Decision Support Systems 48 (2010) 430–436

Contents lists available at ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

Classification by vertical and cutting multi-hyperplane decision tree induction ☆

Marco Better [a], Fred Glover [a,*], Michele Samorani [b]

[a] OptTek Systems, Inc., 1919 Seventh Street, Boulder, Colorado 80302, United States
[b] Leeds School of Business, University of Colorado at Boulder, Campus Box 419, Boulder, Colorado 80309, United States

ARTICLE INFO

Available online 17 June 2009

Keywords:
Data mining
Discrimination analysis
Piecewise-linear models
Mathematical programming

ABSTRACT

Two-group classification is a key task in decision making and data mining applications. We introduce two new mixed integer programming formulations that make use of multiple separating hyperplanes. They represent a generalization of previous piecewise-linear models that embed rules having the form of hyperplanes, which are used to successively separate the two groups. In fact, the classifiers obtained are particular types of decision trees which are allowed to grow in depth and not in width. Computational results show that our models achieve better classification accuracy in less time than previous approaches.

1. Introduction

One of the key tools for classifying data in real-world data mining applications [15] is Discriminant Analysis. This technique is fundamental to business applications in marketing, accounting, advertizing, sales, manufacturing financial analysis, strategic planning and many other domains [5,6,16,17,19,22,23,26]. It is a crucial component of any decision that involves determining which people, groups, elements or processes to target in order to achieve desired goals or to implement particular policies most effectively. Recent contributions to this area grow out of mathematical programming approaches based on multivariate decision trees, piecewise-linear models, data envelopment analysis and kernel-based classifiers [2,7,10]. These models are commonly formulated with the goal of classifying data into two groups, although they can be easily adapted to accommodate more classes.

The induction of multivariate decision trees (MDTs) consists in finding a binary tree where, at each internal node, the data set is recursively split by a hyperplane into two parts, and where each leaf node is associated with one group, determining in this fashion the predicted group for each object [21]. The methods to build MDTs, such as OC1 [18], are often based on a two-step procedure: a construction step, which consists of a heuristic algorithm that builds the initial MDT, and a pruning step, which aims to reduce its size, enhancing both the accuracy and the interpretability of the final result. Support Vector Machine (SVM) methods [7] usually rely on a limited number of known kernel transformations to project the data set into a high-dimensional space with the hope of rendering it linearly separable. Despite their good performance (provided that a suitable kernel is available), they do not provide interpretable classification rules. Finally, piecewise-linear models solve the classification problem directly, finding a set of hyperplanes that forms, in fact, a MDT. Therefore, they have both advantages of optimality and interpretability. An additional advantage is that these models, including the reigning one, proposed by [12], have stringent requirements that limit their ability to handle complex structures. For example, they require that the elements of one group lie in a convex region, so they must be solved twice: each time constraining one group to lie in the convex region. Recently, [25] have explored the application of discriminant analysis and have compared it to the use of data envelopment analysis in a detailed assessment of corporate failure. The authors additionally employ these tools to analyze performance in the Japanese construction industry in [24]. Our present work may be viewed as an extension of the types of classification methodology used in these studies, and provides a basis for carrying out this classification analysis in greater depth to refine the inferences obtained.

In this paper, in order to provide models that have useful features for classification problems arising in real-world contexts, we present two mixed integer formulations for the induction of MDTs, which we call *Vertical Decision Tree* and *Cutting Decision Tree*. These new formulations build on the sequence of progressively more advanced classification models that began with the original goal programming models of [9]. By this connection, our present work may be viewed as having a link to the goal programming model conceptions of [3] and of [4]. Our models overcome many of the defects of the existing classifiers based on similar techniques. We overcome them by the following efforts:

1. Constructing a MDT through an exact algorithm, without relying on any heuristic and/or pruning procedures;
2. Generalizing piecewise-linear approaches by eliminating the requirement that one group must lie in a convex region, and by using significantly fewer binary variables (about a 50% reduction), and
3. Not projecting the data into high-dimensional space, which avoids the problem of finding a suitable kernel.

☆ This paper is dedicated to William W. (Bill) Cooper, who has served as a mentor for one of us and a source of inspiration for all of us.

* Corresponding author.
E-mail addresses: better@opttek.com (M. Better), michael.samorani@colorado.edu (M. Samorani).

0167-9236/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.dss.2009.06.004

Both models prevent the tree from growing in width by guaranteeing the presence of at least one leaf node at each depth (or level) of the tree. Furthermore, in the case of the *Cutting Decision Tree*, the model requires that the misclassifications – if any – happen only at the maximum depth of the tree. We show that this apparent limitation allows the depth of the tree to be parametrically decided and does not lead to a decrease in classification accuracy. In fact, the *Cutting Decision Tree* model uses fewer binary variables and its solution times are significantly shorter. Our models outperform the one proposed by Glen [10–12], both in terms of accuracy and computing time, and obtain similar or better results than SVMs and OC1. Our models are particularly suitable for those real-world classification problems whose classification rules can be expressed as a logical expression involving many conditions (rather than a single condition), where a "condition" refers to a linear combination of attributes. For example, consider the following logical expression involving 4 attributes a, b, c, and d:

$$a \geq b \text{ AND } c \leq 2d$$

Due to its tree structure, our approach may find a classification rule that uses this expression to discriminate between the two groups of objects. On the other hand, approaches using a single separating hyperplane cannot find this kind of logical pattern. For example, DEA cannot find any set of coefficients that yields the logical expression above.

The next section of the paper provides background information about the classification problem and reviews relevant literature on single and multiple hyperplane formulations that provide the context for our models. Then, we introduce our two new mathematical models mentioned above and report on computational tests that evaluate the accuracy on two well-established benchmark data sets to demonstrate the advantages of our approaches with respect to state-of-the-art alternatives. Finally, we summarize our conclusions and opportunities for future research.

2. Hyperplane-based classification

Let a_{ij} denote the value of a specific characteristic of an element in a data set, where each element i ($i = 1,\ldots, m$) is described by a range of attributes j ($j = 1,\ldots, n$). We seek to classify these elements in such a manner that correctly identifies whether a vector $A_i = (a_{i1},\ldots, a_{in})$ for a given element i should result in classifying the element as belonging to Group 1 or Group 2 (denoted G_1 and G_2, respectively).

The decision rules we investigate here are based on hyperplane separation approaches, viewing the A_i vectors as points in an n-dimensional space. In the simplest case, these approaches use a single hyperplane, characterized by a vector $X = (x_1,\ldots, x_n)$ and a scalar b, to differentiate the points A_i for $i \in G_1$ from the points for $i \in G_2$, according to the following decision rule:

$$A_i X < b \text{ for } i \in G_1 \text{ and } A_i X > b \text{ for } i \in G_2 \quad (1)$$

Previous studies have proposed mixed integer formulations that aim to minimize the number of misclassifications by a single separating hyperplane (see [1,10,11,13,22]). However, these formulations have various deficiencies, and generally require an excessive number of variables. For example, recent formulations [10,11] require doubling the number of original x_j variables by splitting each one into a positive and negative part, and then introducing a normalization constraint that sets the sum of these variables to 1.

On the other hand, the common goal of piecewise-linear models and MDTs is to approximate more complex structures in the underlying data than those that can be achieved by a single hyperplane, as shown in Fig. 1. Thus, we refer to these techniques as Multi-hyperplane approaches.

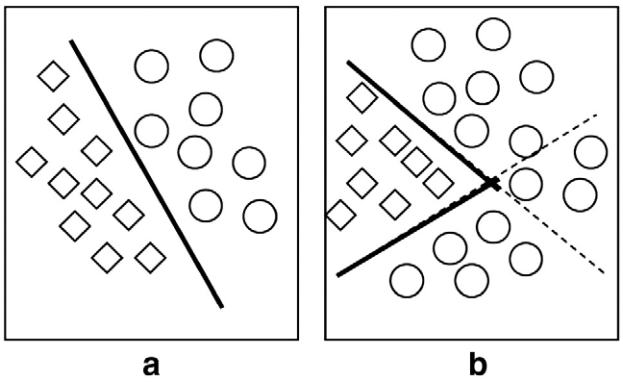

Fig. 1. Single and multiple separating hyperplanes, where we separate *round* elements from *diamonds* elements.

Glen (see [12]) introduced a mixed integer model that constructs a set of piecewise-linear segments in order to separate elements into two groups, requiring that the elements in one of the groups lie in a convex region (i.e. the piecewise-linear segments must form a convex region). Because of the convexity assumption, the model must be solved twice to determine which group should be treated as convex to provide better results. Apart from doubling the number of x_j variables by splitting them into x_j^+ and x_j^- (as in a special ordered set), the model also makes use of additional binary variables in every hyperplane constraint to determine whether the element lies in the convex region or not.

3. Multi-hyperplane models for multivariate decision trees

This section describes our two proposed Multi-hyperplane models, which aim to build a *Vertical Decision Tree* and a *Cutting Decision Tree*, respectively. We begin by defining each type of decision tree as follows:

Definition 1. *Vertical Decision Tree.*

A *Vertical Decision Tree* (VDT) is a MDT for which each internal node is a parent of at least one leaf node.

Thus, at each depth except for its maximum depth d^*, the elements lying on one side of the hyperplane are classified into one of the two groups, while the classification of the residual elements is delegated to the hyperplane at the subsequent depth. Fig. 2 illustrates the structure of a general VDT whose depth is $d^* = 3$ (it uses 3 hyperplanes). We seek to separate points represented as *circles* (which belong to G_1) from points represented as *diamonds* (which belong to G_2). In illustrations throughout this paper, the left child of any internal node corresponds to G_1, and the right child corresponds to G_2. This implies that leaf nodes associated with G_1 can appear only as left children, and leaf nodes associated with G_2 only as right children. Note that this assumption reflects only on the graphical notation that we use, and does not cause any loss of generality. The points are separated by three hyperplanes, denoted by h_1, h_2 and h_3. The decision boundary is formed by segments *PQ*, *QR* and *RS* of the three hyperplanes. The dark diamonds depicted in Fig. 2a, o_1 and o_2, are misclassified by the tree. Fig. 2b indicates the leaf nodes where they *fall*. We consider that an element e "falls into" a (leaf or internal) node n if e encounters n throughout the path made by classifying e [20].

Definition 2. *Cutting Decision Tree.*

A *Cutting Decision Tree* (CDT) is a VDT where misclassifications at intermediate leaves are not allowed.

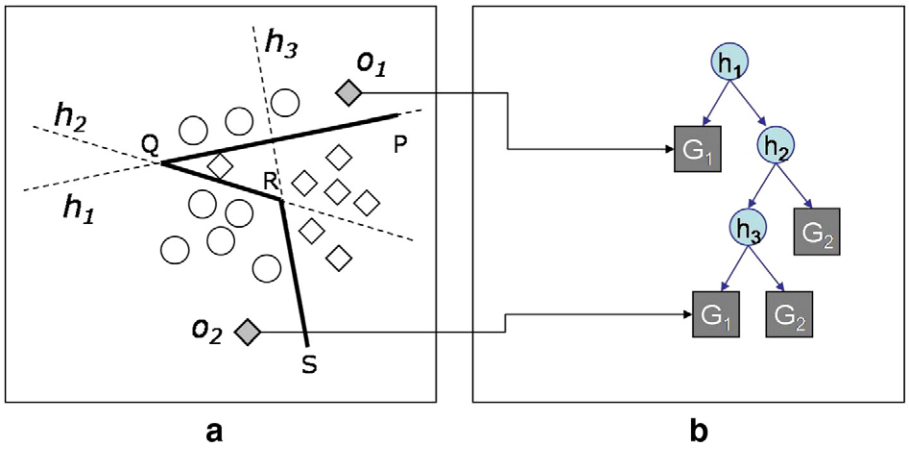

Fig. 2. a) Shows a data set where *circles* belong to G_1 and *diamonds* to G_2; The corresponding VDT with $d^*=3$ is shown in b) the numbers in the leaf nodes indicate the associated group. Objects o_1 and o_2 are misclassified: they fall into the wrong leaf node at respectively depths 1 and 3.

Thus, classification errors – if any – may only occur at maximum depth d^*. The classification procedure aims to "cut" part of the data set at each level by compelling all the elements of one group to lie on one side of the hyperplane. The VDT in Fig. 2b is not a CDT, since o_2 is misclassified at depth $d=1$. However, if we deleted o_2 from the data set the tree would be a CDT, because o_1, which would then be the only misclassified element and falls in a leaf node located at the last level of the tree.

3.1. A model for vertical decision trees

We first provide a model for inducing a VDT. For the sake of simplicity, we limit our formulation to the case of VDTs of depth $d^*=3$. The type of tree induced is determined by the position of its internal nodes and leaf nodes. Fig. 3 shows all possible VDT types for the case where $d^*=3$. The top two rows shown in Fig. 3 denote the meaning of special binary variables sl_1 and sl_2, which we will call *slicing variables*. These variables constitute our structure variables, since they define the particular structure of the induced tree. Specifically, if $sl_1=0$, the tree will be "sliced" to the right at depth 1. Hence, the right node at depth 2 must be a leaf node associated with G_2. Conversely, if $sl_1=1$, the right node will be an internal node and the left node a leaf node associated with G_1. The same logic applies for depth 2, where sl_2 indicates whether the tree is sliced on the right ($sl_2=0$) or on the left ($sl_2=1$).

Fig. 3 lists the 4 types of tree: tree (0, 0), (0, 1), (1, 0) and (1, 1), where the numbers in parenthesis correspond respectively to the value of sl_1 and sl_2. The complete mathematical model for a VDT of maximum depth $d^*=3$ can be expressed as follows:

$$\text{Let } z_i^* = \left\{ \begin{array}{l} 0 \text{ if object } i \text{ is correctly classified by the tree} \\ 1 \text{ otherwise} \end{array} \right\}$$

$$\text{Let } z_{id} = \left\{ \begin{array}{l} 0 \text{ if object } i \text{ is correctly classified by hyperplane } h_d, \text{ according to Eq. (1)} \\ 1 \text{ otherwise} \end{array} \right\}$$

Because the goal is to minimize the number of misclassified elements, the objective function is simply:

$$\text{Minimize} \sum_{i=1}^{n} z_i^*$$

First, we include the hyperplane constraints that are based on inequalities (1) for each depth d of the tree:

$$A_i X_d - M z_{id} \le b_d - \varepsilon \quad \text{for } i \in G_1, d = 1,2,3$$
$$A_i X_d + M z_{id} \ge b_d + \varepsilon \quad \text{for } i \in G_2, d = 1,2,3$$

where ε is a constant quantity that we treat parametrically to induce a "separation zone" in order to prevent points of both groups from ending up exactly on the boundary.

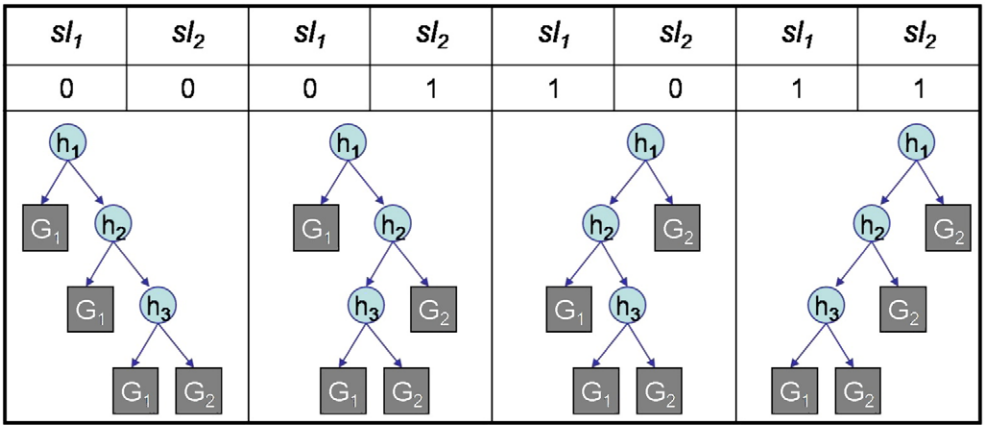

Fig. 3. All possible VDT types for $d^*=3$.

We then include constraints that identify the optimal tree structure for the data set for classification purposes. Thus, for tree type (0,0), we write:

$$M(sl_1 + sl_2) + z_i^* \geq z_{i1} + z_{i2} + z_{i3} - 2 \text{ for } i \in G_1$$
$$M(sl_1 + sl_2) + Mz_i^* \geq z_{i1} + z_{i2} + z_{i3} \text{ for } i \in G_2$$

where M is a large constant. Only when $sl_1 = 0$ and $sl_2 = 0$, the actual constraints will $z_i^* \geq z_{i1} + z_{i2} + z_{i3} - 2$ (for $i \in G_1$) and $Mz_i^* \geq z_{i1} + z_{i2} + z_{i3}$ (for $i \in G_2$) be "activated." So, for a (0,0) tree, an object of G_1 is misclassified only if it is misclassified by all 3 hyperplanes, while an object of G_2 is misclassified if it is misclassified by at least one hyperplane. Similarly, constraints for tree type (1,1) will only be activated when sl_1 and sl_2 are both equal to 1. These constraints are:

$$M(2 - sl_1 - sl_2) + Mz_i^* \geq z_{i1} + z_{i2} + z_{i3} \text{ for } i \in G_1$$
$$M(2 - sl_1 - sl_2) + z_i^* \geq z_{i1} + z_{i2} + z_{i3} - 2 \text{ for } i \in G_2$$

For tree types (0,1) and (1,0), the logic is similar in that the appropriate part of the constraints will be active when the slicing variables have the corresponding value. However, for these types of trees, the decision structure requires additional constraints. We use an additional binary variable w_i to activate or deactivate certain "either-or" constraints. For example, the constraints for tree (0,1) are:

$$M(1 + sl_1 - sl_2) + z_i^* \geq z_{i1} - Mw_i \quad \text{for } i \in G_1$$
$$M(1 + sl_1 - sl_2) + Mz_i^* \geq z_{i2} + z_{i3} - M(1 - w_i) \quad \text{for } i \in G_1$$
$$M(1 + sl_1 - sl_2) + z_i^* \geq z_{i1} \quad \text{for } i \in G_2$$
$$M(1 + sl_1 - sl_2) + z_i^* \geq z_{i2} + z_{i3} - 1 \quad \text{for } i \in G_2$$

When $sl_1 = 0$ and $sl_2 = 1$, these constraints are activated, in which case the variables w_i are used to express the fact that a correctly classified element of G_1 is correctly classified **either** by hyperplane h_1 **or** by **both** hyperplane h_2 **and** hyperplane h_3. An element of G_2, on the other hand, will be correctly classified by the tree if it is correctly classified **either** by hyperplanes h_1 **and** h_2 **or** by hyperplanes h_1 **and** h_3. The case that corresponds to tree type (1,0) is just a mirror image of the previous case.

Let M be a "large" constant as a proxy for an unknown upper bound[1]; let ε be a "small" constant to induce a strict (non-zero) displacement from a hyperplane, and let G be the union of G_1 and G_2. We can now formulate the full $d^* = 3$ VDT model as follows:

$$\text{Minimize} \sum_{i=1}^{n} z_i^* \tag{1.1}$$

Subject to

$$A_i X_d - Mz_{di} \leq b_d - \varepsilon \quad \text{for } i \in G_1, d = 1,2,3 \tag{1.2}$$
$$A_i X_d + Mz_{di} \geq b_d - \varepsilon \quad \text{for } i \in G_2, d = 1,2,3 \tag{1.3}$$
$$M(sl_1 + sl_2) + z_i^* \geq z_{i1} + z_{i2} + z_{i3} - 2 \quad \text{for } i \in G_1 \tag{1.4}$$
$$M(sl_1 + sl_2) + Mz_i^* \geq z_{i1} + z_{i2} + z_{i3} \quad \text{for } i \in G_2 \tag{1.5}$$
$$M(2 - sl_1 - sl_2) + Mz_i^* \geq z_{i1} + z_{i2} + z_{i3} \quad \text{for } i \in G_1 \tag{1.6}$$
$$M(2 - sl_1 - sl_2) + z_i^* \geq z_{i1} + z_{i2} + z_{i3} - 2 \quad \text{for } i \in G_2 \tag{1.7}$$
$$M(1 + sl_1 - sl_2) + z_i^* \geq z_{i1} - Mw_i \quad \text{for } i \in G_1 \tag{1.8}$$
$$M(1 + sl_1 - sl_2) + Mz_i^* \geq z_{i2} + z_{i3} - M(1 - w_i) \quad \text{for } i \in G_1 \tag{1.9}$$
$$M(1 + sl_1 - sl_2) + z_i^* \geq z_{i1} \quad \text{for } i \in G_2 \tag{1.10}$$
$$M(1 + sl_1 - sl_2) + z_i^* \geq z_{i2} + z_{i3} - 1 \quad \text{for } i \in G_2 \tag{1.11}$$
$$M(1 - sl_1 + sl_2) + z_i^* \geq z_{i1} \quad \text{for } i \in G_1 \tag{1.12}$$
$$M(1 - sl_1 + sl_2) + z_i^* \geq z_{i2} + z_{i3} - 1 \quad \text{for } i \in G_1 \tag{1.13}$$
$$M(1 - sl_1 + sl_2) + z_i^* \geq z_{i1} - Mw_i \quad \text{for } i \in G_2 \tag{1.14}$$
$$M(1 - sl_1 + sl_2) + Mz_i^* \geq z_{i2} + z_{i3} - M(1 - w_i) \quad \text{for } i \in G_2 \tag{1.15}$$
$$\sum_{j=1}^{n} \sum_{d=1}^{3} x_{jd} = 1 \quad \textit{normalization} \tag{1.16}$$
$$z_i^* \in \{0,1\} \quad i \in G \tag{1.17}$$
$$z_{id} \in \{0,1\} \quad i \in G,\ d = 1,2,3 \tag{1.18}$$
$$w_i \in \{0,1\} \quad i \in G \tag{1.19}$$
$$sl_k \in \{0,1\} \quad k = 1,2 \tag{1.20}$$
$$x, b \textit{ unrestricted} \tag{1.21}$$

The normalization constraint (1.16) sets the sum of all x_j variables to 1 (or any constant value), and is necessary to avoid problems arising from rotations or translations of the data and to be able to handle negative data. To cover the possible cases, this normalization must utilize both a positive and a negative constant term (i.e., -1 as well as 1). We have chosen the positive variant for convenience, but more general considerations about this and other forms of normalization are discussed in [14].

Although this model performs well for $d^* = 3$ (as shown later in our Experiments and results section), it has a clear limitation: the formulation strongly depends on d^*, which is clearly a disadvantage. Also, as d^* increases the number of tree types increases at the rate of 2^{d^*} and so do the binary *slicing variables*. Given these limitations, we developed a general model for CDTs which, on one hand, introduces the limitation of allowing misclassifications only at depth $d = d^*$. But, on the other hand, it is fully general for any given maximum depth and the number of variables grows linearly with d^*.

3.2. A model for cutting decision trees

For each depth d we use variables x_d, b_d and z_{id}, corresponding to the same variables of the VDT model. Since misclassifications at intermediate depths are not allowed, now the objective function involves only the last level of the tree:

$$\text{Minimize} \sum_{i \in G} z_{id^*}$$

Note that, at each depth d of a CDT, one group is entirely correctly classified. Otherwise, there would be forbidden misclassifications. For each depth $d = 1,\ldots d^* - 1$, a binary variable y_d indicates which group is correctly classified at depth d:

$$y_d = \begin{cases} 0 \text{ if all of } G_1 \text{ is correctly classified by hyperplane } h_d \text{ according to Eq. (1),} \\ 1 \text{ if all of } G_2 \text{ is correctly classified by hyperplane } h_d \text{ according to Eq. (1)} \end{cases}$$

[1] The value of M can be made to depend on the inequality in which it appears, and in cases where problem information or preprocessing allows M to be reduced, the resulting model can often be solved more efficiently by existing solution methods.

This definition implies that if $y_d = 0$, then $z_{id} = 0$ for all $i \in G_1$. Conversely, if $y_d = 1$, then $z_{id} = 0$ for all $i \in G_2$. At any given depth d except the final one,

$y_d \cdot z_{id}$ for all $i \in G_1$

$1 - y_d \cdot z_{id}$ for all $i \in G_2$

so that all elements of either group G_1 or G_2 are correctly classified. Also, variable y_d automatically determines the direction in which the tree grows at depth d: if $y_d = 0$, the tree grows to the left (Fig. 4a). If $y_d = 1$, the tree grows to the right (Fig. 4b). Whether or not element i falls into the leaf node at level d is indicated by the binary variables v_{id} (one for each element $i \in G$ and depth $d = 1,\ldots, d^* - 1$):

$$v_{id} = \begin{cases} 0 \text{ if } i \text{ does not fall into the leaf node at depth } d. \\ 1 \text{ if } i \text{ falls into the leaf node at depth } d. \end{cases}$$

The relationships among y_d, v_{id} and z_{id} are represented in Fig. 4. Considering Fig. 4a (Fig. 4b is analogous), the objects of G_2 falling into the leaf node have $z_{id} = 0$ and $v_{id} = 1$, and the objects of G_2 falling into the internal node at depth $d+1$ have $z_{id} = 1$ and $v_{id} = 0$. Meanwhile, the objects of G_1, which all fall into the internal node at depth $d+1$, have $z_{id} = 0$ and $v_{id} = 0$. These conditions are expressed through the constraints:

$v_{id} \le y_d \quad i \in G_1,\ d = 1,\ldots,d^* - 1$

$v_{id} \le 1 - y_d \quad i \in G_2,\ d = 1,\ldots,d^* - 1$

$v_{id} \le 1 - z_{id} \quad i \in G,\ d = 1,\ldots,d^* - 1$

It follows that the necessary conditions for an element i to be misclassified by a CDT are:

1. It does not fall into any leaf at depth $d < d^*$; that is, $v_{id} = 0$ for each $d < d^*$;
2. The last hyperplane misclassifies it, that is $z_{id^*} = 1$.

Then, the hyperplane constraints become:

$$A_i x_d - M\left(\sum_{h=1}^{d-1} v_{ih} + z_{id}\right) \le b_d - \varepsilon \quad i \in G_1,\ d = 1,\ldots,d^*$$

$$A_i x_d + M\left(\sum_{h=1}^{d-1} v_{ih} + z_{id}\right) \ge b_d + \varepsilon \quad i \in G_2,\ d = 1,\ldots,d^*$$

Thus, the complete formulation of the CDT model is:

$$\text{Minimize} \sum_{i \in G} z_{iD} \tag{2.1}$$

Subject to:

$$A_i x_d - M\left(\sum_{h=1}^{d-1} v_{ih} + z_{id}\right) \le b_d - \varepsilon \quad i \in G_1,\ d = 1,\ldots,d^* \tag{2.2}$$

$$A_i x_d + M\left(\sum_{h=1}^{d-1} v_{ih} + z_{id}\right) \ge b_d + \varepsilon \quad i \in G_2,\ d = 1,\ldots,d^* \tag{2.3}$$

$$y_d \ge z_{id} \quad i \in G_1,\ d = 1,\ldots,d^* - 1 \tag{2.4}$$

$$1 - y_d \ge z_{id} \quad i \in G_2\ d = 1,\ldots,d^* - 1 \tag{2.5}$$

$$v_{id} \le y_d \quad i \in G_1,\ d = 1,\ldots,d^* - 1 \tag{2.6}$$

$$v_{id} \le 1 - y_d \quad i \in G_2,\ d = 1,\ldots,d^* - 1 \tag{2.7}$$

$$v_{id} \le 1 - z_{id} \quad i \in G,\ d = 1,\ldots,d^* - 1 \tag{2.8}$$

$$\sum_{d=1}^{D} \sum_{j=1}^{F} x_{jd} = 1 \quad \text{“}\textit{Normalization}\text{”} \tag{2.9}$$

$$x_d,\ b_d \text{ unrestricted} \quad d = 1,\ldots,d^* \tag{2.10}$$

$$z_{id} \in \{0,1\} \quad d = 1,\ldots,d^* \tag{2.11}$$

$$y_{id} \in \{0,1\} \quad d = 1,\ldots,d^* - 1 \tag{2.12}$$

$$0 \le v_{id} \le 1 \quad d = 1,\ldots,d^* - 1 \tag{2.13}$$

Since the objective functions of both VDT and CDT models aim at minimizing the number of misclassified points, both models may have multiple equivalent optimal solutions of value v, each corresponding to a set of misclassified points whose cardinality is v. A possible improvement is to define an objective function that has the second goal of maximizing the distance of the correctly classified points to the misclassification region.

4. Experiments and results

We tested our models on two benchmark data sets from real-world applications, providing a basis to compare our outcomes with other models. In particular, the performance of the VDT and CDT models is compared to our implementation of the piecewise-linear model proposed by [12], which separates the two groups with 3 hyperplanes and requires one group to lie in a convex region. We first tested the 3 models on the Japanese Banks data set, provided by [23], which Glen showed to be separable by three hyperplanes. For this reason, we used depth $d^* = 3$ for all of our tests. In order to eliminate large discrepancies in scale among the attribute values, we first standardized the data using the following standardization:

$$V_{ij} = \frac{(v_{ij} - \overline{v}_j)}{s_j},$$

where V_{ij} is the standardized value of attribute j for element i, v_{ij} is the original value of attribute j for element i, $\overline{v}_j$ is the sample mean for attribute j and s_j is the sample standard deviation for attribute j.

We used the leave-one-out (LOO) testing procedure, which is an n-fold cross validation, where n is the cardinality of the data set (in the Japanese Banks data set, $n = 100$). The LOO “hit rate” is the percentage of correct classifications of the holdout element with respect to the total number of tests performed [8]. We treated ε as a parameter in all three models (VDT, CDT, and Glen), testing for various “separation zone” widths. We ran Glen's model twice, alternating the convexity requirement between G_1 and G_2, and recorded the outcome that produced the best LOO classification. Even if our main goal is to compare against Glen (see [12]), we also report the results obtained by OC1 and SVM. We used the implementation of OC1 available at http://www.cs.jhu.edu/~salzberg/announce-oc1.html (accessed on 5th December 2008), and the implementation of SVMs called SMO, available in Weka [27]. All tests were performed using CPLEX version 10.0, on a Dell Dimension 8400 workstation equipped with a Pentium 4 processor at 3.60 GHz and 1.0 GB RAM. We used parameter $M = 1000$. Tables 1 and 2 summarize our results.

As Table 1 shows, both tree-based models perform comparatively well and neither dominates the other. However, in terms of computational complexity and scalability, the CDT model shows a slight advantage even at the present level (although 2^3, which identifies the complexity of the VDT model, is not a large number). Compared to Glen in terms of testing accuracy, our models perform better in the majority of cases. Furthermore, on average, each LOO test (meant as

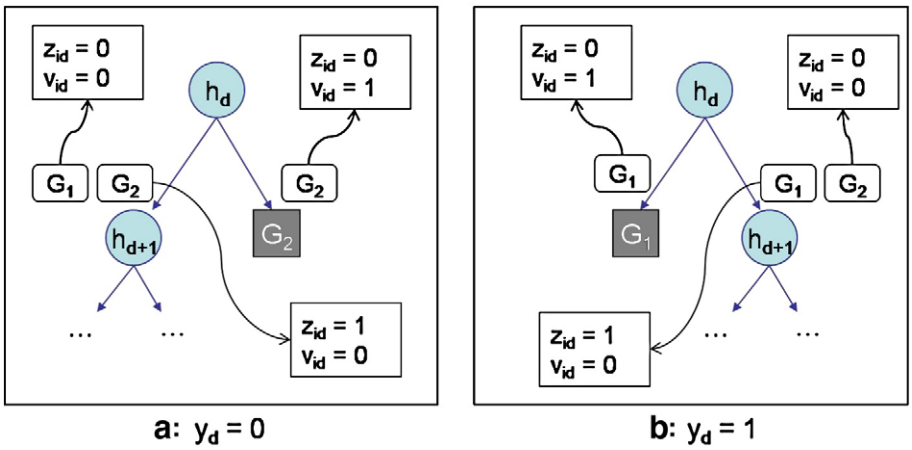

Fig. 4. Relation among y_d, z_{id} and v_{id}. If $y_d = 0$ (a), the tree grows to the left, then the objects of G_2 that fall on the right (into a leaf node associated with G_2) are correctly classified by the tree at depth d ($z_{id} = 0$, $v_{id} = 1$). Conversely, objects of both groups may fall on the left: those of G_1 have $z_{id} = 0$, $v_{id} = 0$, those of G_2 have $z_{id} = 1$, $v_{id} = 0$. If $y_d = 1$ (b), the situation is symmetric.

the set of all 100 runs, each composed by training and testing the classifier) takes 245.4 s for the VDT model and 100.9 s for the CDT model. Our implementation of Glen's piecewise-linear model takes an average of 228.3 s for each LOO test. Notably, our models obtain a higher accuracy than OC1 and SVM.

Next, we tested our models on another well-known data set: the Wisconsin Breast Cancer database, available online from the UCI data management repository (http://archive.ics.uci.edu/ml/ accessed on 5th December 2008). The data set consists of 683 patients screened for breast cancer (cases with missing values excluded), and 9 attributes per case. The class variable is binary, representing a benign or a malignant tumor. Since the domain and scale of all the descriptive attributes are the same, no standardization of the data was necessary.

As with the Japanese Banks, we ran various tests for different values of ε. It should be noted that for these tests, given the size of the data set, we decided to set a time limit of 120 s per holdout element test, at which time we recorded the best solution found. We found that, in most cases, the solution was either optimal (0 misclassifications in the training set) or very close to optimal (1 or 2 misclassifications). The trade-off between classification accuracy and the time to obtain an optimal solution greatly justified the use of a time limit. Table 3 summarizes the results for $\varepsilon = 0.00004$, 0.00005 and 0.00006. We ran experiments with different orders of magnitude for ε, but the results did not vary significantly from the ones reported here.

As Table 3 shows, the CDT model outperforms both the VDT model and Glen's model. It must be noted that our models can separate this data set by using three hyperplanes, while the piecewise-linear model, due to its convexity requirement, fails to do so. Both OC1 and SVM obtain a higher accuracy than CDT, but solving CDT takes less time (see Table 4) than OC1 and may find interpretable rules, unlike SVM. CDT is almost twice as fast as Glen, while the VDT model does not scale up efficiently to this large data set (average LOO test times were above 12,000 s). Although CDT is significantly faster than Glen, we believe that it would be computationally impractical to solve exactly our model in order to classify very large data sets (e.g. hundreds of attributes and millions of observations).

For both data sets, we performed a paired Student's t-test to assess whether there is a statistically significant difference between the accuracy obtained by our methods and the one obtained by Glen, for the same value of ε. The accuracy obtained by CDT is significantly higher (at $\alpha = 0.05$) than the one obtained by Glen (the p-value is equal to 0.0253 for the test on the Japanese Banks data set, and 0.004 for the one on the Breast Cancer data set). On the other hand, the accuracy obtained by VDT is not significantly higher than the one obtained by Glen. Therefore, we can conclude that CDT dominates Glen's approach.

In terms of the model complexity, Table 5 shows the number of binary variables, total variables, and constraints for each model. Glen's model is very complex, in that it uses a large number of binary variables, both in absolute terms and in comparison to the number of total variables. Although Glen's model uses fewer constraints than our models, its hyperplane constraints involve the use of two types of

Table 1
LOO hit rates for Japanese Banks.

ε	VDT	CDT	Glen	SVM	OC1
0.0005	84	84	86	87	80
0.0010	89	88	86		
0.0100	88	88	85		
0.0200	86	90	85		
0.0300	87	86	81		
0.0400	92	89	85		
0.0450	88	90	90		

Table 2
Time (in seconds) for each LOO test for the Japanese Banks data set.

ε	VDT	CDT	Glen	SVM	OC1
0.0005	201.0	7.5	126.9	10.2	181.9
0.0010	240.2	8.5	126.6		
0.0100	245.8	165.2	113.6		
0.0200	258.1	120.5	142.2		
0.0300	256.4	117.4	168.1		
0.0400	254.4	186.3	414.4		
0.0450	261.9	101.0	506.2		

Table 3
LOO hit rates for Breast Cancer data set.

ε	VDT	CDT	Glen	SVM	OC1
0.00004	80.4	92.8	84.6	97.07	96.19
0.00005	86.6	94.2	84.6		
0.00006	84.2	91.5	84.6		

Table 4
Average time (in seconds) for each LOO test for the Breast Cancer data set.

VDT	CDT	Glen	SVM	OC1
12,000	600	1200	86	1232.1

Table 5
Size of the optimization problems for Breast Cancer data.

Model characteristics	VDT	CDT	Glen
Binary variables	3412	2048	4396
Total variables	3442	4124	4453
Equality constraints	1	1	3
Total constraints	6139	6139	4574

binary variables (due to the convexity requirement) and twice the number of continuous variables.

5. Conclusions and future research

VDTs and CDTs constitute an innovation in the area of decision trees that has a broad application to important problems in real-world data mining. Our models produce particular tree structures that are constrained not to grow in width. The benefit of this constraint consists in the simplicity of the models and – in the case of the CDT – their straight-forward generalization to any maximum depth. Our results show that this apparent limitation does not affect the accuracy obtained.

Both models compare favorably in accuracy and speed to the main previous mixed integer Multi-hyperplane model. The improved speed promises to have valuable consequences for solving larger and more complex classification problems. Finally, the ability of our approach to operate without being confined by the convexity requirement of the piecewise-linear model yields additional advantages for dealing with problems where this requirement is inappropriate. On the other hand, our approach would not be suitable to classify very large data sets; furthermore it can handle only cross-sectional data, i.e. the observations in the data set are independent items that can be thought of as belonging to a sample taken at the same time. To perform a longitudinal analysis, where information relative to different times is available for each items, an appropriate design of the attributes is needed to represent the trend of the characteristics of each item through time.

The advances in modeling flexibility and in classification power made possible by our new formulations represent one more link in the chain of contributions originally stimulated by the goal programming model conception of [4]. The present formulations may in this sense be viewed as a way of generalizing and adapting the goal programming framework to include special forms of discrete conditions that enable it to apply more effectively to non-convex decision spaces in the context of classification analysis.

We envision several avenues for future research. First, our mixed integer models exhibit special structure that makes them susceptible to tailored accelerated and scalable solution methods. These tailored methods can take greater advantage of the linear programming relaxations of these models than customary procedures in today's state-of-the-art MIP software packages, by means of post-optimality analysis. In addition, the mechanisms underlying linear programming post-optimality analysis can also be used to generate new attributes for classification, to yield special non-linear and logic-based combinations of the original attributes. This provides a dynamic method for generating kernel functions and gives a new design for treating such functions that can supplement the usual procedures for creating kernel functions proposed in the SVM domain. The mutually reinforcing nature of these research avenues strongly motivates their exploration in future work as a natural outgrowth of our present study.

References

[1] P.L. Abad, W.J. Banks, New LP based heuristics for the classification problem, European Journal of Operational Research 67 (1993) 88–100.
[2] K.P. Bennett, J.A. Blue, A support vector machine approach to decision trees, in: J.A. Blue (Ed.), Proceedings of the IEEE World Congress on Computational Intelligence, Anchorage, AK, USA, 1998, pp. 2396–2401.
[3] A. Charnes, W.W. Cooper, Management models and industrial applications of linear programming, Wiley, New York, 1961.
[4] A. Charnes, W.W. Cooper, R.O Ferguson, Optimal estimation of executive compensation by linear programming, Management Science 1 (1955) 138–151.
[5] M. Chau, H. Chen, A machine learning approach to web page filtering using content and structure analysis, Decision Support Systems 44 (2008) 482–494.
[6] K. Cheung, J.T. Kwok, M. Law, K. Tsui, Mining customer product ratings for personalized marketing, Decision Support Systems 35 (2003) 231–243.
[7] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, Cambridge University Press, New York, 2000.
[8] A.J. Feelders, Statistical Concepts, Intelligent Data Analysis, Springer-Verlag New York, Inc., New York, 2003.
[9] N. Freed, F. Glover, Simple but powerful goal programming models for discriminant problems, European Journal of Operational Research 7 (1981) 44–60.
[10] J.J. Glen, Integer programming methods for normalisation and variable selection in mathematical programming discriminant analysis models, Journal of the Operational Research Society 50 (1999) 1043–1053.
[11] J.J. Glen, An iterative mixed integer programming method for classification accuracy maximizing discriminant analysis, Computers & Operations Research 30 (2003) 181–198.
[12] J.J. Glen, Mathematical programming models for piecewise-linear discriminant analysis, Journal of the Operational Research Society 56 (2004) 331–341.
[13] F. Glover, Improved linear programming models for discriminant analysis*, Decision Sciences 21 (1990) 771–785.
[14] F. Glover, G. Kochenberger, M. Better, Improved Classification and Discrimination by Successive Hyperplane and Multi-hyperplane Separation, Working Paper, University of Colorado, Boulder, Boulder, 2006.
[15] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers Inc., San Francisco, CA, 2000.
[16] H. Ince, T.B. Trafalis, A hybrid model for exchange rate prediction, Decision Support Systems 42 (2006) 1054–1062.
[17] X. Li, A scalable decision tree system and its application in pattern recognition and intrusion detection, Decision Support Systems 41 (2005) 112–130.
[18] S.K. Murthy, S. Kasif, S. Salzberg, A system for the induction of oblique decision trees, Journal of Artificial Intelligence Research 2 (1994) 1–32.
[19] Y. Peng, G. Kou, Y. Shi, Z. Chen, A multi-criteria convex quadratic programming model for credit data analysis, Decision Support Systems 44 (2008) 1016–1030.
[20] J.R. Quinlan, Induction of decision trees, Machine Learning 1 (1986) 81–106.
[21] L. Rokach, O. Maimon, Top-down induction of decision trees classifiers – a survey, IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews 35 (2005) 476–487.
[22] A. Stam, C.T. Ragsdale, On the classification gap in mathematical programming-based approaches to the discriminant problem, Naval Research Logistics 39 (1992) 545–559.
[23] T. Sueyoshi, Extended DEA-discriminant analysis, European Journal of Operational Research 131 (2001) 324–351.
[24] T. Sueyoshi, M. Goto, DEA-DA for bankruptcy-based performance assessment: misclassification analysis of Japanese Construction Industry, European Journal of Operational Research 199 (2009) 579–594.
[25] T. Sueyoshi, M. Goto, Methodological Comparison between DEA (Data Envelopment Analysis) and DEA-DA (Discriminant Analysis) from the perspective of bankruptcy assessment, European Journal of Operational Research, Accepted Manuscript 199 (2009) 561–575.
[26] M. Sun, M. Xiong, A mathematical programming approach for gene selection and tissue classification, Bioinformatics 19 (2003) 1243–1251.
[27] I.H. Witten, E. Frank, Data mining: practical machine learning tools and techniques, Morgan Kaufman, Boston, MA, 2005.

Dr. Marco Better is the Director of Custom Solutions of OptTek Systems, Inc. He obtained his Ph.D. in Operations Research from the Leeds School of Business of the University of Colorado at Boulder. He holds a B.S. in industrial engineering and an M.B.A. Dr. Better has over 15 years of professional work experience in the automobile, banking, and telecommunications industries, both in the US and in Latin America. His current interests lie in the application of optimization and data mining technology to solve complex problems in the industry.

Dr. Fred Glover is Chief Technology Officer for OptTek Systems, Inc., and a Distinguished Professor at the University of Colorado, Boulder. He has authored or co-authored more than 370 published articles and eight books in the fields of mathematical optimization, computer science and artificial intelligence. He is the recipient of the von Neumann Theory Prize, and is an elected member of the National Academy of Engineering. His numerous other awards include those from the American Association for the Advancement of Science (AAAS), the NATO Division of Scientific Affairs, the Institute of Operations Research and Management Science (INFORMS), the Decision Sciences Institute (DSI), the Energy Research Institute (ERI), the Institute of Cybernetics of the Ukrainian Academy of Science, and the Miller Institute for Basic Research in Science.

Michele Samorani is a PhD student at the Leeds School of Business of the University of Colorado at Boulder. His research interests include the application of operations research to solve data mining problems and the application of data mining techniques to enhance optimization procedures.