0

Table 1: **Comparison of MSMT, C4.5, and CART on Three Databases**
TRAIN ERROR := Percent error on entire data set
CV ERROR := Percent cross-validation error (10-fold)

the Heart Disease Database which has categorical variables. C4.5 handles categorical variables very efficiently. MSMT requires that the attributes be either linearized if possible or encoded as binary attributes in a higher dimensional space. This can cause a loss of information and more computation time. Like previous methods for linear-combination splits, MSMT works best on numerical attributes. On the Bank Failure Database, MSMT was much faster than C4.5. This indicates that MSMT can work well on larger data sets.

5 CONCLUSIONS

We have presented a linear programming method of constructing two-class decision trees. Unlike previous linear-combination splitting methods, the LP approach has no problems with local minima, choice of parameters, and convergence criteria. The MSMT algorithm compares favorably with classical decision tree methods in terms of accuracy, training time, and size of trees. Currently, the LP decision trees are limited to two-class problems. However, the LP split can be generalized to produce multi-category splits similar to those proposed for linear machine decision trees [5] and neural tree networks [22]. This extension of the algorithm is currently being explored. We have demonstrated that LP-based decision tree algorithms compare very favorably with other approaches and that they warrant further investigation and application to a variety of machine learning problems.

References

[1] R. S. Barr (Southern Methodist University) and T. F. Siems (Federal Reserve Bank of Dallas), Private Communication to O. L. Mangasarian, July 20, 1990.

[2] K. P. Bennett, and O. L. Mangasarian, "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets", Computer Sciences Technical Report 1054, University of Wisconsin-Madison, 1991.

[3] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*, Wadsworth International, CA, 1984.

[4] G. B. Dantzig, *Linear Programming and Extensions*, Princeton University Press, Princeton, New Jersey, 1963.

[5] P. E. Utgoff and C. E. Brodley, "Linear Machine Decision Trees", COINS Technical Report 91-10, University of Massachusetts - Amherst, 1991.

[6] P. E. Utgoff, "Perceptron Trees: A Case Study in Hybrid Concept Representations", *Connection Science*, Vol 1, No. 4, 1989, pp. 377-391.

[7] R. Detrano, A. Janosi, W. Steinbrunn, M. Pfisterer, J. Schmid, S. Sandhu, K. Guppy, S. Lee, V. Froelicher, "International Application of a New Probability Algorithm for the Diagnosis of Coronary Artery Disease", *American Journal of Cardiology*, 64, 1989, pp. 304-310.

[8] S. Gallant, "Optimal Linear Discriminants", *Proceedings of the International Conference on Pattern Recognition*, IEEE Computer Society Press, pp 849-852.

[9] F. Glover, *Improved Linear Programming Models for Discriminant Analysis,* Manuscript, Center for Applied Artificial Intelligence, University of Colorado, Boulder 1989.

[10] R. C. Grinold, "Mathematical Programming Methods of Pattern Classification", *Management Science*, 19, 1972, pp. 272-289.

[11] N. Karmarkar, "A New Polynomial Time Algorithm for Linear Programming", *Combinatorica*, 4, 1984, pp. 373-395.

[12] L.G. Khachian, "A Polynomial Algorithm in Linear Programming", *Dokl. Akad. Nauk SSR*, 244(5), 1979, pp. 1093-1096.

[13] O. L. Mangasarian, Multisurface Method of Pattern Separation, *IEEE Transactions on Information Theory*, IT-14(6), 1968, pp. 801-807.

[14] O.L. Mangasarian, R. Setiono, and W. H. Wolberg, "Pattern Recognition Via Linear Programming: Theory and Application to Medical Diagnosis", in: *Large-Scale Numerical Optimization*, Thomas F. Coleman nd Yuying Li, editors, SIAM, Philadelphia, 1990, pp. 22-30.

[15] J.L. McClelland and D.E. Rummelhart, *Explorations in Parallel Distributed Processing: A Handbook of Models, Programs, and Exercises*, MIT Press, Cambridge, MA, 1987.

[16] M. Minsky and S. Papert, *Perceptrons: An Introduction to Computational Geometry* (expanded edition), MIT Press, Cambridge, MA, 1972.

[17] J. Mingers, "An Empirical Comparison of Pruning Methods for Decision Tree Induction", *Machine Learning*, 4, 1989, pp. 227-243.

[18] J. R. Quinlan, "Induction of Decision Trees", *Machine Learning*, 1, 1984, pp. 81-106.

[19] J. R. Quinlan, "Decision Trees as Probabilistic Classifiers", in Langley (ED), *Proceedings of Fourth International Workshop on Machine Learning*, Morgan Kaufmann, Los Altos, 1987.

[20] J. R. Quinlan, "Simplifying Decision Trees", *International Journal of Man-Machine Studies*, 27, 1987, pp 221-231.

[21] D.E. Rumelhart, G.E. Hinton, and J.L. McClelland, "Learning Internal Representations", in D.E. Rumelhart and J.L. McClelland (Eds.) *Parallel Distributed Processing*, Vol. I, M.I.T. Press, Cambridge, Massachusetts,1986, pp. 318-362.

[22] A. Sankar and R. J. Mammone, "Speaker Independent Vowel Recognition using Neural Tree Networks", *Proceedings of IJCNN*, Seattle, 1991.

[23] A. Sankar and R. J. Mammone, "Optimal Pruning of Neural Tree Networks for Improved Generalization", *Proceedings of IJCNN*, Seattle, 1991.

[24] F. W. Smith, "Pattern Classifier Design by Linear Programming", *IEEE Transactions on Computers C-17*, 4, 1968, pp. 367-372.

[25] W.H. Wolberg and O.L. Mangasarian, "Multisurface Method of Pattern Separation Applied to Breast Cytology Diagnosis", *Proceedings of National Academy of Sciences*, USA, 87, 1990, pp. 9193-9196.