Working Paper

# An Efficient Mixed Integer Programming Algorithm for Minimizing the Training Sample Misclassification Cost in Two-group Classification

*A. Pedro Duarte Silva**
*Antonie Stam***

WP-96-93
August 1996

# An Efficient Mixed Integer Programming Algorithm for Minimizing the Training Sample Misclassification Cost in Two-group Classification

*A. Pedro Duarte Silva**
*Antonie Stam***

WP-96-93
August 1996

*Universidade Católica Portuguesa, Faculdade de Ciencias
Economicas e Empresariais, Centro Regional do Porto,
Rua Diogo Botelho 1327, 4150 Porto, Portugal

**Department of Management, Terry College of Business,
The University of Georgia, Athens, GA 30602, U.S.A.
and
International Institute for Applied Systems Analysis
Laxenburg, Austria

# AN EFFICIENT MIXED INTEGER PROGRAMMING ALGORITHM FOR MINIMIZING THE TRAINING SAMPLE MISCLASSIFICATION COST IN TWO-GROUP CLASSIFICATION

## ABSTRACT

In this paper, we introduce the Divide and Conquer (D&C) algorithm, a computationally efficient algorithm for determining classification rules which minimize the training sample misclassification cost in two-group classification. This classification rule can be derived using mixed integer programming (MIP) techniques. However, it is well-documented that the complexity of MIP-based classification problems grows exponentially as a function of the size of the training sample and the number of attributes describing the observations, requiring special-purpose algorithms to solve even small size problems within a reasonable computational time. The D&C algorithm derives its name from the fact that it relies, *a.o.*, on partitioning the problem in smaller, more easily handled sub-problems, rendering it substantially faster than previously proposed algorithms.

The D&C algorithm solves the problem to the *exact* optimal solution (*i.e.*, it is *not* a heuristic that approximates the solution), and allows for the analysis of much larger training samples than previous methods. For instance, our computational experiments indicate that, on average, the D&C algorithm solves problems with 2 attributes and 500 observations more than 3 times faster, and problems with 5 attributes and 100 observations over 50 times faster than Soltysik and Yarnold's software, which may be the fastest existing algorithm. We believe that the D&C algorithm contributes significantly to the field of classification analysis, because it substantially widens the array of data sets that can be analyzed meaningfully using methods which require MIP techniques, in particular methods which seek to minimize the misclassification cost in the training sample. The programs implementing the D&C algorithm are available from the authors upon request.

**Keywords:** Classification Analysis, Mixed Integer Programming, Nonparametric Statistics.

# AN EFFICIENT MIXED INTEGER PROGRAMMING ALGORITHM FOR MINIMIZING THE TRAINING SAMPLE MISCLASSIFICATION COST IN TWO-GROUP CLASSIFICATION

## 1. INTRODUCTION

The classification problem in discriminant analysis involves assigning observations to exactly one of several well-defined mutually exclusive, collectively exhaustive groups, based on their characteristics on a set of relevant attributes. Classification analysis has been used widely in many different disciplines, including medicine (Spiegelhalter and Knill-Jones 1984; Yarnold *et al.* 1994), the social sciences, psychology (Huberty 1984), finance (Eisenbeis 1977; Pinches and Mingo 1973), credit granting (Campbell and Dietrich 1983; Capon 1982; Srinivasan and Kim 1987), and strategic management (Kim and Kim 1985; Ramanujan *et al.*, 1986).

In this paper, we will limit ourselves to two-group classification. Define group $k$ by $G_k$, $k = 1$, 2, and denote the $p$-dimensional attribute vector describing the characteristics of observation $i$ by $\mathbf{a}_i = (a_{i1}, \ldots, a_{ip})^{\mathrm{T}}$. The classical statistical approach to solving the classification problem uses estimates of the prior group membership probabilities $\pi_k$ $(k = 1, 2)$ and the conditional probability density functions $\mathrm{p}(\mathbf{a}_i \mid G_k)$ to derive a classification rule which minimizes either the probability of misclassification or the expected misclassification cost. Another approach is to estimate the posterior probabilities $\mathrm{p}(G_k \mid \mathbf{a}_i)$ directly, and build a classification rule which weighs these probabilities by the applicable misclassification costs (Anderson 1972, McLachlan 1992).

A third approach is to pre-specify a functional form, and determine a classification rule that optimizes some measure of discrimination or classification accuracy in the training sample. Fisher's linear discriminant function (LDF) (Fisher 1936) and Smith's quadratic discriminant function (QDF) (Smith 1947) are the most widely known methods in this class. The LDF rule has been shown to be optimal if the attribute populations are multivariate normally distributed with equal cross-group covariances, whereas the QDF is optimal under multivariate normality with covariances that are unequal across groups (Anderson 1984).

Both the LDF and QDF use criteria based on $\mathrm{L}_2$-norm distance measures. However, an $\mathrm{L}_2$-norm criterion may not be appropriate for non-normal data conditions. Although researchers have found that the classification accuracy of the LDF is fairly robust if the normality assumption is moderately violated, it tends to classify poorly if the deviations from normality are significant (Lachenbruch *et al.* 1973; Fatti *et al.* 1982). It is well-known that criteria based on higher order norm distances tend to be influenced heavily by extreme training sample observations. Real-life data sets often have highly skewed or heavy-tailed attribute distributions and are frequently contaminated by outliers (Eisenbeis 1977; Glorfeld and Kattan 1989; Stam and Ragsdale 1992). In the presence of such data conditions, it may be useful to consider nonparametric classification methods based on absolute distances ($\mathrm{L}_1$-norm methods) or methods which minimize misclassification costs ($\mathrm{L}_0$-norm methods), which, like the LDF and QDF, pre-specify the functional form of the classification rule but are potentially more robust.

McLachlan (1992, p. 16) remarks that classification accuracy depends mostly on how well the classification rule can handle observations of doubtful origin, rather than on how it deals with observations of obvious origin. In other words, what matters above all is the classification performance in the region of the attribute space where the groups overlap. As mathematical programming (MP)-based methods do not make any assumptions about the distributional characteristics of the attribute populations, and focus on the boundaries of the groups where overlap occurs and group membership is most uncertain, McLachlan's argument supports the use of MP classification methods.

For two-group classification, Freed and Glover (1981b) popularized the $L_1$-norm minimize the sum of absolute deviations (MSD) method, variants of which had been proposed previously by Mangasarian (1965), Koford and Groner (1966), Smith (1968) and Hand (1981). Freed and Glover (1981a) proposed the $L_\infty$-norm-based minimize the maximum deviation (MMD) method, which almost all studies have found to yield poor classification results (Bajgier and Hill 1982; Freed and Glover 1986; Joachimsthaler and Stam 1990; Mahmood and Lawrence 1987; Markowski and Markowski 1987). There is some empirical and experimental evidence that under certain non-normal data conditions the MSD is more accurate than the LDF and QDF (Duarte Silva 1995; Freed and Glover 1986; Joachimsthaler and Stam 1988, 1990; Srinivasan and Kim 1987). Other $L_1$-norm two-group classification methods include the optimize the sum of deviations (OSD) method (Bajgier and Hill 1982; Markowski and Markowski 1985) and the Hybrid method (Glover, Keene and Duea 1988). The $L_1$- and $L_\infty$-norm methods can be solved using linear programming (LP) techniques. Stam and Joachimsthaler (1989) proposed a class of general $L_p$-norm criteria $(1 \leq p < \infty)$ which require nonlinear programming (NLP) techniques. Recently, Gochet *et al.* (1993) have generalized the LP methodology for the two-group classification problem to the case of multiple groups.

Another MP approach, the mixed integer programming (MIP) method, minimizes the number of misclassified training sample observations directly. Named after the MP optimization technique which is often used to solve this formulation, the MIP method can be viewed as an $L_p$-norm method with $p \rightarrow 0$. Hence, we will refer to these methods as MP-$L_0$ methods. With appropriate weighting factors in the objective function, the MP-$L_0$ method minimizes the misclassification costs in the training sample. The MP-$L_0$ method for two-group classification was proposed by Ibaraki and Muroga (1970), Warmack and Gonzalez (1973), Liitschwager and Wang (1978), and popularized by Bajgier and Hill (1982), Asparoukhov (1985), Koehler and Erenguc (1990), Stam and Joachimsthaler (1990) and Stam and Jones (1990). Gehrlein (1986) introduced a general formulation for the multiple-group case.

Unfortunately, the MP-$L_0$ formulation is NP-complete, with computational requirements which increase exponentially as a function of the training sample size. As a consequence, standard MIP solvers such as MPSX-MIP (International Business Machines 1975) or LINDO (Schrage 1991) can be used only to solve MP-$L_0$ problems with relatively small training samples. Although special-purpose algorithms which take advantage of the special structure and characteristics of the problem formulation alleviate this problem somewhat and facilitate the analysis of larger problems (Banks and Abad 1991; Koehler and Erenguc 1990; Rubin 1990; Soltysik and Yarnold 1994), their application is still limited to training samples of up to several hundred observations.

The initially proposed MP-based classification rules were linear in the original attributes. Several recent studies have shown that under certain conditions, for instance if the variance-covariances across groups differ substantially, classification rules which are nonlinear in the original attributes, in particular quadratic and polynomial ones, can yield much more accurate classification results than their linear counterparts (Banks and Abad 1994; Duarte Silva and Stam 1994; Rubin 1994). In practice, the optimal classification rule may be approximated most accurately by general polynomial functions with a large number of parameters, and using training samples which are "large" relative to the number of parameters. Thus, accurate estimation in the presence of unequal variance-covariances across groups may call for the use of large training samples. In this paper, we present the Divide and Conquer (D&C) algorithm, a special-purpose, computationally efficient algorithm which takes advantage of the special structure of the MP-$L_0$ problem, enabling an MIP analysis for substantially larger training samples than previous special-purpose algorithms. As its computational requirements grow at a slower rate than previous methods, the relative efficiency of the D&C algorithm increases with the number of coefficients to be estimated, $t$, and the training sample size, $n$. The computational results in Section 5 indicate that the D&C algorithm is superior to previous methods for $n \geq 200$ if $t = 2$, for $n \geq 100$ if $t = 3$, and for $n \geq 50$ if $t \geq 4$.

Throughout this paper, we will focus on the MIP method for classification analysis, and will use the pre-specified MP classification rule in (1.1), which is comprised of $t$ functions, $\xi_{i1} = \mathrm{f}^1(\mathbf{a}_i)$, ..., $\xi_{it} = \mathrm{f}^t(\mathbf{a}_i)$, of the original attribute vectors $\mathbf{a}_i$:

$$\left.\begin{array}{l} \text{Classify observation } i \text{ into } G_1, \text{ if } \sum_{j=1}^{t} c_j\xi_{ij} < c_0, \\ \text{Classify observation } i \text{ into } G_2, \text{ if } \sum_{j=1}^{t} c_j\xi_{ij} > c_0, \end{array}\right\} \tag{1.1}$$

where the $c_j$ ($j = 0, ..., t$) are unknown coefficients (parameters), determined such that the appropriate classification criterion is optimized, and observation $i$ is not classified if $\sum_{j=1}^{t} c_j\xi_{ij} = c_0$. The functions $\mathrm{f}^j(\mathbf{a}_i)$ may be nonlinear. The linear classification rule is a special case of (1.1), with $t = p$ and $\xi_{ij} = a_{ij}$, for all $i$, $j$.

Let the classification score of observation $i$ be given by $f_i = \sum_{j=1}^{t} c_j\xi_{ij}$, and denote the set of misclassified observations by $\mathcal{M}$. The surface defined by $f_i = c_0$ separates the two groups, and $d_i = |f_i - c_0|$ measures the distance of observation $i$ from the separating surface. If $i \in \mathcal{M}$, $d_i$ measures the extent of misclassification, otherwise $d_i$ measures the extent of correct classification. Assuming equal costs of misclassification, the MSD criterion minimizes $\sum_{i \in \mathcal{M}} d_i$, the MMD criterion minimizes $\max_{i \in \mathcal{M}} d_i$, and the MIP criterion minimizes $\sum_i \delta_i$, where $\delta_i = 1$ if $i \in \mathcal{M}$ and $\delta_i = 0$ otherwise.

## 2. PREVIOUS MP-$L_0$ METHODS FOR MINIMIZING MISCLASSIFICATION COSTS

The critical factors affecting the computational effort of solving MIP models are the number of integer variables, the structure of the model, and the number of constraints (Hillier and Lieberman 1990, p. 467). Specifically, MIP models with a structure characterized by "tight" constraints, *i.e.*, models for which the convex hull of the feasible region is close to the feasible region of the linear relaxation of the MIP model, require considerably less computational effort than models with "loose" constraints.

The first MP-$L_0$ formulation is due to Ibaraki and Muroga (1970). For a training sample of $n$ observations, Ibaraki and Muroga's formulation requires $n$ binary variables and $3n$ constraints, as well as the specification of an arbitrarily large scalar $M$, which may lead to loose constraints. The computational requirements of Ibaraki and Muroga's model quickly become prohibitive as $n$ increases. Warmack and Gonzalez (1973) developed a special-purpose $L_0$ classification algorithm which uses a non-enumerative search procedure based on the geometrical properties of the problem, and is not based on MP models. Liitschwager and Wang (1978) proposed a MP-$L_0$ formulation requiring $n+2t$ integer variables, $n+1$ constraints and $2t$ simple bounds, which does not involve an arbitrary large scalar and has tighter constraints than Ibaraki and Muroga's model. Koehler and Erenguc (1990) developed a model with $n$ integer variables that uses "large" scalars, which may lead to loose constraints. Their special-purpose MP-$L_0$ algorithm solves successive LP models with no more than $t+1$ constraints. Banks and Abad (1991) used a similar strategy, solving LP models with $t+1$ tighter constraints and without arbitrary large scalars. Their model formulation has $n$ integer variables. Athough Banks and Abad's model requires $n$ more non-integer variables than that of Koehler and Erenguc, the tighter model constraints more than compensate for the additional computational effort. Soltysik and Yarnold (1993, 1994) showed that their modified version of the non-MP based Warmack and Gonzalez algorithm is still more efficient, computationally, than any of the MP-based algorithms mentioned above.

MP-$L_0$ formulations with secondary objectives can be found in Ibaraki and Muroga (1970), Bajgier and Hill (1982) and Rubin (1990). Soltysik and Yarnold (1992) present alternative ways of tightening the model constraints. Heuristic procedures for solving MP-$L_0$ models can be found in Rubin (1990), Koehler and Erenguc (1990), Ragsdale and Stam (1991), Abad and Banks (1993) and Chen (1996).

## 3. SOME IMPROVEMENTS ON MP-$L_0$ METHOD ALGORITHMS

The purpose of our research is to develop faster algorithms for solving MP-$L_0$ models, thus enabling the analyst to analyze larger size training samples. In this paper, we introduce the D&C algorithm, an MP-$L_0$ method which, like Koehler and Erenguc (1990) and Banks and Abad (1991), replaces the task of solving the original MIP formulation by that of solving several LP models with fewer constraints. In addition, the D&C algorithm partitions the parameter space $\mathfrak{C}$ into sub-spaces, and solves each corresponding sub-problem with tighter constraints separately.

The contribution of the D&C algorithm can be divided into (1) improvements on Liitschwager and Wang's (1978) formulation of the MP-$L_0$ classification problem, which we will discuss in Section 3.1, and (2) special-purpose algorithms which partition the original problem into more easily solved sub-problems, which will be the topic of Section 3.2.

### 3.1. Tightening the Model Constraints

Liitschwager and Wang's (1978) formulation is presented in Model I.

**Model I**

$$\text{Minimize } z_1 = \frac{\pi_1 C(2\,|\,1)}{n_1} \sum_{i \in G_1} r_i + \frac{\pi_2 C(1\,|\,2)}{n_2} \sum_{i \in G_2} r_i, \tag{3.1}$$

Subject to:

$$\sum_{j=1}^{p} c_j a_{ij} - Mr_i \le c_0, \quad i \in G_1, \tag{3.2}$$

$$\sum_{j=1}^{p} c_j a_{ij} + Mr_i \ge c_0, \quad i \in G_2, \tag{3.3}$$

$$-1 + 2e_j \le c_j \le 1 - 2g_j, \quad j = 1, \ldots, p, \tag{3.4}$$

$$\sum_{j=1}^{p} (e_j + g_j) = 1, \tag{3.5}$$

$$M = 2p\text{Max}_{i,j}(\,|\,a_{ij}\,|\,), \tag{3.6}$$

$$c_j \text{ unrestricted}, \; j = 0, \ldots, p, \tag{3.7}$$

$$e_j,\, g_j,\, r_q \in \{0, 1\}; \; j = 1, \ldots, p; \; q = 1, \ldots, n_1 + n_2, \tag{3.8}$$

where $C(k\,|\,m)$ represents the cost of classifying an observation that belongs to $G_m$ into $G_k$ ($k$, $m = 1$, 2; $k \ne m$), and $n_k$ is the number of training sample observations belonging to $G_k$ ($k = 1$, 2), for a total training sample size of $n = n_1 + n_2$.

Model I assumes a linear classification rule, and criterion $z_1$ in (3.1) represents the per unit misclassification cost in the training sample. The scalar $M$ should be large enough to ensure that (3.2) and (3.3) are always satisfied, so that $r_i = 1$ if observation $i \in \mathcal{M}$, and $r_i = 0$ if $i \notin \mathcal{M}$. By (3.5), exactly one of the $e_j$, $g_j$, $j = 1, \ldots, p$, equals 1. The constraint set (3.4) forces exactly one of the $c_j$ ($j = 1, \ldots, p$), say $c_h$, to either $-1$ (if $g_h = 1$) or $+1$ (if $e_h = 1$), and restricts all other $c_m$ ($m = 1, \ldots, p$; $m \ne h$) to $|\,c_m\,| \le 1$. Forcing $|\,c_h\,| = 1$ eliminates the trivial solution $c_0 = c_1 = \ldots = c_p = 0$ from consideration. The scaling of the $c_j$ does not exclude any classification rule from consideration, because all proportional rules of the form (1.1) are equivalent. Scaling the problem such that $|\,c_j\,| \le 1$, $j = 1$, $\ldots$, $p$, guarantees that in the optimal solution $|\,c_0\,| \le p\text{Max}_{ij}(\,|\,a_{ij}\,|\,)$, so that (3.2) and (3.3) are always satisfied if $r_i = 1$ and $M = 2p\text{Max}_{ij}(\,|\,a_{ij}\,|\,)$. In the remainder of this paper, we denote the parameter space of all $c_j$ except $c_h$ by $\mathcal{S}$.

Even without special-purpose algorithms, the D&C algorithm improves on Model I in several different ways. First, the D&C algorithm generalizes the linear classification rules in Model I to the form (1.1). Second, as the minimization of $z_1$ may yield several non-equivalent classification rules with the same minimum training sample misclassification cost, the D&C algorithm includes a secondary criterion $z_2$ which serves to resolve ties in the achievement of $z_1$:

$$\text{Minimize } z_2 = \frac{\pi_1 C(2\mid 1)}{n_1} \sum_{i \in G_1} \left\{ \sum_{j=1}^{t} c_j \xi_{ij} - c_0 \right\} + \frac{\pi_2 C(1\mid 2)}{n_2} \sum_{i \in G_2} \left\{ c_0 - \sum_{j=1}^{t} c_j \xi_{ij} \right\}. \tag{3.9}$$

The secondary criterion $z_2$ measures the extent by which the observations are misclassified minus the extent by which the observations are classified correctly. As $z_2$ should never affect the achievement of $z_1$, $z_1$ and $z_2$ of the modified objective function are optimized lexicographically.

Third, the model structure can be improved by imposing individual lower ($L_{c_j} < 0$) and upper ($U_{c_j} > 0$) bounds on each $c_j$ ($j = 0, \ldots, t$), instead of the bounds of $-1$ and $+1$ used in Model I. If $z_1$ is the single criterion, the optimal criterion value remains unchanged by imposing individual bounds, provided that $L_{c_j}$ and $U_{c_j}$ have opposite signs, because any classification rule can be converted to an equivalent rule with bounds $L_{c_j} < 0$, $U_{c_j} > 0$, by multiplying all $c_j$ by an appropriate scalar. However, if $z_2$ is introduced into the model, the choice of $L_{c_j}$ and $U_{c_j}$ is no longer arbitrary, because the scaling of $c_j$ affects the type of solution obtained directly. For instance, if $z_2 < 0$, which will be the case for most "reasonable" classification rules (Glover, Keene and Duea 1988; Glover 1990; Gochet *et al.* 1993), minimizing $z_2$ implies maximizing $|z_2|$, introducing a bias towards rules with $c_j$ that are close to their bounds. Thus, the choice of $L_{c_j}$ and $U_{c_j}$ should reflect reasonable tentative values for $c_j$, for instance values derived using some other classification method. In the current implementation of the D&C algorithm, $L_{c_j}$ and $U_{c_j}$ are selected symmetric about 0, one being the value $A$ estimated by the LDF (if the MP-L$_0$ rule is linear in the original attributes) or the QDF (if the MP-L$_0$ rule is quadratic in the original attributes), and the other being $-A$. The bounds are then normalized such that $|L_{c_0}| = |U_{c_0}| = 1$.

Fourth, the D&C algorithm implements the recommendation by Soltysik and Yarnold (1992) to replace $M$ in (3.2) and (3.3) by observation-specific scalars $M_i$ ($i = 1, \ldots, n_1 + n_2$). As $L_{c_j} \le c_j \le U_{c_j}$, we can set $M_i = \sum_{j=1}^{t} \text{Max}\left\{L_{c_j}\xi_{ij}, U_{c_j}\xi_{ij}\right\} - L_{c_0}$ for each $i \in G_1$, without affecting the feasibility of (3.2). Similarly, for $i \in G_2$ we can set $M_i = U_{c_0} - \sum_{j=1}^{t} \text{Min}\left\{L_{c_j}\xi_{ij}, U_{c_j}\xi_{ij}\right\}$, without affecting (3.3).

Fifth, we observe that a branch-and-bound algorithm (Hillier and Lieberman 1990, 469–485) which branches first on the binary variables $e_j$ and $g_j$ in Model I, will set each of these variables equal to 1 in turn. However, this is equivalent to omitting the $e_j$ and $g_j$ from Model I altogether, and setting each $c_j$ to $-1$ (or to $L_{c_j}$) and $+1$ (or to $U_{c_j}$) in turn. This solution strategy, adopted in the D&C algorithm, reduces the number of binary variables by $2(t+1)$ and involves solving $2(t+1)$ separate problems P($l$), $l = 1, \ldots, 2(t+1)$, as (1.1) has $t+1$ different coefficients $c_j$ ($c_0, \ldots, c_t$).

Summarizing, the D&C algorithm solves $2(t+1)$ problems of the form Model II in (3.10)–(3.19):

**Model II: MP-L$_0$ Major Sub-Problem**

$$\text{Minimize } z_3 = \frac{\pi_1 C(2\,|\,1)}{n_1}\sum_{i\in G_1}(r_i-\eta s_i) + \frac{\pi_2 C(1\,|\,2)}{n_2}\sum_{i\in G_2}(r_i-\eta s_i), \tag{3.10}$$

Subject to:

$$\sum_{j=0}^{t} c_j\xi_{ij}-M_i r_i+s_i=0,\ \ i\in G_1, \tag{3.11}$$

$$\sum_{j=0}^{t} c_j\xi_{ij}+M_i r_i-s_i=0,\ \ i\in G_2, \tag{3.12}$$

$$L_{c_j}\le c_j\le U_{c_j},\ \ j=0,\ \dots,\ t,\ t\ne h, \tag{3.13}$$

$$c_h=B_{c_h}, \tag{3.14}$$

$$M_i=\sum_{j=0,j\ne h}^{t}\text{Max}\left\{L_{c_j}\xi_{ij},\ U_{c_j}\xi_{ij}\right\}+B_{c_h}\xi_{ih},\ i\in G_1, \tag{3.15}$$

$$M_i=-\sum_{j=0,j\ne h}^{t}\text{Min}\left\{L_{c_j}\xi_{ij},\ U_{c_j}\xi_{ij}\right\}+B_{c_h}\xi_{ih},\ i\in G_2, \tag{3.16}$$

$$c_j \text{ unrestricted},\ j=0,\ \dots,\ t;\ j\ne h, \tag{3.17}$$

$$r_i\in\{0,\ 1\},\ \forall\ i\in G_1\cup G_2, \tag{3.18}$$

$$s_i\ge 0,\ \forall\ i\in G_1\cup G_2, \tag{3.19}$$

where $\boldsymbol{\xi}_i=(\xi_{i0},\ \dots,\ \xi_{it})^{\mathrm{T}}=[-1,\ \mathrm{f}^1(\mathbf{a}_i),\ \dots,\ \mathrm{f}^t(\mathbf{a}_i)]^{\mathrm{T}}$, $c_h$ is fixed to $B_{c_h}$ (either $L_{c_h}$ or $U_{c_h}$), the $s_i$ are slack ($i\in G_1$) or surplus ($i\in G_2$) variables, $\mathbf{s}=(s_1,\ \dots,\ s_{n_1+n_2})^{\mathrm{T}}$, and $\eta$ is a constant satisfying (3.20),

$$\eta<\text{Min}\left(\frac{1}{M_i},\ \frac{\text{Min}\left(\frac{\pi_1 C(2\,|\,1)}{n_1},\ \frac{\pi_2 C(1\,|\,2)}{n_2}\right)}{\frac{\pi_1 C(2\,|\,1)}{n_1}\sum_{i\in G_1}U_{s_i}+\frac{\pi_2 C(1\,|\,2)}{n_2}\sum_{i\in G_2}U_{s_i}}\right). \tag{3.20}$$

The $U_{s_i}$ in (3.20) represent the upper bounds of $s_i$, which, using (3.11)–(3.14), can be expressed as in (3.21) and (3.22),

$$U_{s_i}=M_i-\sum_{j=0,j\ne h}^{t}\text{Min}\left\{L_{c_j}\xi_{ij},\ U_{c_j}\xi_{ij}\right\}-B_{c_h}\xi_{ih},\ i\in G_1, \tag{3.21}$$

$$U_{s_i}=M_i+\sum_{j=0,j\ne h}^{t}\text{Max}\left\{L_{c_j}\xi_{ij},\ U_{c_j}\xi_{ij}\right\}+B_{c_h}\xi_{ih},\ i\in G_2. \tag{3.22}$$

Since $s_i\le U_{s_i}$, $\forall i\in G_1\cup G_2$, (3.20) implies (3.23),

$$\left|-\eta\left(\frac{\pi_1 C(2\,|\,1)}{n_1}\sum_{i\in G_1}s_i+\frac{\pi_2 C(1\,|\,2)}{n_2}\sum_{i\in G_2}s_i\right)\right|<\text{Min}\left(\frac{\pi_1 C(2\,|\,1)}{n_1},\ \frac{\pi_2 C(1\,|\,2)}{n_2}\right), \tag{3.23}$$

so that the $r_i$ ($i=1,\ \dots,\ n_1+n_2$) in the optimal solution are affected only by the minimization of those components of $z_3$ which correspond to $z_1$. By substitution, the left-hand side of (3.23) equals (3.24),

$$\left|\eta\left(\frac{\pi_1 C(2\,|\,1)}{n_1}\sum_{i\in G_1}\left\{\sum_{j=0}^{t} c_j\xi_{ij}-M_i r_i\right\}+\frac{\pi_2 C(1\,|\,2)}{n_2}\sum_{i\in G_2}\left\{-\sum_{j=0}^{t} c_j\xi_{ij}-M_i r_i\right\}\right)\right|. \tag{3.24}$$

Therefore, once the $r_i$'s are fixed (3.24) is a linear transformation of $z_2$ in (3.9), and minimizing $z_3$ is equivalent to minimizing $z_1$, followed by minimizing $z_2$ as a secondary objective to resolve the case of alternative optimal solutions. In a preliminary experimental comparison (not reported here) we found that, due mainly to a tighter model structure, solving the $2(t+1)$ Model II problems requires substantially less computational effort than solving the corresponding Model I problem.

## 3.2. Partitioning the Global Model

The major contribution of the D&C algorithm is that it greatly improves solution efficiency by dividing $\mathcal{C}$ into several sub-spaces, and solving the resulting partial models separately. Before discussing the partitioning strategy of the D&C algorithm, we turn our attention to the special case of perfectly separated training samples.

### 3.2.1. The Case of Perfect Separation

If it is possible to determine a rule for which all training sample observations are classified correctly, we can obtain perfect separation of the groups in the training sample. In this case, it is not necessary to divide the global model into partial models, and the optimal solution of Model II, with secondary criterion $z_2$, can be found by solving Model III in (3.25)–(3.30). Therefore, the D&C algorithm solves Model III prior to creating $2(t+1)$ Model II problems.

**Model III: Perfect Separation Model**

$$\text{Minimize } z_4 = -\frac{\pi_1 C(2\,|\,1)}{n_1}\sum_{i\in G_1} s_i - \frac{\pi_2 C(1\,|\,2)}{n_2}\sum_{i\in G_2} s_i, \tag{3.25}$$

Subject to:

$$\sum_{j=0}^{t} c_j\xi_{ij}+s_i=0,\quad i\in G_1, \tag{3.26}$$

$$\sum_{j=0}^{t} c_j\xi_{ij}-s_i=0,\quad i\in G_2, \tag{3.27}$$

$$L_{c_j}\le c_j\le U_{c_j},\quad j=0,\ldots,t, \tag{3.28}$$

$$c_j \text{ unrestricted},\ j=0,\ldots,t, \tag{3.29}$$

$$s_i\ge 0,\ \forall\ i\in G_1\cup G_2. \tag{3.30}$$

Model III has $t+1$ structural variables, $n_1$ slack variables, $n_2$ surplus variables, $n_1+n_2$ constraints and $2(t+1)$ simple bounds. The D&C algorithm solves the dual of Model III, which has substantially less constraints (excluding simple bounds), using the simplex method for bounded variables (Hillier and Lieberman 1990, pp. 58–111, 302–304). Perfect separation is possible if the optimal solution is finite. If the dual yields an unbounded solution, perfect separation in the training sample is impossible, and the D&C algorithm proceeds with partitioning the global model.

### 3.2.2. Global Model Partitioning Strategy

As noted above, after checking whether the training sample is perfectly separable, the first step in the D&C algorithm is to divide the original MP-$L_0$ problem into $2(t+1)$ major Model II sub-problems $P(l)$ $(l = 1, ..., 2(t+1))$, in which all $c_j$ are treated as variables, except for $c_h$, which is fixed at one of its bounds. Let $\mathcal{C}(l) \subset \Re^{t+1}$ be the parameter space of all $c_j$ associated with $P(l)$, and let $\mathcal{S}(l) \subset \Re^t$ be the parameter space of the $c_j$ $(j = 0, ..., t;\ j \neq h)$ of $P(l)$ that are not fixed.

The second step is to divide $\mathcal{S}(l)$ into $r$ sub-spaces $\mathcal{S}_1(l), ..., \mathcal{S}_r(l)$, with "tight" constraints. If a sub-space does not contain solutions that are close to the global optimal solution, the computational loss due to a loose formulation tends to be relatively mild. However, it is important to have very tight formulations for sub-spaces which contain solutions that are close to the global optimum. Thus, rather than partitioning $\mathcal{S}(l)$ into sub-spaces of equal size, the D&C algorithm creates larger sub-spaces in regions with solutions that have relatively high training sample misclassification costs and smaller ones in regions with solutions for which the misclassification costs tend to be lower. The D&C algorithm generates a set of reasonably good solutions $\mathcal{B}(l)$, and uses the characteristics of these solutions in order to partition $\mathcal{S}(l)$ effectively. $\mathcal{B}(l)$ is comprised of the the $\alpha N$ best among $N$ sample solutions $(0 < \alpha < 1)$ in $\mathcal{S}(l)$, determined using a limited breadth-first branch-and-bound search strategy. The values of $N$ and $\alpha$ are selected by the analyst.

$\mathcal{S}(l)$ is partitioned into rectangular regions which are parallel to the principal axes of variation of $\mathcal{B}(l)$. The principal axes of variation are determined using principal component analysis (Morrison 1990, pp. 312–342). The initial sub-space $\mathcal{S}_1(l)$, built around the centroid of $\mathcal{B}(l)$, is a $t$-dimensional square region with sides of length $SZO(l)$. The volume of $\mathcal{S}_1(l)$ should be a function of the anticipated effort required to solve each problem. One important factor affecting this effort is the number of misclassified observations in the optimal solution. The D&C algorithm uses the number of misclassifications in the incumbent solution (*i.e.*, the best solution found so far) as a proxy of the number of misclassifications in the optimal solution, and $SZO(l)$ is inversely related to this quantity. $SZO(l)$ is also inversely related to the difference between the objective value of the incumbent solution and the objective value of the linear relaxation of the sub-problem $P(l)$ under consideration. The rationale for the latter is that if these objective function values are similar (close), the branch-and-bound algorithm will be able to quickly fathom most of the nodes of its search tree, in which case it is not necessary to have a very tight formulation and a larger value for $SZO_1(l)$ suffices.

As we move away from the centroid along the principal axes of $\mathcal{B}(l)$, the side length along the $k^{th}$ dimension $SZ_k(l)$ of the subspaces $\mathcal{S}_a(l)$, $a = 2, ..., r$, is increased by a factor $IFT_k(l) > 1$, $k = 1, ...,$ $t$, which is is inversely related to the contribution of the $k^{th}$ principal component of $\mathcal{B}(l)$ to the total variance of $\mathcal{B}(l)$, as measured by the corresponding eigenvalue $v_k(l)$. Proceeding in this way, the volume of the sub-spaces $\mathcal{Y}_a(l)$ increases at successively higher rates, as we move from $\mathcal{Y}_1(l)$ along directions which contribute less to the total variance of $\mathcal{B}(l)$. Figure 1 illustrates a typical partition of $\mathcal{Y}(l)$ for $t = 2$.

---

Figure 1 About Here

---

### 3.2.3 Solution Strategy

Let $\mathcal{Y}_a(l)$ be some sub-space created by partitioning $\mathcal{Y}(l)$. The partial sub-problem of Model II restricted to $\mathcal{Y}_a(l)$ requires the introduction of a set of $t$ simple bounds in order to restrict the non-fixed coefficients $c_j$ $(j = 0, \ldots, t;\ j \neq h)$ to $\mathcal{Y}_a(l)$. This is done by applying the change of variable transformation $(y_1, \ldots, y_{t+1})^{\mathrm{T}} = \mathbf{y} = \mathbf{Uc}$, where $\mathbf{U}$ is a $(t+1) \times (t+1)$ matrix, such that $u_{ij}$ $(i \neq t+1,\ j \neq h)$ is the coefficient of $c_j$ on the $i^{th}$ principal component of $\mathcal{B}(l)$, $\mathbf{u}_h = (0, \ldots, 0, 1)^{\mathrm{T}}$, and $u_{t+1,j} = 0$, for $j \neq h$. The first $t$ elements of $\mathbf{y}$ represent the principal axes of variation of $\mathcal{B}(l)$, and $y_{t+1} = c_h$. The partial sub-problem for $\mathcal{Y}_a(l)$ is given as Model IV.

**Model IV: Partial Sub-Problem**

$$\text{Minimize } z_5 = \frac{\pi_1 C(2\,|\,1)}{n_1} \sum_{i \in G_1} (r_i - \eta s_i) + \frac{\pi_2 C(1\,|\,2)}{n_2} \sum_{i \in G_2} (r_i - \eta s_i), \tag{3.31}$$

Subject to:

$$\sum_{j=1}^{t} y_j \beta_{ij} + B_{y_{t+1}} \beta_{i,t+1} + s_i = M_i r_i, \quad i \in G_1 \cup G_2, \tag{3.32}$$

$$L_{c_j} \leq \sum_{k=1}^{t} u^{-1}_{j+1,k} y_k \leq U_{c_j}, \; j = 0, \ldots, t,\ j \neq h, \tag{3.33}$$

$$L_{y_j} \leq y_j \leq U_{y_j}, \quad j = 1, \ldots, t, \tag{3.34}$$

$$B_{y_{t+1}} = B_{c_h}, \tag{3.35}$$

$$y_j \text{ unrestricted}, \; j = 1, \ldots, t, \tag{3.36}$$

$$r_i \in \{0, 1\}, \; \forall\, i \in G_1 \cup G_2, \tag{3.37}$$

$$s_i \geq 0, \; \forall\, i \in G_1 \cup G_2, \tag{3.38}$$

Constraint set (3.32) is derived from (3.11) and (3.12). Specifically, $\boldsymbol{\beta}_i = (\beta_{i1}, \ldots, \beta_{i,t+1})^{\mathrm{T}} = \mathbf{U}^{-1}\boldsymbol{\xi}_i$ for $i \in G_1$, and $\boldsymbol{\beta}_i = -\mathbf{U}^{-1}\boldsymbol{\xi}_i$ for $i \in G_2$. Constraint set (3.33) corresponds directly with (3.13). If $\mathcal{Y}_a(l)$ is located within the interior of $\mathcal{Y}(l)$, (3.33) is never binding. Constraint set (3.34) restricts each $y_j$ to the region of $\mathcal{Y}_a(l)$. Algorithm IV details how $L_{y_j}$ and $U_{y_j}$ can be computed. The values of the $M_i$ are defined by (3.39)–(3.42). Whereas (3.39) and (3.40) correspond with the original restrictions on the $c_j$, (3.41) reflects that $\mathbf{c} = (c_0, \ldots, c_t)^{\mathrm{T}}$ must lie inside $\mathcal{Y}_a(l)$.

$$M_i^1 = \sum_{j=0,\, j \neq h}^{t} \operatorname{Max}\left\{L_{c_j}\xi_{ij},\ U_{c_j}\xi_{ij}\right\} + B_{c_h}\xi_{ih}, \; i \in G_1, \tag{3.39}$$

$$M_i^1 = -\Big(\sum_{j=0, j\neq h}^{t} \operatorname{Min}\Big\{L_{c_j}\xi_{ij},\ U_{c_j}\xi_{ij}\Big\} + B_{c_h}\xi_{ih}\Big),\ i \in G_2, \tag{3.40}$$

$$M_i^2 = \sum_{j=1}^{t} \operatorname{Max}\Big\{L_{y_j}\beta_{ij},\ U_{y_j}\beta_{ij}\Big\} + B_{y_{t+1}}\beta_{i,t+1},\ i \in G_1 \cup G_2, \tag{3.41}$$

$$M_i = \operatorname{Min}(M_i^1,\ M_i^2),\ i \in G_1 \cup G_2, \tag{3.42}$$

Model IV has $t$ continuous structural variables, $n_1 + n_2$ slack variables, $n_1 + n_2$ binary variables, $n_1 + n_2 + 2t$ constraints and $2t$ simple bounds. Solving Model IV using the branch-and-bound algorithm implies solving successive LP models (linear relaxations of MIP models) in which some of the $r_i$ are fixed to 0, others are fixed to 1, and yet others are converted into continuous variables bounded between 0 and 1. Each linear relaxation has $n_1 + n_2 + 2t$ rows and, depending on how many $r_i$ are fixed, between $2t$ and $n_1 + n_2 + 2t$ columns. Koehler and Erenguc (1990) developed a model similar to Model IV with linear relaxations of only $t$ rows. Koehler and Erenguc first express the $r_i$ in terms of the remaining variables, and then solve the dual of the revised model, which has as many constraints as coefficients to be estimated and can be solved very efficiently. Applying this approach to the linear relaxations of Model IV leads to Model V.

**Model V: Reformulated Linear Relaxation — Primal**

$$\text{Minimize } z_6 = \frac{\pi_1 C(2\mid 1)}{n_1}\left(\sum_{i\in G_1}\frac{\sum_{j=1}^{t}\beta_{ij}y_j + B_{y_{t+1}}\beta_{i,t+1} + s_i}{M_i} - \eta s_i\right)$$

$$+\ \frac{\pi_2 C(1\mid 2)}{n_2}\left(\sum_{i\in G_2}\frac{\sum_{j=1}^{t}\beta_{ij}y_j + B_{y_{t+1}}\beta_{i,t+1} + s_i}{M_i} - \eta s_i\right), \tag{3.43}$$

Subject to:

$$\sum_{j=1}^{t}\beta_{ij}y_j + s_i = M_i r_i - B_{y_{t+1}}\beta_{i,t+1},\ \forall\ i \in G_1 \cup G_2 \text{ for which } r_i \text{ is fixed}, \tag{3.44}$$

$$\sum_{j=1}^{t}\beta_{ij}y_j + s_i \geq -B_{y_{t+1}}\beta_{i,t+1},\ \forall\ i \in G_1 \cup G_2, \text{ for which } r_i \text{ is not fixed}, \tag{3.45}$$

$$L_{c_j} \leq \sum_{k=1}^{t} u_{j+1,k}^{-1} y_k \leq U_{c_j},\ j = 0, \ldots, t;\ j \neq h, \tag{3.46}$$

$$L_{y_j} \leq y_j \leq U_{y_j},\ j = 1, \ldots, t, \tag{3.47}$$

$$y_j \text{ unrestricted},\ j = 1, \ldots, t, \tag{3.48}$$

$$B_{y_{t+1}} = B_{c_h}, \tag{3.49}$$

$$s_i \geq 0,\ \forall\ i \in G_1 \cup G_2, \tag{3.50}$$

The scalars $M_i$ in Model V are calculated using (3.39)–(3.42). Some of the $r_i$ are fixed, either to 0 or 1, and the corresponding constraints in (3.44) are derived from (3.32). The constraints (3.45) apply to those $r_i$ which are not fixed, and are derived from (3.32) and the fact that $r_i \geq 0$. Theorem

3.1 shows that in the optimal solution to Model IV all $r_i \leq 1$, so that it is not necessary to include these constraints explicitly.

**Theorem 3.1.**

Let ($\mathbf{y}^*$, $\mathbf{s}^*$, $\mathbf{r}^*$) be an optimal solution to Model IV, in which some of variables $r_i$ are fixed to 0, others to 1, and the remaining ones satisfy $r_i \geq 0$, rather than (3.37). Then it follows that $r_i^* \leq 1$, $\forall$ $i \in G_1 \cup G_2$.

**Proof:** Suppose that ($\mathbf{y}^*$, $\mathbf{s}^*$, $\mathbf{r}^*$) is an optimal solution and $r_m^* > 1$, for some $m$. From (3.32), (3.34), (3.41) and (3.42) it follows that $s_m^* \geq M_m(r_m^*-1)$. Hence, $s_m$ and $r_m$ can be reduced by $\Delta s_m$ and $\Delta r_i$, respectively, and still remain feasible, as long as $\Delta s_m = M_m \Delta r_m$ and $\Delta s_m \leq s_m^*$, thus reducing the objective function value by $\frac{\Pi_1 C(2\,|\,1)}{n_1}(\frac{1}{M_m} - \eta)\Delta s_m$ if $m \in G_1$ or by $\frac{\Pi_2 C(1\,|\,2)}{n_2}(\frac{1}{M_m} - \eta)\Delta s_m$ if $m \in G_2$, which are positive quantities because $M_m^{-1} - \eta > 0$ by (3.20). Therefore, ($\mathbf{y}^*$, $\mathbf{s}^*$, $\mathbf{r}^*$) cannot be optimal, so that we conclude by contradiction that Theorem 3.1 is true.

Instead of Model V, the D&C algorithm solves its computationally more efficient dual, Model VI.

**Model VI: Reformulated Linear Relaxation — Dual**

$$\text{Maximize } z_7 = \sum_{i \in G_1 \cup G_2} (-B_{y_{t+1}} \beta_{i,t+1} \omega_i) + \sum_{i \in G_1 \cup G_2:\ r_i = 1} M_i \omega_i + \sum_{j=0, j \neq h}^{t} (L_{c_j} \theta_j^- - U_{c_j} \theta_j^+) + \sum_{j=1}^{t} (L_{y_j} \gamma_j^- - U_{y_j} \gamma_j^+), \tag{3.51}$$

Subject to:

$$\sum_{i \in G_1 \cup G_2} \beta_{ij} \omega_i + \sum_{k=0, k \neq h}^{t} (u_{k+1,j}^{-1} \theta_k^- - u_{k+1,j}^{-1} \theta_k^+) + \gamma_j^- - \gamma_j^+ = \frac{\pi_1 C(2\,|\,1)}{n_1} \sum_{i \in G_1} \frac{\beta_{ij}}{M_i} + \frac{\pi_2 C(1\,|\,2)}{n_2} \sum_{i \in G_2} \frac{\beta_{ij}}{M_i},\ j = 1, \ldots, t, \tag{3.52}$$

$$\omega_i \leq \frac{\pi_1 C(2\,|\,1)}{n_1} (\frac{1}{M_i} - \eta),\ \forall\ i \in G_1, \tag{3.53}$$

$$\omega_i \leq \frac{\pi_2 C(1\,|\,2)}{n_2} (\frac{1}{M_i} - \eta),\ \forall\ i \in G_2, \tag{3.54}$$

$$\omega_i \geq 0,\ \forall\ i \in G_1 \cup G_2 \text{ for which } r_i \text{ is not fixed}, \tag{3.55}$$

$$\theta_j^-, \theta_j^+ \geq 0,\ j = 0, \ldots, t,\ j \neq h, \tag{3.56}$$

$$\gamma_j^-, \gamma_j^+ \geq 0,\ j = 1, \ldots, t, \tag{3.57}$$

The optimal values for Models V and VI, $z_6^*$ and $z_7^*$, respectively, differ by a constant, as shown in (3.58),

$$z_6^* = z_7^* + \frac{\pi_1 C(2\,|\,1)}{n_1} \sum_{i \in G_1} \frac{B_{y_{t+1}} \beta_{i,t+1}}{M_i} + \frac{\pi_2 C(1\,|\,2)}{n_2} \sum_{i \in G_2} \frac{B_{y_{t+1}} \beta_{i,t+1}}{M_i}. \tag{3.58}$$

Model VI has $t$ rows, $n_1 + n_2 + 4t$ columns and $n_1 + n_2$ simple bounds. The dual variables $\theta_j^-$ and $\theta_j^+$ correspond with the primal constraints that impose the original lower and upper bounds on the $c_j$ coefficients, the $\gamma_j^-$ and $\gamma_j^+$ with the primal constraints associated with the boundaries of $\mathcal{S}_a(l)$, and the $\omega_i$ with the primal constraints that determine the contribution (through $s_i$ and $r_i$) of each observation to the objective function.

For sub-spaces located within the interior of $\mathcal{S}(l)$, the initial bounds on the $c_j$ are satisfied automatically, in which case the $\theta_j^-$ and $\theta_j^+$ can be eliminated from Model VI. We can verify which of these variables need to be included in the model formulation, by checking whether (3.59) or (3.60) holds.

$$\sum_{k=1}^{t} \text{Min}(L_{y_k} u_{jk}^{-1},\ U_{y_k} u_{jk}^{-1}) \geq L_{c_j}, \tag{3.59}$$

If (3.59) holds for a given sub-space $\mathcal{S}_a(l)$, then $\sum_{k=1}^{t} u_{jk}^{-1} y_k \geq L_{c_j}$ can never be violated, so that $\theta_j^-$ can be omitted from Model VI.

$$\sum_{k=1}^{t} \text{Max}(L_{y_k} u_{jk}^{-1},\ U_{y_k} u_{jk}^{-1}) \leq U_{c_j}, \tag{3.60}$$

Similarly, if (3.60) is satisfied, then $\sum_{k=1}^{t} u_{jk}^{-1} y_k \leq U_{c_j}$ is always satisfied and $\theta_j^+$ can be omitted from Model VI. Furthermore, if $\mathbf{c}$ is restricted to a sufficiently narrow sub-space of its domain, it is possible to identify sets of observations that will be misclassified or classified correctly. For instance, observation $i$ will always be classified correctly if (3.61) holds, and we can set $r_i = 0$. Conversely, observation $i$ will always be misclassified if (3.62) holds, and we can set $r_i = 1$.

$$\sum_{j=1}^{t} \text{Max}(L_{y_j} \beta_{ij},\ U_{y_j} \beta_{ij}) + B_{y_{t+1}} \beta_{i,t+1} < 0, \tag{3.61}$$

$$\sum_{j=1}^{t} \text{Min}(L_{y_j} \beta_{ij},\ U_{y_j} \beta_{ij}) + B_{y_{t+1}} \beta_{i,t+1} > 0, \tag{3.62}$$

We also know that if (3.63) is true, observation $i$ will always be assigned to $G_1$, so that we can set $r_i$ to 0 if $i \in G_1$ or to 1 if $i \in G_2$. If (3.64) holds, observation $i$ will always be assigned to $G_2$, and we can set $r_i$ to 1 if $i \in G_1$ or to 0 if $i \in G_2$.

$$\sum_{j=0,\, j \neq h}^{t} \text{Max}(L_{c_j} \xi_{ij},\ U_{c_j} \xi_{ij}) + B_{c_h} \xi_{ih} < 0, \tag{3.63}$$

$$\sum_{j=0,\, j \neq h}^{t} \text{Min}(L_{c_j} \xi_{ij},\ U_{c_j} \xi_{ij}) + B_{c_h} \xi_{ih} > 0, \tag{3.64}$$

In Model VI, those $\omega_i$ variables corresponding to observations for which the group assignment is not yet fixed (*i.e.*, $r_i$ is variable) are forced to be nonnegative by (3.55). If observation $i$ is forced to be classified correctly – either by pre-checking, or in the course of the branch-and-bound search – and $r_i$ is fixed to 0, the corresponding constraint of the form (3.55) is removed from the model. If observation $i$ is forced to be misclassified and $r_i$ is fixed to 1, the associate constraint (3.55) is removed from the model and the constant $M_i$ is added to the objective coefficient of $\omega_i$.

## 4. THE DIVIDE AND CONQUER ALGORITHM

We are now ready to outline the steps of the D&C algorithm.

**Algorithm I: The Main Algorithm**

**Step 1:** Determine the global bounds $L_{c_j}$ and $U_{c_j}$ for each $c_j$, $j = 0, ..., t$, and normalize the $c_j$ such that $|L_{c_o}| = 1$.

**Step 2:** Solve the dual of Model III. If the optimal solution $z_4^*$ is finite, the groups are perfectly separable and the classification function that yields perfect separation while mimimizing $z_2$ was found. Otherwise, go to Step 3.

**Step 3:** Formulate the $2(t+1)$ Model II sub-problems $P(l)$, $l = 1, ..., 2(t+1)$, fixing each $c_h$ to $L_{c_h}$ and $U_{c_h}$ in turn, and use the procedure described in Algorithm II to solve the corresponding linear relaxations. Denote the space of the $c_j$ that are not fixed by $\mathcal{S}(l)$.

**Step 4:** Sequence the P($l$) in increasing order of the optimal objective function value $z_3^*$ found in Step 2.

For $l = 1$ to $2(t+1)$ do Steps 5 through 9:

**Step 5:** Use the procedure described in Algorithm III to generate $N$ solutions for model $P(l)$. Create the set $\mathcal{B}(l)$ with the best $\alpha N$ of these solutions.

**Step 6:** Reorder $P(l)$ in increasing order of the optimal objective function values $z_3^*$ found in Step 5.

**Step 7:** Compute the principal components of $\mathcal{B}(l)$. Determine the inverse of the $(t+1) \times (t+1)$ matrix $\mathbf{U}$ defined by $\mathbf{y} = \mathbf{Uc}$, where $y_k$ ($k = 1, ..., t$) equals the $k^{\text{th}}$ principal component of $\mathcal{B}(l)$, and $y_{t+1}$ equals the coefficient $c_h$ that is fixed in $P(l)$.

**Step 8:** Use the procedure in Algorithm IV to partition $\mathcal{S}(l)$ into several sub-spaces $\mathcal{S}_a(l)$, $a = 1, ..., r$.

**Step 9:** Use the procedure described in Algorithm V to solve Model IV for each sub-space $\mathcal{S}_a(l)$.

**Step 10:** Stop. The current solution is optimal for the MP-$L_0$ classification problem, using $z_2$ as the secondary objective.

In Steps 4 and 6 of Algorithm I, $P(l)$ is ordered in increasing order of the objective values $z_3^*$ of the incumbent solutions for Model II. The purpose of the ordering is to solve the models which are more likely to have good solutions first, so that solutions with low objective function values may be identified relatively fast. Proceeding in this manner renders the D&C algorithm faster, because the pre-checking process can quickly eliminate several models from consideration, speeding up the branch-and-bound algorithm for the remaining models.

Algorithm II describes the procedure used to solve the linear relaxation of Model II. Algorithm III details how to generate an initial set of solutions $\mathfrak{B}(l)$ for each sub-problem P($l$). Algorithm IV describes the procedure to partition $\mathfrak{I}(l)$ into sub-spaces $\mathfrak{I}_1(l)$, ..., $\mathfrak{I}_r(l)$. Algorithm V describes the procedure used to solve the model associated with each sub-space.

**Algorithm II: Solving the Linear Relaxation of Each Major Sub-Problem**

**Step 1:** Use equations (3.63) and (3.64) to determine which observations are always classified correctly when $c_h = B_{c_h}$, and set the associated variables $r_i = 0$.

**Step 2:** Use (3.63) and (3.64) to determine which observations are always misclassified if $c_h = B_{c_h}$ (*i.e.*, either $L_{c_h}$ or $U_{c_h}$), and let the corresponding $r_i = 1$.

**Step 3:** Formulate and solve Model VI without the $\gamma_j^-$ and the $\gamma_j^+$ variables and with $\mathbf{U} = \mathbf{I}$, where $\mathbf{I}$ is the $(t+1) \times (t+1)$ identity matrix.

**Algorithm III: Generating An Initial Set of Solutions $\mathfrak{B}(l)$ For Each Sub-Problem**

**Step 1:** Use equations (3.63) and (3.64) to determine which observations are always correctly classified if $c_h = B_{c_h}$, and let the corresponding $r_i = 0$.

**Step 2:** Use equations (3.63) and (3.64) to determine which observations are always misclassified if $c_h = B_{c_h}$, and let the corresponding $r_i = 1$.

**Step 3:** Generate a set of solutions for problem $P(l)$ using a limited breadth-first, branch-and-bound search. Start by treating all $r_i$ variables not fixed in Steps 1 and 2 as free variables. At each level of the search tree, set one of the $r_i$ equal to 1, in turn.

**Step 4:** For each of the linear relaxations in Step 3, formulate and solve Model VI without the $\gamma_j^-$ and the $\gamma_j^+$ variables and with $\mathbf{U} = \mathbf{I}$. Determine $\mathfrak{B}(l)$ as the set of $\alpha N$ best among the solutions found.

**Algorithm IV: Partitioning the Space of the Non-Fixed Coefficients $\mathfrak{I}(l)$**

**Step 1:** Select values for SZO($l$) and IFT$_k$($l$), $k = 1$, ..., $t$, determine the centroid $\overline{\mathbf{c}}$ of the set $\mathfrak{B}(l)$, and compute $\overline{\mathbf{y}} = \mathbf{U}\overline{\mathbf{c}}$, where $\mathbf{U}$ is described in Step 7 of Algorithm I.

**Step 2:** Create $\mathfrak{I}_1(l)$ by setting SZ$_k$($l$) = SZO($l$), $k = 1$, ..., $t$. Set the bounds $L_{y_j}$ and $U_{y_j}$ to the following values: $L_{y_j} = \overline{y}_j - 0.5\text{SZ}_j(l)$, $U_{y_j} = \overline{y}_j + 0.5\text{SZ}_j(l)$, $j = 1, ..., t$.

**Step 3:** Create $\mathfrak{I}_2(l)$, ..., $\mathfrak{I}_r(l)$ by changing the bounds $L_{y_j}$ and $U_{y_j}$. Start from $\mathfrak{I}_1(l)$ and move towards the boundaries of $\mathfrak{I}(l)$. Each time a move is made along a direction associated with the $j^{\text{th}}$ principal component of $\mathfrak{B}(l)$, multiply the side lengths SZ$_k$($l$) by the factor IFT$_k$($l$), $k = 1$, ..., $j$. Stop when $\mathfrak{I}(l)$ is totally covered by the sub-spaces created in this step.

**Algorithm V: Solving a Partial Sub-Problem**

**Step 1:** Use equations (3.59) and (3.60) to find the boundaries of $\mathfrak{I}(l)$ that are active in the current sub-space $\mathfrak{I}(l)$.

**Step 2:** Use equations (3.61), (3.63) and (3.64) to determine which observations are always classified correctly when $c_h = B_{c_h}$ and **c** is restricted to $\mathcal{I}_a(l)$. Set the associated variables $r_i$ equal to 0.

**Step 3:** Use equations (3.62), (3.63) and (3.64) to determine which observations are always misclassified when $c_h = B_{c_h}$. Restrict the coefficients $c_j$ that are not fixed to $\mathcal{I}_a(l)$, and set the corresponding variables $r_i$ equal to 1.

**Step 4:** Solve Model IV by the branch-and-bound algorithm. Use the formulation in Model VI to solve each linear relaxation derived from Model IV. Include only the variables $\theta_j^-$ and $\theta_j^-$ associated with the boundaries of $\mathcal{I}(l)$ that are active in $\mathcal{I}_a(l)$.

## 5. COMPUTATIONAL EXPERIMENTS

The D&C algorithm described in Section 4 is implemented in the $C^{++}$ programming language. The code is available from the authors upon request. Some of the functions used in this code were developed by Koehler and Erenguc (1990), who were kind enough to share the source code of their programs. Other functions are our original work, and yet others are adapted from Koehler and Erenguc's (1990) code. In this section, we report the results of simulation experiments to assess the relative computational performance of the D&C algorithm, the two fastest existing MP-based algorithms for solving MP-$L_0$ classification problems, one developed by Banks and Abad (1991) (B&A), the other by Koehler and Erenguc (1990) (K&E), and the adapted Warmack-Gonzalez algorithm as implemented by Soltysik and Yarnold (1994) (S&Y). The S&Y algorithm is not MP-based.

Initially, in Table 1 we compare the computational effort of the D&C algorithm with the results reported in B&A and K&E, in terms of the number of LPs, major pivots and pricings. The S&Y algorithm cannot be compared in terms of statistics pertaining to MP operations. All of the results in Table 1 refer to problems with 100 training sample observations ($n_1 = n_2 = 50$) and 3 independent, identically distributed attributes. The attributes of the observations in $G_1$ are normally distributed with a mean of 0 and a variance of 1, whereas those in $G_2$ follow the normal distribution with a mean of 0.6 and a variance of 2. The misclassification costs and prior probabilities are assumed to be equal across groups.

-------------------------------

Table 1 About Here

-------------------------------

The results reported by K&E are based on 100 replications, while the B&A and D&C experiments involve 20 replications. The figures in Table 1 indicate that the D&C algorithm is much more efficient computationally than the K&E and B&A algorithms. The computational effort of the D&C algorithm is greatly reduced, with on average about 35 times less LPs solved, 4.5 times less pivots and 90 times less pricings than K&E, and with about 5 times less LPs and 3 times less pivots than Banks and Abad. Koehler and Erenguc reported an average CPU time for their algorithm of 3 minutes on an IBM 3090/400 mainframe. On average, the D&C algorithm required 18 CPU seconds on a 486 DX2 (66 Mhz), DX2 Personal Computer with 16MB of RAM. Banks and Abad did not report solution times or pricing information for their algorithm.

The two right-most columns of Table 1 provide information on the average number of sub-problems created (SBP-CRT) and actually solved (SBP-SLV) by the D&C algorithm. Not all sub-problems created are actually solved, since it is possible to recognize *a priori* that some sub-regions of the coefficient space cannot contain solutions with a lower training sample cost than the current incumbent solution. The two right-most columns do not apply to the B&A and K&E algorithms, since these did not divide the original problem into sub-problems.

Soltysik and Yarnold (1994) report experimental results which show that, at least for data conditions similar to those studied by Koehler and Erenguc (1990) and Banks and Abad (1991), with 100 training sample observations and 3 attributes), the S&Y algorithm is considerably faster than the MP-based B&A and K&E algorithms. Therefore, we performed an experiment to measure the relative efficiency of the D&C and S&Y algorithms, comparing the CPU time required to find the linear MP-$L_0$ classification rule using PC implementations of both algorithms, run on a 486 DX2 (66 Mhz), DX2 Personal Computer with 16MB of RAM. Since the S&Y algorithm is not based on MP models, the two algorithms can be compared only in terms of CPU times.

In our experiments, we analyzed problems with 2, 3, 4 and 5 attributes. The attributes of the training sample observations in $G_1$ were generated from the multivariate normal distribution with mean vector $\boldsymbol{\mu} = (0, ..., 0)^T$ and variance-covariance matrix $\boldsymbol{\Sigma} = \mathbf{I}$, and those in $G_2$ from the multivariate normal distribution with $\boldsymbol{\mu} = (1, ..., 1)^T$ and $\boldsymbol{\Sigma} = \mathbf{I}$. This data condition corresponds with the "high discrimability problems" considered by Soltysik and Yarnold (1994). The training samples generated were balanced. Intitially, we generated and solved problems with a total of 50 and 100 observations. Subsequently, as long as none of the problems required more than 10 CPU minutes to solve, we extended the computational experiment to larger training samples, in increments of 100 observations. The largest problem considered had two attributes and 1,000 observations in the training sample. All of the classification rules used were linear, and the computational results are based on 10 replications for each data condition. The computational results of our experiments are summarized in Tables 2–5.

---

Figure 2 and Tables 2–5 About Here

---

The figures in Tables 2–5 clearly show that the time required to determine the MP-$L_0$ classification rule increases exponentially as a function of the training sample size. For instance, when the training sample size is doubled from 50 to 100 observations, the mean solution time of the S&Y algorithm increases from 10.7 to 263.2 seconds for 4-attribute problems, and from 45.4 to 4,200 seconds for 5-attribute problems. Although the computational effort for the D&C algorithm is much less, the exponential growth in computational burden is evident for this algorithm as well, with an increase from 5.3 to 29.5 and from 3.5 to 78.5 seconds for 4-attribute and 5-attribute problems, respectively. The exponential growth for the case of 3-attribute problems is displayed graphically in Figure 2.

The exponential growth becomes more dramatic as the number of attributes increases. For instance, when the training sample size for 2-attribute problems is increased from 50 to 100, the CPU time grows by a factor of about 2.5 (from 0.5 to 1.4 and from 0.4 to 0.9 seconds for the D&C and the S&Y algorithm, respectively). For 4-attribute problems, the corresponding growth factors are 5.6 (D&C) and 24.5 (S&Y), and for 5-attribute problems are 22.7 (D&C) and 92.4 (S&Y).

This computational behavior has two important consequences: (1) for "small problems," with few attributes and small training samples, it is possible to determine MP-$L_0$ classification rules very quickly. However, for "larger problems" the computer resources required become prohibitive; (2) the training sample size for which it is possible to find MP-$L_0$ rules within a "reasonable" time strongly depends on how many attributes the problem has. For 2-attribute problems, training samples with more than 1,000 observations can still be analyzed, but the limit on the training sample size decreases quickly as the number of attributes increases.

Although the S&Y is faster than the D&V algorithm for "small" problems (*i.e.*, for 2-attribute problems with less than 200 observations and 3-attribute problems with less than 50 observations), it is evident from Tables 2–5 that the solution time of the S&Y algorithm grows faster than that of the D&C algorithm, as the training sample size and the number of number of attributes increases. For example, for 4-attribute problems with more than 200 observations and 5-attribute problems with more than 100 observations, the D&C algorithm is over 50 times faster than the S&Y algorithm. This impressive improvement in relative efficiency of the D&C algorithm is due to the fact that, whereas the solution times of the S&Y algorithm explode quickly, even for moderate numbers of attributes, the D&C algorithm is able to moderate this effect by judicially dividing the problem into sub-problems, thus reducing the computational burden and facilitating the solution of substantially larger size problems within a reasonable time.

In order to better understand the effect of training sample size on the CPU times required by both algorithms, we regressed the logarithms of the CPU seconds, $ln(T)$, against the logarithms of the number of training sample observations, $ln(n)$, for each number of attributes considered. The estimated regressions, with the standard errors within brackets below the coefficient estimates, are presented in Table 6.

-----------------------------

Table 6 About Here

-----------------------------

The high $R^2$ values of between 85 and 98 percent reveal a strong linear relation between $ln(T)$ and $ln(n)$, implying an exponential relationship between the original variables, $T$ and $n$. The effect of the number of attributes on the rate of exponential growth of the CPU time for the S&Y algorithm is shown by the increase of the coefficient of $ln(n)$. For the S&Y algorithm, this coefficient increases from 2.57 for 2-attribute problems to 6.59 for 5-attribute problems; for the D&C algorithm it increases from

1.73 to 4.20. As the coefficients of $ln(n)$ for the D&C algorithm are always smaller than the corresponding coefficients for the S&Y algorithm, the regression models confirm the notion that the growth rate of the computational effort for the D&V algorithm is slower than that for the S&Y algorithm.

We can also use the regression models in Table 6 to estimate the maximum training sample sizes that could be solved within a given amount of time $T$, using a 486 DX2 (66 Mhz) PC. The estimates of the largest training sample sizes that can be solved in 10, 60, 600 and 3,600 CPU seconds are presented in Table 7.

---

Figure 3 and Table 7 About Here

---

Table 7 illustrates that a small reduction in the growth rate of the solution time can yield a dramatic improvement in computational efficiency. For instance, in ten CPU minutes (600 seconds) the D&C algorithm is able to solve problems with about three times more observations than the S&Y algorithm. Problems with 5 attributes and 100 observations would take about one CPU hour (3600 seconds) and one CPU minute (60 seconds) using the S&Y and D&V algorithm, respectively. The projection for the case of 4-attribute problems are shown in Figure 3.

In spite of these promising results, the current implementation of the D&C algorithm is still unable to find solutions to some common pattern recognition problems involving more than 1,000 observations and more than 10 attributes, within a reasonable amount of time. Furthermore, for these problems the availability of faster hardware would not have a significant impact, as the increase in training sample size that can be analyzed associated with a given reduction of the computational requirement is less than proportional. For instance, for 5-attribute problems a tenfold increase in CPU time is not enough to double the maximum training sample size that can be handled. However, further reductions in the *growth rate* of the CPU solution time can increase the size of problems that can be analyzed substantially.

Our current research focuses on how to improve and fine-tune the D&C algorithm. It is our belief that future implementations of the D&C algorithm will have a considerably lower growth rate of CPU time, and will facilitate the solution of considerably larger problems, particularly problems with larger numbers of attributes, than is currently feasible.

## 6. CONCLUSIONS

In this paper, we introduce the Divive and Conquer (D&C) algorithm, a special-purpose algorithm for solving MP-$L_0$ classification problems. Our computational tests show that, except for small sample sizes – in which case there are no computational difficulties anyway – the D&C algorithm solves the MP-$L_0$ classification problem much faster than previously proposed algorithms, MP-based and non-MP-based alike. By partitioning the problem into smaller sub-problems, the D&C

algorithm reduces the computational effort required dramatically. As, in comparison with existing special-purpose algorithms, the D&C algorithm greatly reduces the exponential growth rate of the computational requirements as a function of the training sample size and the number of attributes, it contributes significantly to the field of MP-$L_0$ classification analysis, facilitating the analysis of much larger training sample data sets than previously possible.

The current research can be extended in several different ways. First, it is worthwhile to explore parallel implementations of the D&C algorithm, improving the computational efficiency even further. Second, the D&C algorithm can be refined in several respects. It appears particularly useful to focus on improvements which reduce the exponential growth rate of the computational requirements. Third, whereas the D&C algorithm solves to optimality, it may be possible to develop tabu-search heuristics that provide close approximations to the optimal solution of large MP-$L_0$ classification problems.

## REFERENCES

Abad, P. L. and Banks, W. J., "New LP Based Heuristics for the Classification Problem," *European Journal of Operational Research*, **27**, 1993, 88–100.

Anderson, J. A., "Separate Sample Logistic Discrimination," *Biometrika*, **59**, 1972, 19–35.

Anderson, T. W., *An Introduction to Multivariate Statistical Analysis*, Second Edition, Wiley, New York, NY, 1984.

Asparoukhov, O. K., *Microprocessor System for Investigation of Thromboembolic Complications*, Unpublished Ph.D. Dissertation, Technical University of Sofia, Bulgaria, 1985 (in Bulgarian).

Bajgier, S. M. and Hill, A., "An Experimental Comparison of Statistical and Linear Programming Approaches to the Discriminant Problem," *Decision Sciences*, **13**, 1982, 604–618.

Banks, W. J. and Abad, P. L., "An Efficient Optimal Solution Algorithm for the Classification Problem," *Decision Sciences*, **22**, 1991, 1008–1023.

Banks, W. J. and Abad, P. L., "On the Performance of Linear Programming Heuristic Applied on a Quadratic Transformation in the Classification Problem," *European Journal of Operational Research*, **74**, 1994, 23–28.

Campbell, T. S. and Dietrich, J. K., "The Determinants of Default on Insured Conventional Residential Mortgage Loans," *Journal of Finance*, **38**, 1983, 1569–1581.

Capon, N., "Credit Scoring Systems: A Critical Analysis," *Journal of Marketing*, **46**, 1982, 82–91.

Chen, C., "Hybrid Misclassification Minimisation," paper presented at the National INFORMS Meeting, Washington, DC, May 1996.

Duarte Silva, A. P., *Minimizing Misclassification Costs in Two-Group Classification Analysis*, Unpublished Ph.D. Dissertation, The University of Georgia, 1995.

Duarte Silva, A. P. and Stam, A., "Second Order Mathematical Programming Formulations for Discriminant Analysis," *European Journal of Operational Research*, **74**, 1994, 4–22.

Eisenbeis, R. A., "Pitfalls in the Application of Discriminant Analysis," *Journal of Finance*, **32**, 1977, 875–900.

Fatti, L. P., Hawkins, D. M. and Raath, E. L., "Discriminant Analysis," in: *Topics in Applied Multivariate Analysis*, D. M. Hawkins (Ed.), Cambridge University Press, Cambridge, England, 1982, 1–77.

Fisher, R. A., "The Use of Multiple Measurements in Taxonomy Problems," *Annals of Eugenics*, **7**, 1936, 179–188.

Freed, N. and Glover, F., "A Linear Programming Approach to the Discriminant Problem," *Decision Sciences*, **12**, 1981a, 68–74.

Freed, N. and Glover, F., "Simple But Powerful Goal Programming Formulations for the Discriminant Problem," *European Journal of Operational Research*, **7**, 1981b, 44–60.

Freed, N. and Glover, F., "Evaluating Alternative Linear Programming Models to Solve the Two-Group Discriminant Problem," *Decision Sciences*, **17**, 1986, 151–162.

Gehrlein, W. V., "General Mathematical Programming Formulations for the Statistical Classification Problem," *Operations Research Letters*, **5**, 1986, 299–304.

Glorfeld, L. W. and Kattan, M. W., "A Comparison of the Performance of Three Classification Procedures When Applied to Contaminated Data," in: *Proceedings of the 21st Annual Meeting of the Decision Sciences Institute*, 1989, 1153–1155.

Glover, F., "Improved Linear Programming Models for Discriminant Analysis," *Decision Sciences*, **21**, 1990, 771–785.

Glover, F., Keene, S. and Duea, B., "A New Class of Models for the Discriminant Problem," *Decision Sciences*, **19**, 1988, 269–280.

Gochet, W., Srinivasan, V., Stam, A. and Chen, S., "Multi-Group Discriminant Analysis Using Linear Programming," *Onderzoeksrapport #9305*, Department of Applied Economic Sciences, Katholieke Universiteit Leuven, Leuven, Belgium, 1993 (Forthcoming in *Operations Research*).

Hand, D. J., *Discrimination and Classification*, John Wiley, New York, NY, 1981.

Hillier, F. S. and Lieberman, G. J., *Introduction to Operations Research, Fifth Edition*, McGraw-Hill, New York, 1990.

Huberty, C. J., "Issues in the Use and Interpretation of Discriminant Analysis," *Psychological Bulletin*, **95**, 1984, 156–171.

Ibaraki, T. and Muroga, S., "Adaptive Linear Classifier by Linear Programming," *IEEE Transactions on Systems, Science and Cybernetics*, **SSC-6**, 1970, 53–62.

International Business Machines, *IBM Mathematical Programming Systems Extended/370 (MPSX/370), Mixed Integer Programming/370 (MIP/370)*, White Plains, New York, NY, 1975.

Joachimsthaler, E. A. and Stam, A., "Four Approaches to the Classification Problem in Discriminant Analysis: An Experimental Study," *Decision Sciences*, **19**, 1988, 322–333.

Joachimsthaler, E. A. and Stam, A., "Mathematical Programming Approaches for the Classification Problem in Two-Group Discriminant Analysis," *Multivariate Behavioral Research*, **25**, 1990, 427–454.

Kim, L. and Kim, Y., "Innovation in a Newly Industrializing Country: A Multiple Discriminant Analysis," *Management Science*, **31**, 1985, 312–322.

Koehler, G. J. and Erenguc, S. S., "Minimizing Misclassifications in Linear Discriminant Analysis," *Decision Sciences*, **21**, 1990, 63–85.

Koford, J. S. and Groner, G. F., "The Use of an Adaptive Threshold Element to Design a Linear Optimal Pattern Classifier," *IEEE Transactions on Information Theory*, **IT-12**, 1966, 42–50.

Lachenbruch, P. A., Sneeringer, C. and Revo, L. T., "Robustness of the Linear and Quadratic Discriminant Function to Certain Types of Non-Normality," *Communications in Statistics*, **1**, 1973, 39–57.

Liitschwager, J. M. and Wang, C., "Integer Programming Solution of a Classification Problem," *Management Science*, **24**, 1978, 1515–1525.

Mahmood, M. A. and Lawrence, E. C., "A Performance Analysis of Parametric and Nonparametric Discriminant Approaches to Business Decision Making," *Decision Sciences*, **18**, 1987, 308–326.

Mangasarian, O. L., "Linear and Nonlinear Separation of Patterns by Linear Programming," *Operations Research*, **13**, 1965, 444–452.

Markowski, C. A. and Markowski, E. P., "Some Difficulties and Improvements in Applying Linear Programming Formulations to the Discriminant Problem," *Decision Sciences*, **16**, 1985, 237–247.

Markowski, C. A. and Markowski, E. P., "An Experimental Comparison of Several Approaches to the Discriminant Problem With Both Qualitative and Quantitative Variables," *European Journal of Operational Research*, **28**, 1987, 74–78.

McLachlan, G. J., *Discriminant Analysis and Statistical Pattern Recognition*, Wiley, New York, NY, 1992.

Morrison, D. F., *Multivariate Statistical Methods*, Third Edition, McGraw-Hill, New York, NY, 1990.

Pinches, G. E. and Mingo, K. A., "A Multivariate Analysis of Industrial Bond Ratings," *Journal of Finance*, **28**, 1973, 1–18.

Ragsdale, C. T. and Stam, A., "An Efficient Heuristic Method for Minimizing the Number of Misclassified Observations in Discriminant Analysis," Working Paper, Terry College of Business, The University of Georgia, 1991.

Ramanujan, V., Venkatraman, N. and Camillus, J. C., "Multi-Objective Assessment of Effectiveness of Strategic Planning: A Discriminant Analysis Approach," *Academy of Management Journal*, **29**, 1986, 347–372.

Rubin, P. A., "Heuristic Solution Procedures for a Mixed-Integer Programming Discriminant Model," *Managerial and Decision Economics*, **11**, 1990, 255–266.

Rubin, P. A., "A Comment Regarding Polynomial Discriminant Analysis," *European Journal of Operational Research*, **72**, 1994, 29–31.

Schrage, L., *LINDO: User's Manual, Release 5.0*, The Scientific Press, South San Francisco, CA, 1991.

Smith, C. A. B., "Some Examples of Discrimination," *Annals of Eugenics*, **13**, 1947, 272–282.

Smith, F. W., "Pattern Classifier Design by Linear Programming," *IEEE Transactions on Computers*, **C–17**, 1968, 367–372.

Soltysik, R. C. and Yarnold, P.R., "Fast Solutions to Optimal Discriminant Analysis Problems," presented at the *TIMS/ORSA National Meeting* (Invited), Orlando, FL, April 1992.

Soltysik, R. C. and Yarnold, P.R., "*ODA 1.0: Optimal Discriminant Analysis for DOS*, Optimal Data Analysis, Chicago, IL, 1993.

Soltysik, R. C. and Yarnold, P. R., "The Warmack-Gonzalez Algorithm for Linear Two-Category Multivariate Optimal Discriminant Analysis," *Computers & Operations Research*, **21**, 1994, 735–745.

Spiegelhalter, D. J. and Knill-Jones, R. P., "Statistical and Knowledge-Based Approaches to Clinical Decision-Support Systems, With an Application to Gastroenterology," *Journal of the Royal Statistical Society, Series A 147*, Part 1, 1984, 35–77.

Srinivasan, V. and Kim, Y. H., "Credit Granting: A Comparative Analysis of Classification Procedures," *Journal of Finance*, **42**, 1987, 665–683.

Stam, A. and Joachimsthaler, E. A., "Solving the Classification Problem in Discriminant Analysis via Linear and Nonlinear Programming Methods," *Decision Sciences*, **20**, 1989, 285–293.

Stam, A. and Joachimsthaler, E. A., "A Comparison of a Robust Mixed-Integer Approach to Existing Methods for Establishing Classification Rules for the Discriminant Problem," *European Journal of Operational Research*, **46**, 1990, 113–120.

Stam, A. and Jones, D. G., "Classification Performance of Mathematical Programming Techniques in Discriminant Analysis: Results for Small and Medium Sample Sizes," *Managerial and Decision Economics*, **11**, 1990, 243–253.

Stam, A. and Ragsdale, C. T., "On the Classification Gap in MP-Based Approaches to the Discriminant Problem," *Naval Research Logistics*, **39**, 1992, 545–559.

Warmack, R. E. and Gonzalez, R. C., "An Algorithm for the Optimal Solution of Linear Inequalities and its Application to Pattern Recognition," *IEEE Transactions on Computers*, **C–22**, 1973, 1065–1075.

Yarnold, P.R., Soltysik, R. C. and Martin, G. J., "Heart Rate Variability and Susceptibility for Sudden Cardiac Death: An Example of Multivariable Optimal Discriminant Analysis," *Statistics in Medicine*, **13**, 1994, 1015–1021.

Figure 1: A Typical Partition of $\mathcal{I}(I)$

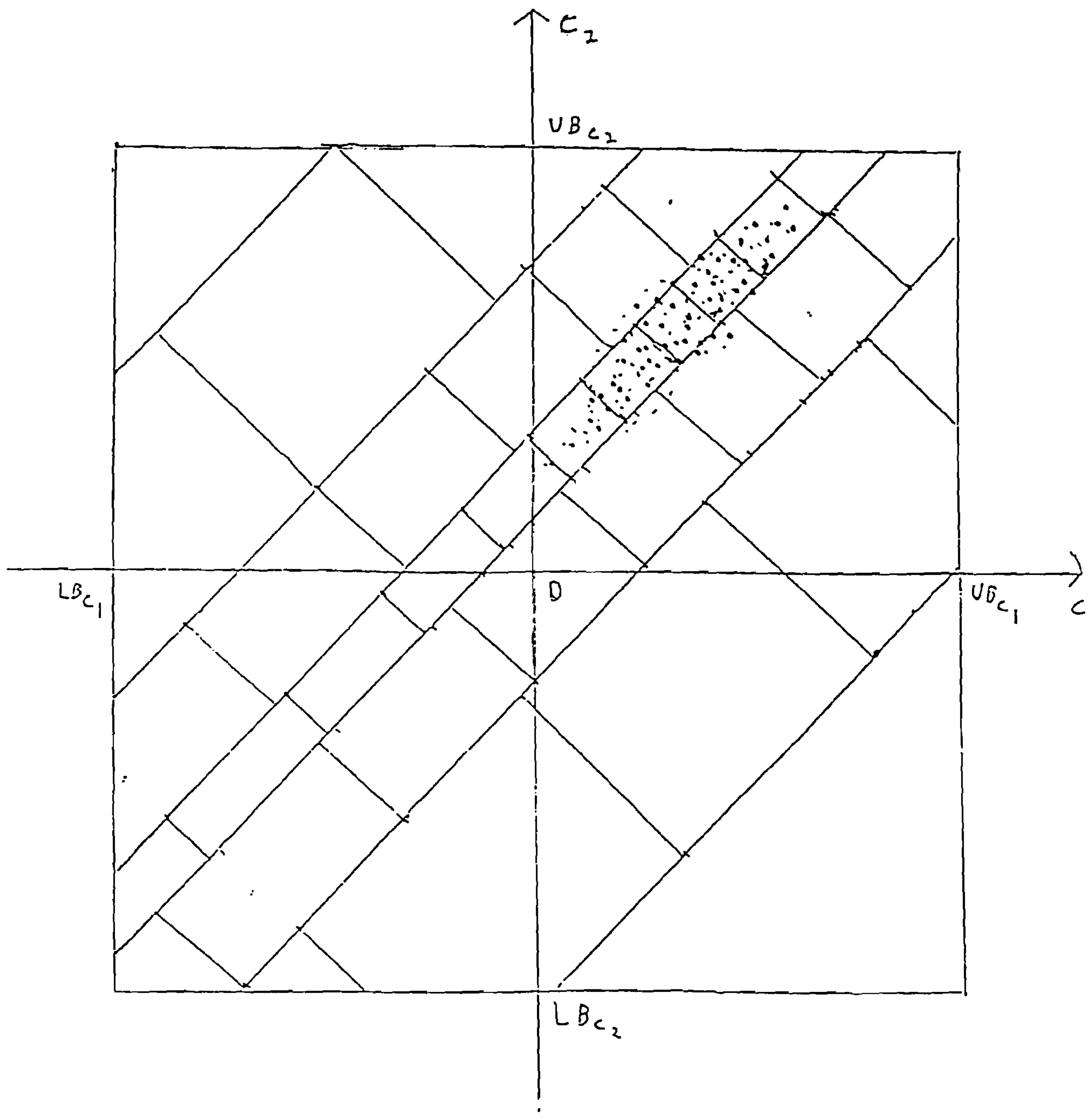

Figure 2: Average Computational Effort for the D&C and S&Y Algorithms, 3-Attribute Problems

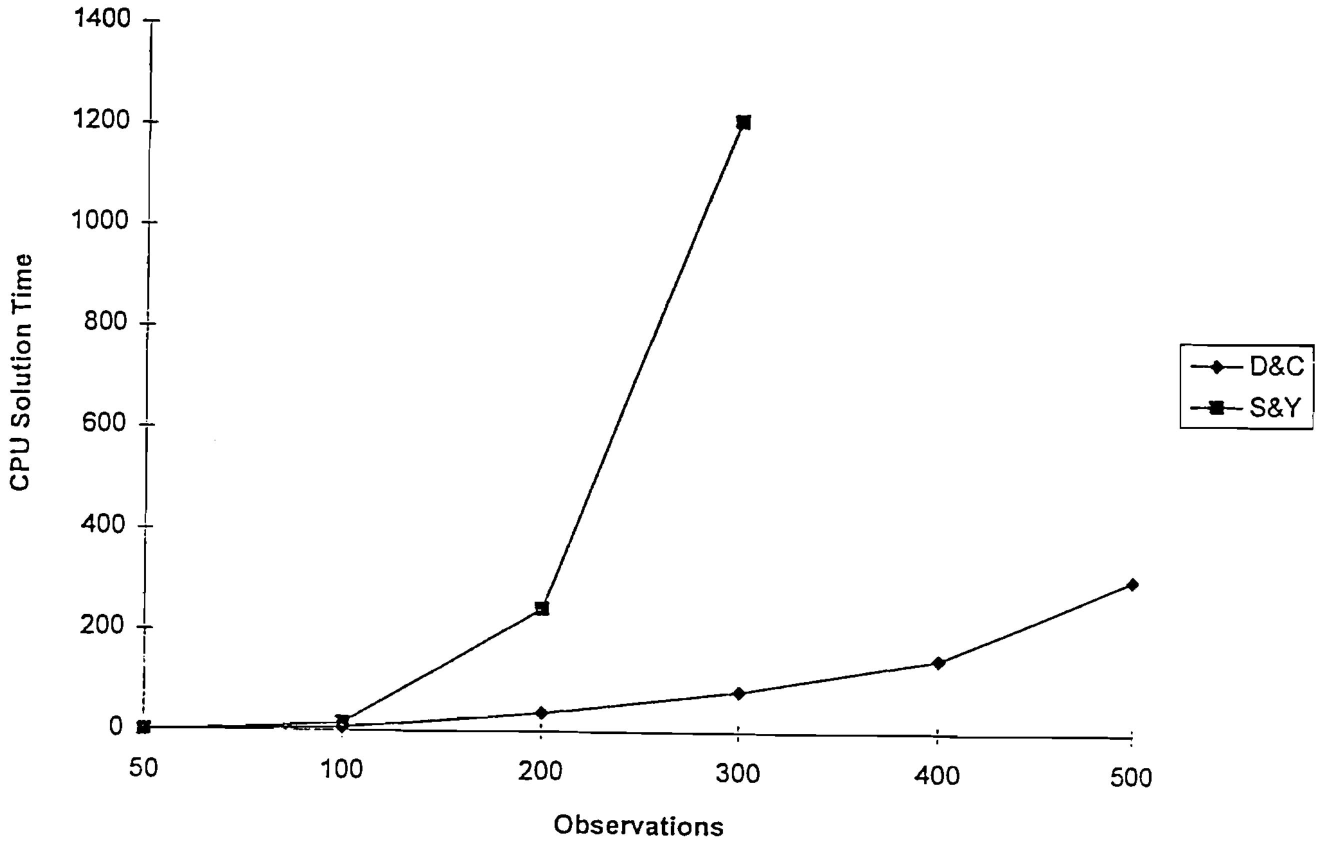

**Figure 3: Projected Relationship of Computational Effort and Maximum Training Sample Sizes for the D&C and S&Y Algorithms, 4-Attribute Problems**

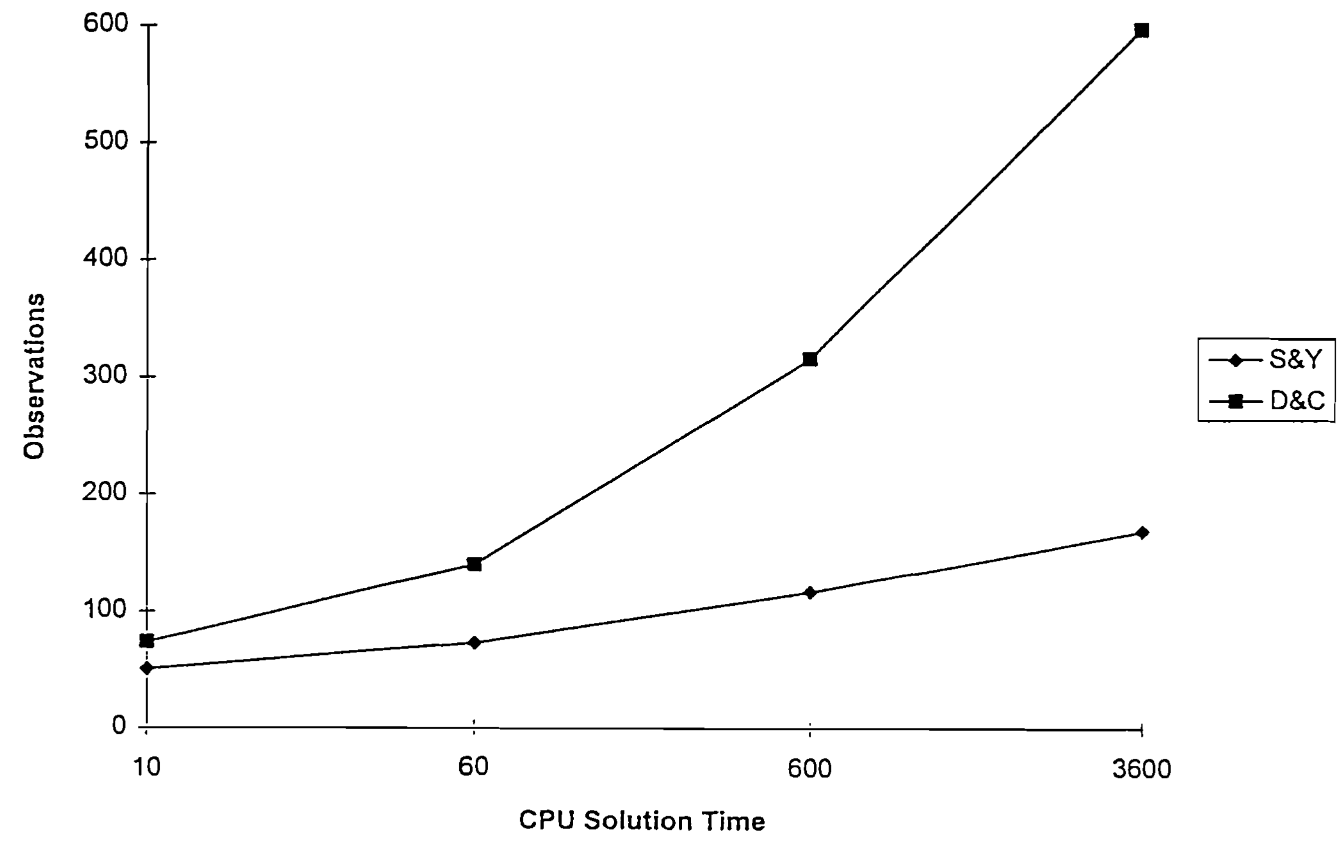

**Table 1: Computational Effort of the B&A, K&E and D&C Algorithms**

| Algorithm | Number of LPs | Number of Major Pivots | Pricing | SBP-CRT | SBP-SLV |
|---|---|---|---|---|---|
| K&E | Mean 61,821 | 53,485 | 15,478,800 | – | – |
| | Std. 30,515 | 16,673 | 6,609,405 | – | – |
| B&A | Mean 8,940 | 35,739 | – | – | – |
| | Std. 5,242 | 15,077 | – | – | – |
| D&C | Mean 1,707 | 12,099 | 172,027 | 2,565 | 290 |
| | Std. 1,040 | 4,266 | 64,372 | 606 | 94 |

**Table 2: Execution Times (CPU Seconds) for the D&C and S&Y Algorithms: Problems with Two Attributes**

| Number of | D&C | | | S&Y | | |
|---|---|---|---|---|---|---|
| Observations | Mean | Min | Max | Mean | Min | Max |
| 50 | 0.5 | 0.4 | 0.8 | 0.4 | 0.2 | 0.6 |
| 100 | 1.4 | 0.9 | 1.9 | 0.9 | 0.5 | 1.5 |
| 200 | 4.0 | 2.9 | 5.1 | 4.5 | 3.4 | 7.4 |
| 300 | 8.4 | 6.5 | 10.8 | 14.9 | 11.2 | 21.5 |
| 400 | 14.0 | 11.5 | 16.4 | 35.3 | 24.6 | 50.8 |
| 500 | 20.6 | 16.5 | 25.8 | 67.2 | 49.1 | 93.4 |

**Table 3: Execution Times (CPU Seconds) for the D&C and S&Y Algorithms: Problems with Three Attributes**

| Number of | D&C | | | S&Y | | |
|---|---|---|---|---|---|---|
| Observations | Mean | Min | Max | Mean | Min | Max |
| 50 | 1.9 | 0.9 | 3.4 | 1.7 | 1.0 | 2.8 |
| 100 | 7.1 | 5.3 | 10.9 | 16.0 | 8.3 | 23.3 |
| 200 | 36.9 | 24.1 | 47.7 | 242.9 | 88.5 | 411.5 |
| 300 | 81.3 | 64.2 | 108.2 | 1,210.9 | 660.1 | 1,871.2 |
| 400 | 146.6 | 109.8 | 184.8 | – | – | – |
| 500 | 305.1 | 185.2 | 687.5 | – | – | – |

**Table 4: Execution Times (CPU Seconds) for the D&C and S&Y Algorithms: Problems with Four Attributes**

| Number of | D&C | | | S&Y | | |
|---|---|---|---|---|---|---|
| Observations | Mean | Min | Max | Mean | Min | Max |
| 50 | 5.3 | 0.6 | 15.2 | 10.8 | 1.9 | 17.4 |
| 100 | 29.5 | 11.3 | 66.0 | 263.2 | 103.4 | 436.2 |
| 200 | 181.5 | 70.0 | 311.3 | 9,076.6 | 4,732.2 | 13,999.3 |
| 300 | 488.2 | 307.9 | 788.1 | – | – | – |

**Table 5: Execution Times (CPU Seconds) for the D&C and S&Y Algorithms: Problems with Five Attributes**

| Number of Observations | D&C Mean | D&C Min | D&C Max | S&Y Mean | S&Y Min | S&Y Max |
|---|---|---|---|---|---|---|
| 50 | 3.5 | 0.1 | 9.4 | 45.4 | 12.6 | 106.2 |
| 100 | 78.5 | 15.5 | 148.8 | 4,200.0 | 1,986.2 | 8,882.5 |
| 200 | 600.9 | 301.8 | 876.2 | – | – | – |

**Table 6: Estimated Regression Models (Computational Effort vs. Training Sample Size)**

| Problems | S&Y Algorithm | D&C Algorithm |
|---|---|---|
| Two-Attribute Problems | $ln(T) = -11.74 + 2.57 ln(n)$ (.224) (.037) $R^2 = 0.98$ | $ln(T) = -7.65 + 1.73 ln(n)$ (.130) (.022) $R^2 = 0.98$ |
| Three-Attribute Problems | $ln(T) = -14.01 + 3.68 ln(n)$ (.340) (.081) $R^2 = 0.98$ | $ln(T) = -8.10 + 2.20 ln(n)$ (.244) (.046) $R^2 = 0.98$ |
| Four-Attribute Problems | $ln(T) = -16.99 + 4.91 ln(n)$ (.765) (.168) $R^2 = 0.97$ | $ln(T) = -9.94 + 2.84 ln(n)$ (.826) (.168) $R^2 = 0.88$ |
| Five-Attribute Problems | $ln(T) = -22.12 + 6.59 ln(n)$ (1.415) (.331) $R^2 = 0.95$ | $ln(T) = -15.65 + 4.20 ln(n)$ (1.505) (.324) $R^2 = 0.85$ |

**Table 7: Estimates of Largest Training Sample Sizes That Can be Analyzed with a 486 PC at 66 MHz**

| Number of | | CPU Time (Seconds) | | | |
|---|---|---|---|---|---|
| Attributes | Algorithm | 10 | 60 | 600 | 3600 |
| 2 | S&Y | 234 | 470 | 1150 | 2308 |
| | D&C | 311 | 876 | 3310 | 9310 |
| 3 | S&Y | 84 | 137 | 257 | 419 |
| | D&C | 114 | 258 | 736 | 1664 |
| 4 | S&Y | 51 | 73 | 117 | 169 |
| | D&C | 74 | 140 | 316 | 596 |
| 5 | S&Y | 40 | 53 | 75 | 99 |
| | D&C | 72 | 110 | 191 | 293 |