Pruning Multivariate Decision Trees by Hyperplane Merging

Miroslav KUBAT[1], Doris FLOTZINGER[2]

1 Institute for Systems Sciences, Johannes Kepler University, Altenbergerstr. 69, A-4040 Linz, Austria, e-mail: mirek@cast.uni-linz.ac.at

2 Department of Medical Informatics, Institute of Biomedical Engineering, Graz University of Technology, Brockmanngasse 41, A-8010 Graz, Austria. e-mail: flotzi@dpmi.tu-graz.ac.at

Abstract

Several techniques for induction of multivariate decision trees have been published in the last couple of years. Internal nodes of such trees typically contain binary tests questioning to what side of a hyperplane the example lies. Most of these algorithms use *cut-off pruning* mechanisms similar to those of traditional decision trees. Nearly unexplored remains the large domain of *substitutional pruning* methods, where a new decision test (derived from previous decision tests) replaces a subtree. This paper presents an approach to multivariate-tree pruning based on merging the decision hyperplanes, and demonstrates its performance on artificial and benchmark data.

1 Introduction

Algorithms for decision-tree induction belong to the most successful machine-learning techniques. In most implementations, the decision tests in the internal nodes question the values of single attributes. For many realistic concepts such *univariate* trees tend to be large and inflexible and, therefore, some researchers investigate algorithms for the induction of *multivariate* trees where the decisions are based on n-ary predicates or functions.

One of the first representatives of this category was the system BTC (Chan, 1988) where the decision tests were boolean functions of symbolic attributes. More recent work concentrates on the relatively general class of decision tests based on hyperplanes—see Utgoff (1989), Utgoff and Brodley (1990), Park and Sklansky (1990), or Murthy et al. (1993)[1]. The hyperplane is defined as $c_j + \Sigma w_i x_i = 0$, where x_i is the value of the i-th attribute, w_i is its weight, and c_j is a constant. If the weighted sum of the numeric attributes is larger than c_j, the example is propagated down the left branch. Otherwise, it is propagated down the right branch. An analysis of various aspects of this approach, with an overview of previous work, can be found in Brodley and Utgoff (1994).

[1]Heng Guo and Gelfand (1992) go one step further and place, at the decision nodes, small neural networks capable of modeling nonlinear concept boundaries

Experiments with benchmark data indicate that multivariate decision trees often outperform univariate trees at the cost of decreased interpretability.

To prevent overfitting and to make decision trees more compact, *pruning* is used. Several pruning techniques are described in Breimann et al. (1984), Quinlan (1990, 1993), and Cestnik and Bratko (1991). Typically, some statistical criterion is used to decide whether to replace a subtree with a leaf or branch. In the sequel, we will call these techniques *cut-off* pruning because they simply dismiss decision tests without creating new ones. An alternative approach can be exemplified by the system Fringe (Pagallo, 1989), where a univariate tree is simplified by replacing some of its subtrees with new attributes constructed as Boolean combinations of existing attributes. This may be referred to as *substitutional* pruning.

The fact that multivariate-tree tests are based on n-ary functions implies a rich class of substitutional-pruning methods based on the idea of replacing the set of tests $t_1(x_1, \ldots, x_n), \ldots, t_m(x_1, \ldots, x_n)$ with a single test $t(x_1, \ldots, x_n)$, derived from them. For instance, a pair of hyperplanes can be replaced with a single hyperplane approximating them. The objective of this paper is to show that this *hyperplane merging* leads to smaller trees that can yield higher classification accuracies on unseen data than cut-off pruning.

2 The System Description

In the sequel, we will report an algorithm for multivariate-tree pruning with hyperplane merging (HPM). For simplicity, we will constrain ourselves to domains where the examples are classified into two classes. Moreover, we will ignore such important aspects of multivariate trees as the selection of features for a decision test—for a detailed treatment of this issue see Brodley and Utgoff (1994). We will suppose that any decision test at an internal node of the tree is a linear combination of *all* attributes. Boolean attributes are turned into numeric by putting -1 for *false* and 1 for *true*[2].

The coefficients of the decision tests for this class of problems can be found by pseudoinverse matrices (for more sophisticated approaches addressing more general tasks see the above mentioned papers). Below, we first describe this simple approach to the induction of multivariate decision trees. Then, the HPM-postpruning method is presented.

2.1 Multivariate Tree Induction

The decisions in multivariate trees usually test the sign of a weighted sum of attribute values, $c_j + \Sigma_i w_i x_i$. To determine the weights, we used *pseudoinverse* matrices described in Duda and Hart (1973, Section 5.8).

Denote by $\mathbf{Y}$ the matrix of training examples and assume that each row represents one example and each column represents one attribute. An additional

[2]Symbolic attributes can be turned into Boolean (and then into numeric) by considering the presence or absence of individual attribute-value pairs as *true* and *false*, respectively.

Table 1: Multivariate-Tree Induction

mti(S).

1. Determine, by Formula (6), the coefficients of the hyperplane that splits the space of examples with minimum mean square error;
2. Place the hyperplane at the root. The decisions made at this node will split S into S_L and S_R;
3. If S_L contains examples from more than one class, run *mti*(S_L); otherwise substitute S_L by a leaf with the label of this class;
4. If S_R contains examples from more than one class, run *mti*(S_R); otherwise substitute S_R by a leaf with the label of this class.

column in $\mathbf{Y}$, $C = [111\ldots1]^T$, is provided for the biases c_j. Denote by $\mathbf{b}$ the (column) vector of the classification values of the examples, where 1 denotes a positive example and -1 denotes a negative example. If $\mathbf{w}$ is the (column) weight vector, then the problem is formulated as follows:

$$\mathbf{Y} \cdot \mathbf{w} = \mathbf{b} \tag{1}$$

Generally speaking, Equation (1) cannot be solved by matrix inversion because the number of rows in $\mathbf{Y}$ is usually much larger than the number of columns. Hence, the alternative task is to find the vector $\mathbf{w}$ that minimizes the difference $\mathbf{e}$ between $\mathbf{Y} \cdot \mathbf{w}$ and $\mathbf{b}$:

$$\mathbf{e} = \mathbf{Y} \cdot \mathbf{w} - \mathbf{b} \tag{2}$$

This is equivalent to minimizing the mean square error

$$E = \Sigma_i(\mathbf{y_i} \cdot \mathbf{w} - b_i)^2 \tag{3}$$

where $\mathbf{y_i}$ is the i-th example multiplied by the weight vector $\mathbf{w}$, and b_i is the class of the i-th example.

To find an extreme of a function amounts to finding its gradient

$$\nabla E = \Sigma_i 2(\mathbf{y_i} \cdot \mathbf{w} - b_i)\mathbf{y_i} \tag{4}$$

and setting it equal to zero, which leads to the following matrix formula:

$$2\mathbf{Y^T}(\mathbf{Yw} - \mathbf{b}) = \mathbf{0}, \tag{5}$$

Equation (5) can be rewritten as $\mathbf{Y^T Yw} = \mathbf{Y^T b}$, and from here

$$\begin{aligned} \mathbf{w} &= (\mathbf{Y^T Y})^{-1}\mathbf{Y^T b} \\ &= \mathbf{Y^P b}. \end{aligned} \tag{6}$$

$\mathbf{Y^P} = (\mathbf{Y^T Y})^{-1}\mathbf{Y^T}$ is called the *pseudoinverse* matrix of $\mathbf{Y}$. Standard functions for its effective calculation are provided in most mathematical software packages.

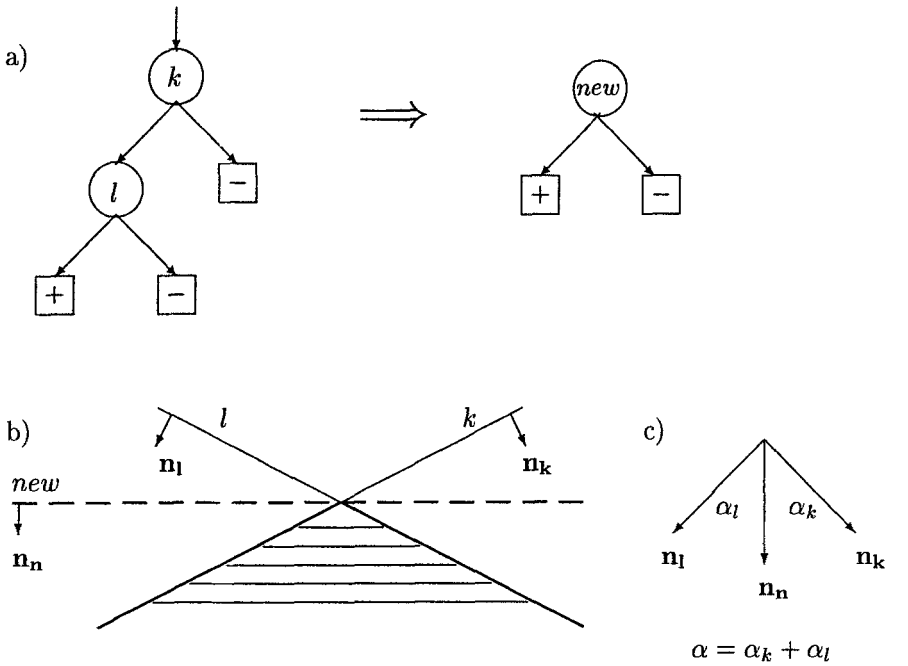

Fig.1. Hyperplane merging. The hyperplanes l and k are substituted by the hyperplane *new*. The shaded area is the original positive subspace. The positive subspace after HPM is below the dashed line. $\mathbf{n_l}, \mathbf{n_k}$ and $\mathbf{n_n}$ are the normal vectors of the respective hyperplanes.

To summarize, knowing the matrix $\mathbf{Y}$ of training examples and knowing the vector $\mathbf{b}$ of their classifications, one can use Formula (6) to determine the weight vector $\mathbf{w}$ minimizing the mean squared error. The only requirement is that $\mathbf{Y^T Y}$ must not be singular, which is satisfied in most cases.

Formula (6) is used within the recursive algorithm for the decision-tree induction, as indicated in Table 1.

2.2 Hyperplane Merging

Figure 1 shows a 2-dimensional illustration of HPM. Two consecutive nodes in the multivariate-tree branch in Figure 1a are represented by two hyperplanes that, if conjuncted, form a subspace boundary (Figure 1b). Cut-off pruning would solely eliminate hyperplane l and the resulting subspace would be bordered by k. HPM, however, replaces the two hyperplanes with a new hyperplane approximating them. Subtree *new* will in many cases better generalize to unseen examples than subtree k.

Let us now specify the HPM procedure (see Figure 1c). The new hyperplane is characterized by its normal vector, $\mathbf{n_n}$, and an arbitrary intersection point of l and k. The normal vector $\mathbf{n_n}$ is a linear combination of the normal vectors of

Table 2: Combined HPM and cut-off pruning algorithm

1. Generate a multivariate decision tree with the algorithm from Table 1;
2. Starting from the bottom level:
   a) For each decision node, attempt its replacement with a leaf;
   b) For each pair of consecutive decision tests, attempt their replacement with a single test using Formula (10), as indicated in Figure 1;
3. Among the replacements attempted in the previous step, select the one with the smallest 'pessimistic error estimate' (Quinlan, 1990). If no error decrease can be achieved, *stop*.
4. Carry out the selected replacement and go to 2.

hyperplanes l and k:

$$\mathbf{n_{n0}} = a_l \cdot \mathbf{n_{l0}} + a_k \cdot \mathbf{n_{k0}} \tag{7}$$

where $\mathbf{n_{n0}}, \mathbf{n_{k0}}$, and $\mathbf{n_{l0}}$ are normalizations of the respective normal vectors to unit length.

The angles between $\mathbf{n_n}$ and $\mathbf{n_l}$ ($\mathbf{n_k}$) can be defined as a percentage of the angle between $\mathbf{n_l}$ and $\mathbf{n_k}$ weighted by the number of examples used to create l (k):

$$\begin{aligned} \mathbf{n_{l0}} \cdot \mathbf{n_{n0}} &= \cos(\alpha_l) \\ \mathbf{n_{k0}} \cdot \mathbf{n_{n0}} &= \cos(\alpha_k) \end{aligned} \tag{8}$$

$$\alpha_l = \frac{\#ex_k \cdot \alpha}{\#ex_l + \#ex_k} \qquad \alpha_k = \frac{\#ex_l \cdot \alpha}{\#ex_l + \#ex_k} \tag{9}$$

where $\#ex_l$ ($\#ex_k$) is the number of examples below hyperplane l (k) and α is the angle between $\mathbf{n_l}$ and $\mathbf{n_k}$.

Substituting (7) into (8), we obtain

$$\begin{bmatrix} \mathbf{n_{l0}^T n_{l0}} & \mathbf{n_{l0}^T n_{k0}} \\ \mathbf{n_{k0}^T n_{l0}} & \mathbf{n_{k0}^T n_{k0}} \end{bmatrix} \cdot \begin{bmatrix} a_l \\ a_k \end{bmatrix} = \begin{bmatrix} \cos(\alpha_l) \\ \cos(\alpha_k) \end{bmatrix} \tag{10}$$

and, from here, the values of coefficients a_l and a_k can easily be found, considering Equations (9).

Note that cut-off pruning can be understood as a special case of HPM, where $\alpha_l = 0$ and $\alpha_k = \alpha$.

The overall pruning algorithm using HPM is described in Table 2. Note that the program has to decide, at each step, whether to use traditional cut-off pruning or hyperplane merging.

3 Experiments

To test the utility of the approach, we experimented with artificial as well as public-domain benchmark data files. Throughout the experiments, we observed whether HPM leads to smaller decision trees with comparable, or even increased, classification accuracy on unseen data. We wanted to know how learning curves are affected and what the behavior in the presence of noise is.

3.1 Experimental Data

Three artificial and four benchmark data were used. Below we provide their brief description and motivation.

Artificial data

To observe the performance on learning concepts with axis-parallel, oblique, and non-linear concept boundaries, we generated the following artificial data sets,

Axis-Parallel Concept (APC). A decision tree involving five numeric attributes from the interval [0,1] was manually drawn. Then 800 examples were constructed so that 50% of them were positive, according to this tree, and 50% were negative. 320 examples were used as a testing set. Note that this concept is not very suited for a learner searching for oblique hyperplanes.

Oblique Concept (OC). The concept was defined as the space between two hyperplanes in a 3-dimensional cube $[0,1] \times [0,1] \times [0,1]$. 3000 examples were generated, 50% of them positive and 50% negative. 1200 examples were used as a testing set.

Nonlinear Concept (NC). The concept was defined as the space inside a sphere positioned in the center of the 3-dimensional cube $[0,1] \times [0,1] \times [0,1]$. 2000 examples were generated, 50% of them positive, 50% negative. 640 examples were used as a testing set.

Public domain benchmark data

We have tested HPM also on some well-known benchmark data. An important methodological note is necessary.

The fact that the number of examples in these data files was relatively small (considering the number of attributes) weakens the validity of error estimates. To insure the statistical reliability of the results, we used the *random subsampling* strategy, as suggested, for instance, by Weiss and Kapouleas (1989): each file is split into two non-overlapping subsets, one for learning and one for testing; the experiments are repeated for several (10 in our case) random splits, and the results are averaged. This methodology is known to provide very good error estimates. In our experiments, the training set contained 60% of the examples, the rest was used as a testing set.

The following public-domain benchmark data files were used:

Table 3: Artificial data: axis-parallel concept (APC), oblique concept (OC), and non-linear concept (NC). Classification accuracy on unseen examples and tree sizes are shown for cut-off pruning and for HPM

	Classif.accuracy		Tree size	
	cut-off	HPM	cut-off	HPM
APC	71.3	72.7	101.0	87.4
OC	97.8	97.7	54.6	48.8
NC	82.8	83.4	281.6	233.6

Congressional Vote. 435 examples described by 16 Boolean attributes. Values *false* are treated as -1 and values *true* are treated as 1. No missing values. The noise was generated in such a way that in $n\%$ of the cases (where n is the noise level) the attribute value was flip-flopped to its opposite.

Hepatitis. 155 examples described by 19 mixed Boolean/numeric attributes. Missing values are replaced with medium values (0 in the case of a Boolean variable).

Liver disorders (BUPA). 345 examples described by 7 numeric attributes. No missing values.

Chess domain. 3197 examples described by 36 symbolic attributes. No missing values.

3.2 Results

Table 3 summarizes the classification accuracy and size of the trees induced from the artificial data. Two kinds of decision trees are compared: multivariate trees with the traditional cut-off pruning, and multivariate trees with HPM. As the examples are noise-free, an increase in classification accuracy was not the primary objective. Rather, we wanted to see, for different shapes of concept boundaries, whether the tree size is reduced by HPM without detriment to performance. This is indeed the case in all of the three domains. The trees are more compact and the classification accuracy after HPM is comparable to that achieved with cut-off pruning.

Table 4 summarizes results achieved on benchmark data. In the hepatitis domain, HPM clearly outperformed cut-off pruning in terms of accuracy, while in the other domains the accuracy remained virtually unchanged. However, in three domains the tree size is dramatically reduced. For instance, in the chess domain, the program was able to find (in all 10 runs) a single decision test providing about the same classification accuracy as cut-off pruning and only about 2 percent deterioration of the accuracy as compared to the original tree that contained, on average, 56 nodes (the chess domain is noise-free). Interestingly, such decision test was not found with the pseudoinverse matrix minimizing the mean square error.

Table 4: Learning from benchmark data under various approaches to pruning: no pruning, cut-off pruning, and hyperplane merging

	Classif.accuracy			Tree size		
	no prng	cut-off	HPM	no prng	cut-off	HPM
Congr.vote	92.4	96.1	96.1	11.6	1	1
Hepatitis	73.1	72.1	77.4	8.6	4.8	2.6
Bupa	60.4	60.0	60.7	61.8	54.8	51.4
Chess	95.8	94.2	94.0	56.0	5.0	1.0

Table 5: Congressional vote in the presence of noise: no pruning, cut-off pruning, and hyperplane merging

	Classification accuracy			Tree size		
noise	no prng	cut-off	HPM	no prng	cut-off	HPM
0%	92.4	96.1	96.1	11.6	1	1
5%	90.4	93.7	93.3	14.2	1	1
10%	86.4	90.2	89.1	17.4	5.8	4.2
15%	85.9	87.9	89.5	18.8	9.6	4.6
20%	83.1	85.1	87.6	20.4	12.8	5.2
25%	77.7	78.7	79.5	23.2	17.0	15.0

Table 5 contains similar results for the congressional-vote data corrupted with noise. HPM performance is more evident for noise levels exceding 10%.

A series of supplementary experiments have revealed that the method of pruning did not affect the learning curves of the systems. In all domains, hyperplane merging provided somewhat better accuracy by arbitrary number of learning examples.

It should be noted that the results provided by the program are not always superior to those that can be achieved by other machine learning systems. Here we concentrate exclusively on the effect of pruning techniques in multivariate trees, ignoring the question whether multivariate trees represent the ideal solution for the given data.

4 Conclusion

A technique for *substitutional pruning* of multivariate decision trees has been presented. The approach is based on replacing a pair of decision hyperplanes with a single hyperplane (hyperplane merging, HPM). To verify the feasibility of the basic idea, we implemented a simple system for the induction of multivariate decision trees with subsequent HPM-pruning.

Experiments indicate that this simplification mechanism leads to substantially smaller decision trees than traditional cut-off pruning techniques. In many cases, the simplified trees yielded even better classification accuracies on unseen examples with smaller standard deviations. HPM seems to perform well especially in the presence of noise.

Improvements to the current version are, of course, possible. For instance, the *new* hyperplane in Figure 1 is always positioned at the intersection of the original two hyperplanes, k and l. It is likely that further tuning is possible by moving the substituted hyperplane along its norm vector. Moreover, we believe that alternative algorithms for more general substitutional pruning (not necessarily limited to hyperplanes) can be found and supplemented to existing multivariate-tree generating programs such as OC1, described in Murthy et al. (1993).

Even though the decision hyperplanes of the trees considered in this study always involved *all* features (which is not an ideal solution, as pointed out by Brodley and Utgoff, 1994), the approach can be easily generalized to decision hyperplanes created from subsets of features when the coefficients of the non-involved features are put equal to 0. Also the application of the technique in multiclass domains should not pose serious problems.

Acknowledgement

The first author was supported by the grant S7002-MAT from the Austrian Science Foundation (FWF).

References

L. Breiman, J. Friedman, R. Olshen, and C.J. Stone (1984). *Classification and Regression Trees.* Wadsworth International Group, Belmont, CA

C.E. Brodley and P.E. Utgoff (1994). Multivariate Decision Trees. *Machine Learning* (in press)

B. Cestnik and I. Bratko (1991). On Estimating Probabilities in Tree Pruning. *Proceedings of the European Working Session on Mashine Learning*, Porto, Portugal, March 6–8, 138–150

P.K. Chan (1988). Inductive Learning with BCT. *Proceedings of the 5th International Conference on Machine Learning*, Morgan Kaufmann

R.O. Duda and P.E. Hart (1973). *Pattern Classification and Scene Analysis.* John Wiley & Sons, New York

Heng Guo and S.B. Gelfand (1992). Classification Trees with Neural Network Feature Extraction. *IEEE Transactions on Neural Networks*, 3:923–933

S. Murthy, S. Kasif, S. Salzberg, and R. Beigel (1993). OC1: Randomized Induction of Oblique Decision Trees. *Proceedings of the 11th National Conference on Artificial Intelligence*, Washington, DC

G. Pagallo (1989). Learning DNF by Decision Trees. *Proceedings of the 11th International Joint Conference on Artificial Intelligence, IJCAI'89*

Y. Park and J. Sklansky (1990). Automated Design of Linear Tree Classifiers. *Pattern Recognition* 23:1393–1412

J.R. Quinlan (1990). Probabilistic Decision Trees. In Kodratoff,Y.–Michalski,R.S. (eds.) *Machine Learning: An Artificial Intelligence Approach*, Volume III, Morgan Kaufmann, 140–152

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

P.E. Utgoff (1989). Perceptron Trees: A Case Study in Hybrid Concept Representations. *Connection Science* 1:377–391

P.E. Utgoff and C.E. Brodley (1990): An Incremental Method for Finding Multivariate Splits for Decision Trees. *Proceedings of the 7th International Conference on Machine Learning*, Morgan Kaufmann

S.M. Weiss and I. Kapouleas (1989). An Empirical Comparison of Pattern Recognition, Neural Nets, and Machine Learning Classification Methods. *Proceedings of the 11th International Joint Conference on Artificial Intelligence, IJCAI'89*, Detroit, MI, 781–787