Feature Selection via Mathematical Programming

P. S. Bradley†, O. L. Mangasarian† and W. N. Street‡

† Computer Sciences Department, University of Wisconsin, 1210 West Dayton Street, Madison, WI 53706, email: *paulb@cs.wisc.edu, olvi@cs.wisc.edu.*

‡ Computer Science Department, 207 Mathematical Sciences, Oklahoma State University, Stillwater, OK 74078, email: *nstreet@a.cs.okstate.edu.*

(Received March 1996; revised April 1997)

Abstract

The problem of discriminating between two finite point sets in n-dimensional feature space by a separating plane that utilizes as few of the features as possible, is formulated as a mathematical program with a parametric objective function and linear constraints. The step function that appears in the objective function can be approximated by a sigmoid or by a concave exponential on the nonnegative real line, or it can be treated exactly by considering the equivalent linear program with equilibrium constraints (LPEC). Computational tests of these three approaches on publicly available real-world databases have been carried out and compared with an adaptation of the optimal brain damage (OBD) method for reducing neural network complexity. One feature selection algorithm via concave minimization (FSV) reduced cross-validation error on a cancer prognosis database by 35.4% while reducing problem features from 32 to 4.

Feature selection is an important problem in machine learning [18, 15, 16, 17, 33]. In its basic form the problem consists of eliminating as many of the features in a given problem as possible, while still carrying out a preassigned task with acceptable accuracy. Having a minimal number of features often leads to better generalization and simpler models that can be more easily interpreted. In the present work, our task is to discriminate between two given sets in an n-dimensional feature space by using as few of the given features as possible. We shall formulate this problem as a mathematical program with a parametric objective function that will attempt to achieve this task by generating a separating plane in a feature space of as small a dimension as possible while minimizing the average distance of misclassified points to the plane. One of the computational experiments that we carried out on our feature selection procedure showed its effectiveness, not only in minimizing the number of features selected, but also in quickly recognizing and removing spurious random features that were introduced. Thus, on the Wisconsin Prognosis Breast Cancer WPBC database [36] with a feature space of 32 dimensions and 6 random features added, one of our algorithms FSV (11) immediately removed the 6 random features as well as 28 of the original features resulting in a separating plane in a 4-dimensional reduced feature space. By using tenfold cross-validation [35], separation error in the 4-dimensional space was reduced 35.4% from the corresponding error in the original problem space. (See Section 3 for details.)

We note that mathematical programming approaches to the feature selection problem have been recently proposed in [4, 22]. Even though the approach of [4] is based on an LPEC formulation, both the LPEC and its method of solution are different from the ones used here. The polyhedral concave minimization approach of [22] is principally involved with theoretical considerations of one specific algorithm and no cross-validatory results are given. Other effective computational applications of mathematical programming to neural networks are given in [30, 26].

We outline now the contents of the paper. In Section 1, we formulate the feature selection problem as a mathematical program with a parametric objective function. Three methods of solution for this problem are proposed: a bilinear function minimization over a polyhedral set, approximation of the step function in the objective function by a sigmoid function and by a concave exponential on the nonnegative real line. Section 2 gives brief descriptions of the three algorithms. Section 3 contains numerical test results as well as comparisons with an adaptation of the optimal brain damage (OBD) method [18] for reducing complexity of a neural network. The adaptation consists of using directional derivatives at a solution point of (4), rather than second derivatives, which do not exist for the piecewise-linear objective function of (4). All our feature selection algorithms easily get rid of random features as well as unimportant features with resulting ten-fold cross-validation error reduction of as high as 35.4% with corresponding feature reduction of 88.2%. A concluding Section 4 ends the paper.

0.1 Notation

A word about our notation now. The symbol ":=" denotes definition. All vectors will be column vectors unless transposed by the superscript T to a row vector. For a vector x in the n-dimensional real space R^n, x_+ will denote the vector in R^n with components $(x_+)_i := \max\{x_i, 0\}$, $i = 1, \ldots, n$. Similarly x_* will denote the vector in R^n with components $(x_*)_i := (x_i)_*$, $i = 1, \ldots, n$, where $(\cdot)_*$ is the step function defined as one for positive x_i and zero otherwise, while $|x|$ will denote a vector of absolute values of components of x. The base of the natural logarithm will be denoted by ε and for $y \in R^m$, ε^{-y} will denote a vector in R^m with component ε^{-y_i}, $i = 1, \ldots, m$. The norm $\|\cdot\|_p$ will denote the p norm, $1 \leq p \leq \infty$, while $A \in R^{m \times n}$ will signify a real $m \times n$ matrix. For such a matrix, A^T will denote the transpose, and A_i will denote row i. For two vectors x and y in R^n, $x \perp y$ will denote $x^T y = 0$. A vector of ones in a real space of arbitrary dimension will be denoted by e. The notation $\arg\min\limits_{x \in S} f(x)$ will denote the set of minimizers of $f(x)$ on the set S. Similarly $\arg\,\text{vertex}\min\limits_{x \in S} f(x)$ will denote the set of vertex minimizers of $f(x)$ on the polyhedral set S, that is the set of vertices of S that solve $\min\limits_{x \in S} f(x)$. By a separating plane, with respect to two given point sets $\mathcal{A}$ and $\mathcal{B}$ in R^n, we shall mean a plane that attempts to separate R^n into two half spaces such that each open halfspace contains points mostly of $\mathcal{A}$ or $\mathcal{B}$. Alternatively, such a plane can also be interpreted as a classical perceptron [32, 12, 20]. Tenfold cross-validation [35] refers to the re-sampling method which successively removes 10% of the available data for testing a separator generated with the remaining 90%.

1 Feature Selection as a Mathematical Program

We consider two nonempty finite point sets $\mathcal{A}$ and $\mathcal{B}$ in R^n consisting of m and k points respectively that are represented by the matrices $A \in R^{m \times n}$ and $B \in R^{k \times n}$. The objective of the feature selection problem is to construct a separating plane:

$$P := \{x \mid x \in R^n,\ x^T w = \gamma\}, \tag{1}$$

with normal $w \in R^n$ and distance $\frac{|\gamma|}{\|w\|_2}$ to the origin, while suppressing as many of the components of w as possible. The separating plane P determines two open halfspaces, $\{x | x \in R^n,\ x^T w > \gamma\}$, containing mostly points belonging to $\mathcal{A}$, and $\{x | x \in R^n,\ x^T w < \gamma\}$, containing mostly points belonging to $\mathcal{B}$. That is we wish to satisfy

$$Aw > e\gamma, \ Bw < e\gamma \tag{2}$$

to the extent possible, or upon normalization

$$Aw \geq e\gamma + e, \ Bw \leq e\gamma - e \tag{3}$$

These conditions, (2) or equivalently (3), can be satisfied if and only if, the convex hulls of $\mathcal{A}$ and $\mathcal{B}$ do not intersect, which in general is not the case. We thus attempt to satisfy (3) in some approximate sense, for example, by minimizing some norm of the average violations of (3) such as

$$\min_{w,\gamma} \ f(w,\gamma) := \min_{w,\gamma} \ \frac{1}{m}\|(-Aw + e\gamma + e)_+\|_1 + \frac{1}{k}\,\|(Bw - e\gamma + e)_+\|_1 \tag{4}$$

Other [21] approximate ways of satisfying (3) consist of minimizing the number misclassified points by the plane P. However, this is a much harder NP-complete [6] formulation of the problem that we shall not employ here. Two principal reasons for using the 1-norm in (4) are:

(i) Problem (4) is reducible to a linear program (5) with many important theoretical properties that make it a very effective computational tool [2].

(ii) The 1-norm leads to insensitivity to outliers such as those resulting from distributions with pronounced tails and hence has a similar effect to that of robust regression [13],[11, pp 82-87].

Formulation (4) is equivalent to the following robust linear programming formulation proposed in [1] and utilized effectively to solve real-world problems [23]:

$$\min_{w,\gamma,y,z} \left\{ \frac{e^Ty}{m} + \frac{e^Tz}{k} \ \middle| \ -Aw + e\gamma + e \leq y, \ Bw - e\gamma + e \leq z, \ y \geq 0, \ z \geq 0 \right\} \tag{5}$$

In order to suppress as many of the components of w as possible we introduce an extra term with parameter $\lambda \in [0,1)$ into the objective of (5) while weighting the original function by $(1-\lambda)$ as follows:

$$\min_{w,\gamma,y,z} \left\{ (1-\lambda)(\frac{e^Ty}{m} + \frac{e^Tz}{k}) + \lambda e^T \mid w \mid_* \ \middle| \ \begin{array}{c} -Aw + e\gamma + e \leq y, \\ Bw - e\gamma + e \leq z, \\ y \geq 0, \ z \geq 0 \end{array} \right\}, \lambda \in [0,1) \tag{6}$$

This is equivalent to the following parametric program

$$\text{(FS)} \min_{w,\gamma,y,z,v} \left\{ (1-\lambda)(\frac{e^Ty}{m} + \frac{e^Tz}{k}) + \lambda e^T v_* \ \middle| \ \begin{array}{c} -Aw + e\gamma + e \leq y, \\ Bw - e\gamma + e \leq z, \\ y \geq 0, z \geq 0, \\ -v \leq w \leq v \end{array} \right\}, \ \lambda \in [0,1) \tag{7}$$

This is our fundamental **feature selection (FS) problem** that will be solved for a suitable value of the parameter $\lambda \in [0,1)$, that is a λ which gives the best cross-validated separation. Typically this will be achieved in a feature space of reduced dimensionality, that is $e^Tv_* < n$. Because of the discontinuity in the step function term e^Tv_*, the FS problem will be approximated by smoothing the step function v_* by using either the standard sigmoid function of neural networks [31, 12] or by using a concave exponential approximation on the nonnegative real line [22]. Thus, the two approximations of the step vector v_* of (7) by a sigmoid and a concave exponential are respectively:

$$v_* \cong s(v,\alpha) := \left(e + \varepsilon^{-\alpha v}\right)^{-1}, \ \alpha > 0 \tag{8}$$

$$v_* \cong t(v,\alpha) := e - \varepsilon^{-\alpha v}, \ \alpha > 0 \tag{9}$$

Here e is a vector of ones, ε is the base of natural logarithms, α is a positive parameter, and the application of either function to a vector is interpreted componentwise as in the standard MATLAB notation [25]. Advantages of the exponential (9) over the standard sigmoid (8) are its simplicity and concavity. These properties lead to a finitely terminating algorithm and a more accurate representation of the step function v_* at 0, because $t(0,\alpha) = 0$, whereas $s(0,\alpha) = \frac{1}{2}e$. With the approximations (8) and (9), the FS problem (7) can be written as the following FSS (FS sigmoid) and FSV (FS concave) problems:

$$\text{(FSS)} \min_{w,\gamma,y,z,v} \left\{ (1-\lambda)\left(\frac{e^T y}{m} + \frac{e^T z}{k}\right) + \lambda e^T (e + \varepsilon^{-\alpha v})^{-1} \;\middle|\; \begin{array}{c} -Aw + e\gamma + e \le y, \\ Bw - e\gamma + e \le z, \\ y \ge 0, z \ge 0, \\ -v \le w \le v \end{array} \right\}, \ \lambda \in [0,1) \tag{10}$$

$$\text{(FSV)} \min_{w,\gamma,y,z,v} \left\{ (1-\lambda)\left(\frac{e^T y}{m} + \frac{e^T z}{k}\right) + \lambda (n - e^T \varepsilon^{-\alpha v}) \;\middle|\; \begin{array}{c} -Aw + e\gamma + e \le y, \\ Bw - e\gamma + e \le z, \\ y \ge 0, z \ge 0, \\ -v \le w \le v \end{array} \right\}, \ \lambda \in [0,1) \tag{11}$$

We now make use of the following lemma [22, Lemma 2.3] in order to get an exact reformulation of (7), without the step function v_*, as a linear program with equilibrium constraints (LPEC) [21, 19].

Lemma 1.1 *Let $a \in R^m$. Then*

$$r = a_*, \ u = a_+ \Leftrightarrow (r,u) = \arg\min_{r,u} \{ e^T r \mid 0 \le r \perp u - a \ge 0, \ 0 \le u \perp -r + e \ge 0 \} \tag{12}$$

With this lemma the FS problem can be written in the completely equivalent form of a linear program with equilibrium constraints (LPEC) as follows. (The "equilibrium" terminology comes from the term involving the $\perp$-condition in the constraints which characterizes complementarity problems [8, 24].)

$$\text{(FSL)} \min_{w,\gamma,y,z,v} \left\{ (1-\lambda)\left(\frac{e^T y}{m} + \frac{e^T z}{k}\right) + \lambda e^T r \;\middle|\; \begin{array}{c} -Aw + e\gamma + e \le y, \\ Bw - e\gamma + e \le z, \\ y \ge 0, z \ge 0, \\ -v \le w \le v, \\ 0 \le r \perp u - v \ge 0 \\ 0 \le u \perp -r + e \ge 0 \end{array} \right\}, \ \lambda \in [0,1) \tag{13}$$

This LPEC reformulation of the FS problem (7) has a number of important consequences, such as existence of a solution to the FS problem (7) [22, Proposition 2.5]. However FSL (13) is difficult to solve computationally, and in fact LPECs are NP-hard in general, because they subsume the general linear complementarity problem which is NP-complete [7]. To get around this difficulty we reformulate (13) as a parametric bilinear program that can be easily processed by solving a finite succession of linear programs that terminate at a stationary point [3, Algorithm 2.1]. In particular, the nonnegative nonlinear terms $r^T(u-v)$ plus $u^T(-r+e)$ are moved as positive penalty terms into the objective function as $-r^Tv + e^Tu$ with penalty parameter $\mu \in (0,1)$ as follows:

$$\text{(FSB)} \quad \min_{w,\gamma,y,z,v,u,r} \left\{ \begin{array}{c|c} (1-\mu)((1-\lambda)(\frac{e^Ty}{m} + \frac{e^Tz}{k}) + \lambda e^T r) & \begin{array}{c} -Aw + e\gamma + e \le y, \\ Bw - e\gamma + e \le z, \\ y \ge 0, z \ge 0, \\ -v \le w \le v, \\ 0 \le r, u - v \ge 0 \\ 0 \le u, -r + e \ge 0 \end{array} \\ +\mu(-r^Tv + e^Tu) & \end{array} \right\}, \lambda \in [0,1), \mu \in (0,1) \tag{14}$$

In the next section, we will briefly describe effective computational algorithms for solving each of the three feature selection problem reformulations: FSS (10), FSV (11) and FSB (14), and will test these algorithms in the subsequent section on some public real-world databases and compare them with an adaptation of the OBD method [18].

2 Algorithms for the Feature Selection Problem

By replacing the variables (w, γ) by the nonnegative variables (w^1, γ^1, ζ^1) using the standard transformation $w = w^1 - e\zeta^1$, $\gamma = \gamma^1 - \zeta^1$, the problems FSS (10), FSV (11) and FSB (14) can be transformed to a minimization problem of the following form:

$$\min_x \{f(x) \mid Ax \le b, \ x \ge 0\}, \tag{15}$$

where $f: R^\ell \to R$, is a differentiable, nonconvex function bounded below on the nonempty polyhedral feasible region of (15), $A \in R^{p\times\ell}$ and $b \in R^p$. Although this transformation is needed in order to establish theoretically the finiteness of our algorithms (the SLA 2.1 and the Bilinear Algorithm 2.3 below), we shall not carry it out in the interest of keeping our notation simple. In fact, all our computational implementations terminated finitely without this transformation.

We first consider the FSS problem (10). Because its objective has no convexity or concavity properties, we prescribe the locally fast iterative quadratic programming (IQP) algorithm [9, 10] which is the equivalent of a Newton or quasi-Newton method, and consists of taking a quadratic approximation of the objective function of (15) and solving the resulting quadratic program for a potential next iterate, which becomes the next iterate if it is the best point along the line joining it to the current iterate, else the best feasible point along that line becomes the next iterate. This algorithm has been implemented professionally using powerful linear algebra packages in the MINOS software package [28]. Thus, MINOS was used to solve the FSS problem (10).

We now consider the FSV problem (11). Because its objective function is a differentiable concave function, and has a vertex solution in the formulation (15), we use a stepless successive linear approximation algorithm. The algorithm solves a finite sequence of linear programs and terminates at a stationary point [22]. We outline the algorithm now.

2.1 Successive Linearization Algorithm (SLA) for FSV (11). Choose $\lambda \in [0,1)$. Start with a random $(w^0, \gamma^0, y^0, z^0, v^0)$. Having $(w^i, \gamma^i, y^i, z^i, v^i)$ determine the next iterate by solving the linear program:

$$(w^{i+1},\gamma^{i+1},y^{i+1},z^{i+1},v^{i+1}) \in \arg\text{vertex}\min_{w,\gamma,y,z,v}\left\{\begin{array}{c|c} (1-\lambda)(\frac{e^Ty}{m}+\frac{e^Tz}{k}) & \begin{array}{c}-Aw+e\gamma+e\leq y,\\ Bw-e\gamma+e\leq z,\\ y\geq 0,\ z\geq 0,\\ -v\leq w\leq v\end{array} \\ +\lambda\alpha(\varepsilon^{-\alpha v^i})^T(v-v^i) & \end{array}\right\} \tag{16}$$

Stop if $(w^i,\gamma^i,y^i,z^i,v^i)$ is feasible and

$$(1-\lambda)(\frac{e^T(y^{i+1}-y^i)}{m}+\frac{e^T(z^{i+1}-z^i)}{k})+\lambda\alpha(\varepsilon^{-\alpha v^i})^T(v^{i+1}-v^i)=0 \tag{17}$$

It can be shown [22, Theorem 4.2] that this algorithm terminates in a finite number of steps at a stationary point, which may be a global solution as well. We state this result as follows.

2.2 SLA Finite Termination for FSV (11). The iterates determined by (16) generate a strictly decreasing sequence of objective function values for the FSV problem (11) and terminate at iteration $\bar{i}$ with a stationary point (which may also be global minimum solution) that satisfies the following necessary optimality criterion.

$$(1-\lambda)(\frac{e^T}{m}(y-y^{\bar{i}})+\frac{e^T}{k}(z-z^{\bar{i}}))+\lambda\alpha(\varepsilon^{-\alpha v^{\bar{i}}})^T(v-v^{\bar{i}})\geq 0,\quad \forall \text{ feasible } (w,\gamma,y,z,v) \tag{18}$$

We now consider the FSB problem (14). Because of the bilinear nature of its objective function, the simple, but fast, bilinear algorithm of [3, Algorithm 2.1] can be applied to (14) as follows.

2.3 Bilinear Algorithm (BA) for FSB (14). Choose $\lambda\in[0,1)$, $\mu\in(0,1)$. Start with any feasible $(w^0,\gamma^0,y^0,z^0,v^0,r^0,u^0)$ to problem (14). Having $(w^i,\gamma^i,y^i,z^i,v^i,r^i,u^i)$ determine the next iterate by solving the two linear programs (in fact, one linear program need be solved, because the first one can be solved in closed form as indicated below):

$$r^{i+1}\in\arg\min_r\{(1-\mu)\lambda e^Tr-\mu v^ir \mid 0\leq r\leq e\}\supset\{(-(1-\mu)\lambda e+\mu v^i)_*\}$$

$$(w^{i+1},\gamma^{i+1},y^{i+1},z^{i+1},v^{i+1},u^{i+1})\in\arg\text{vertex}\min_{w,\gamma,y,z,v,u}\left\{\begin{array}{c|c} (1-\mu)(1-\lambda)(\frac{e^Ty}{m}+\frac{e^Tz}{k}) & \begin{array}{c}-Aw+e\gamma+e\leq y,\\ Bw-e\gamma+e\leq z,\\ y\geq 0, z\geq 0,\\ -v\leq w\leq v\\ u\geq 0,\ u-v\geq 0\end{array} \\ +\mu(-{r^{i+1}}^Tv+e^Tu) & \end{array}\right\} \tag{19}$$

Stop when:

$$(1-\mu)((1-\lambda)(\frac{e^T}{m}(y^{i+1}-y^i)+\frac{e^T}{k}(z^{i+1}-z^i))+\lambda e^T(r^{i+1}-r^i))+\mu(-{r^{i+1}}^Tv^{i+1}+{r^i}^Tv^i+e^T(u^{i+1}-u^i))=0 \tag{20}$$

Comment: The parameter λ was chosen in the set $\{0,\ 0.05,\ 0.10,\ \ldots,\ 1.00\}$, with the desired λ being the one achieving the best cross-validated separation. The parameter μ was chosen as the smallest in the set $\{0.05, 0.15, 0.25, \ldots, 0.95\}$ such that the following complementarity condition holds:

$$-{r^{i+1}}^Tv^{i+1}+e^Tu^{i+1}={r^{i+1}}^T(u^{i+1}-v^{i+1})+{u^{i+1}}^T(r^{i+1}+e)=0$$

.

The following finite termination theorem follows from [3, Theorem 2.1].

2.4 Finite Termination of Bilinear Algorithm for FSB (14). For a fixed $\lambda \in [0,1)$, $\mu \in (0,1)$, the Bilinear Algorithm 2.3 terminates in a finite number of steps at a global solution or a stationary point $(w^i, \gamma^i, y^i, z^i, v^i, u^i, r^{i+1})$ that satisfies the following necessary optimality condition:

$$\begin{array}{c} (1-\mu)((1-\lambda)(\frac{e^T}{m}(y-y^i)+\frac{e^T}{k}(z-z^i))+\lambda e^T(r-r^{i+1}))+ \\ \mu(-{r^{i+1}}^T(v-v^i)-(r-r^{i+1})^Tv^i+e^T(u-u^i))\geq 0, \quad \forall \text{ feasible } (w,\gamma,y,z,v,u,r) \end{array} \tag{21}$$

Before turning our attention to computational testing of the proposed algorithms we discuss some of their properties. Both the FSS and FSV Algorithms start with the same linear programming formulation (5) of the linear separability problem and add a feature-squashing term $e^T|w|_*$ which is then smoothed by a sigmoid function for the FSS Algorithm and by an exponential for the FSV Algorithm. If we attempt to model exactly the discontinuous squashing term $e^T|w|_*$ we arrive at the FSL model (13) with equilibrium constraints and the bilinear model (14) of the FSB Algorithm. Another algorithm, the OBD Algorithm 3.1, patterned after the Optimal Brain Damage method [18], looks at directional derivatives of the nondifferentiable objective function of the linear separation problem (4) at a solution point, and squashes features with directional derivatives less than a certain tolerance.

3 Computational Test Results

We report in this section on numerical tests carried out using the three proposed formulations of the feature selection problem: FSS (10), FSV (11), and FSB (14), as well as an adaptation of the optimal brain damage (OBD) method [18] to our nondifferentiable problem (4) that we outline in the next paragraph. For each of the FSS, FSV and FSB algorithms, once the features have been selected by the algorithm, the linear program (4) is re-solved with all the non-selected features set to zero.

Because the objective function of our separating plane problem (4) is piecewise-linear, the second derivatives needed by the OBD method do not exist. However, at a solution point $(\bar{w}, \bar{\gamma})$ of (4), obtained by solving the linear program (5), even though a subgradient is zero [29, Theorem 1, p. 133], the directional derivatives of $f(w, \gamma)$ of (4) in the directions of the components $\pm w_i$ of w are generally not zero. An OBD procedure for this case can be implemented by examining these directional derivatives in the $2n$ directions $(\pm w_1, \ldots, \pm w_n)$ at the point $(\bar{w}, \bar{\gamma})$, suppressing those features x_i corresponding to w_i for which the directional derivatives in the directions $\pm w_i$ are less than some tolerance, and then re-solving the linear program (5) with these w_i set to zero. We summarize our version of the OBD algorithm [18] for problem (4) as follows.

3.1 Optimal Brain Damage (OBD) Algorithm for (4). Choose small positive numbers δ_1 and δ_2.

(i) Solve (4) for $(\bar{w}, \bar{\gamma})$ or equivalently the linear program (5) for $(\bar{w}, \bar{\gamma}, \bar{y}, \bar{z})$.

(ii) Set $w_i = 0$ for $i \in I$ and re-solve (4) or (5) where:

$$I := \{i \Big| \mid f(\bar{w}_1, \ldots, \bar{w}_{i-1}, \bar{w}_i \pm \delta_1, \bar{w}_{i+1}, \ldots, \bar{w}_n, \bar{\gamma}) - f(\bar{w}, \bar{\gamma}) \mid \leq \delta_2, \ i = 1, \ldots, n\},$$

where $f(w, \gamma)$ is the objective function of (4).

Figures 1 to 8 and Tables 1 to 3 summarize our results for two problems: the Wisconsin Prognostic Breast Cancer (WPBC) problem [36] and the Ionosphere problem [34, 27]. The 32-feature WPBC problem has 28 points in category one of breast cancer patients for which the cancer recurred within 24 months and category two of 119 patients for which the cancer did not recur within 24 months. Several of the features are correlated and hence provide redundant information. Each of the six random features for the WPBC dataset were generated first by selecting a random number from a uniform distribution on $[0,1]$, multiplying it by 3500 (to get the correct scaling), rounding it down to the closest integer, then adding 1. Thus the resulting random feature was a random integer in the range $[1,3500]$. The 34-feature Ionosphere problem has 225 points of radar returns from the ionosphere in one category and 126 points in the other. The 34 original problem features were augmented with 6 random features sampled from a uniform distribution on $[-1,1]$. Each problem was solved by the four feature selection algorithms: FSS (10), FSV (11), FSB (14) and OBD 4.1, using the GAMS general algebraic and modeling system [5]. GAMS was interfaced with the OSL simplex solver [14] on a Sun SparcStation 20, in the last three algorithms and with MINOS [28] in the first algorithm.

Each of Figures 1 to 8 plots the average correctness (percentage of correctly classified points) of ten cross-validation runs versus the number of features retained by each feature selection algorithm. The number of features retained in the FSS, FSV and FSB algorithms is controlled by the size of λ, with $\lambda = 0$ retaining all features, and $\lambda = 1$ suppressing all features. For the OBD algorithm, $\delta_2 = 0$ retains most features while a large enough δ_2 suppresses all features. All of the FS algorithms eliminated the random features at very small values of the parameter λ, whereas the OBD algorithm eliminated them at sufficiently large value of δ_2. This indicates that our algorithms can easily discover and discard irrelevant features. Also, an extensive comparative study of tenfold cross-validation results on data including artificially correlated features, may indicate the relative effectiveness of the various proposed algorithms.

Because the underlying feature selection problem is nonconvex, there is no easy way of obtaining a global solution to it. Our approach has been to start with an initial vector with elements sampled from a uniform distribution on $[-1,1]$ and then apply the various proposed iterative algorithms. In some instances, starting with different random initial points and taking the best of the final solutions may be as good or a better strategy.

Tables 1 and 2 tabulate the maximum reduction in the cross-validation error when feature selection is used compared to the case without feature selection, that is $\lambda = 0$. The maximum reduction was obtained by varying λ in steps of 0.05 in [0,1), for the FSS, FSV and FSB algorithms, and choosing the case with the best accuracy. For the OBD algorithm, the maximal reduction was similarly obtained by varying δ_2 as a multiple $\nu \in [0,1)$ of the maximal directional derivative of the objective function of (4), at a solution point, in the w coordinate directions. For all OBD runs, $\delta_1 = 0.01$. Corresponding to each maximal reduction, the percentage reduction in the number of features relative to the run without feature selection is also given.

Table 3 gives times in seconds for a typical case of $\lambda = 0.05$ for the three FSS, FSV and FSB algorithms and for the OBD algorithm with $\delta_1 = 0.01$ and δ_2 equal to 0.05 times the maximal directional derivative of the objective function of (4), at a solution point, in the w coordinate directions.

We make the following observations on our numerical results:

1. All algorithms obtained a reduction in error with selected features fewer in number than the number of original features.

2. Cross-validation error reduction in the prognosis problem was as high as 35.4% with a corre-

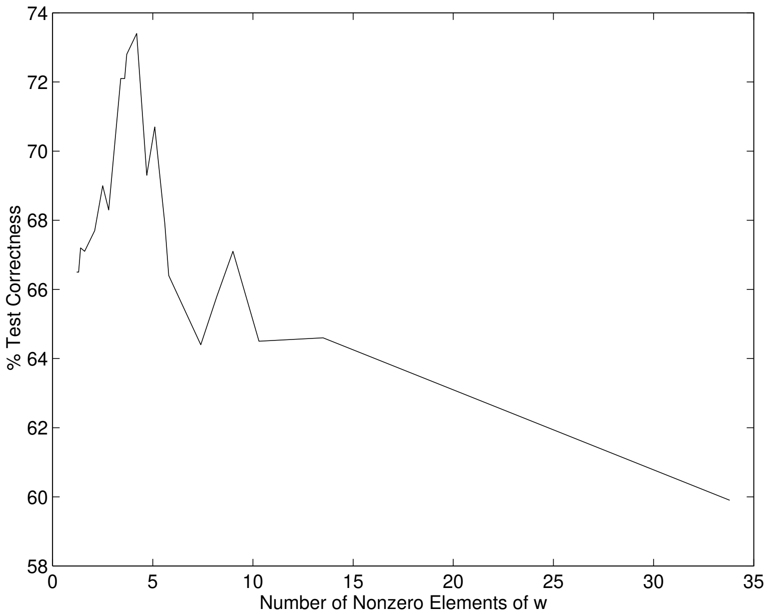

Figure 1: **Prognosis Problem: Tenfold cross-validation correctness versus number of features selected by the FSS Algorithm (10)**

sponding reduction of features of 88.2%, obtained by the FSV algorithm (10).

3. Although cross-validation error reduction for the Ionosphere Problem was only as high as 19.1% in the FSS Algorithm, the corresponding reduction in the number of features used was 50.2%. On the other hand the OBD Algorithm had the highest feature reduction of 65.5% on the Ionosphere problem but the lowest cross-validation error reduction, 9.2%.

4. Typical run times show that the FSV and OBD algorithms are the fastest because the FSV solves a few linear programs, typically three, and the OBD solves two linear programs only.

4 Conclusion

We have proposed a novel approach for feature selection by introducing a parametric objective function in a mathematical program that attempts to separate data points by utilizing as few of the features as possible. We have also given a new and extremely simple realization of the complexity-reducing Optimal Brain Damage procedure for our problem that consists of solving two linear programs. All proposed algorithms are effective in eliminating unimportant and redundant features while reducing cross-validation error. This will enhance generalization and lead to models that are more easily interpreted for many learning problems. Because of its simplicity, speed and effectiveness, we believe that our feature selection approach should be incorporated into any linear separation program. In particular the finitely terminating FSV Algorithm, which had one of the best feature reduction records and solution times, can be seamlessly added to the fundamental robust linear programming formulation (5).

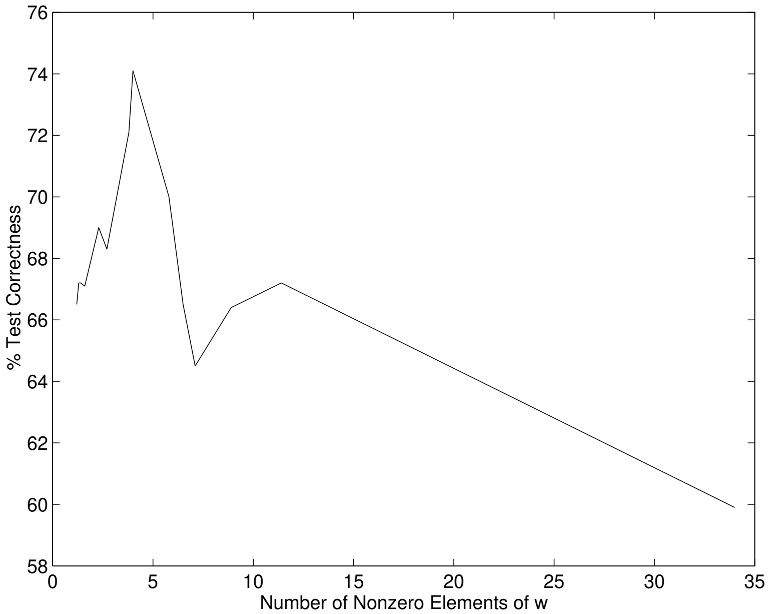

Figure 2: **Prognosis Problem: Tenfold cross-validation correctness versus number of features selected by the FSV Algorithm (11)**

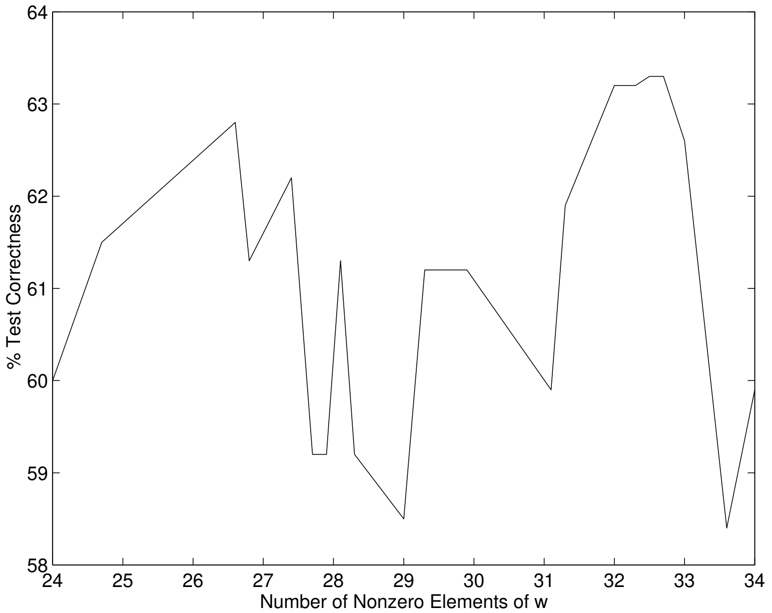

Figure 3: **Prognosis Problem: Tenfold cross-validation correctness versus number of features selected by the FSB Algorithm (14)**

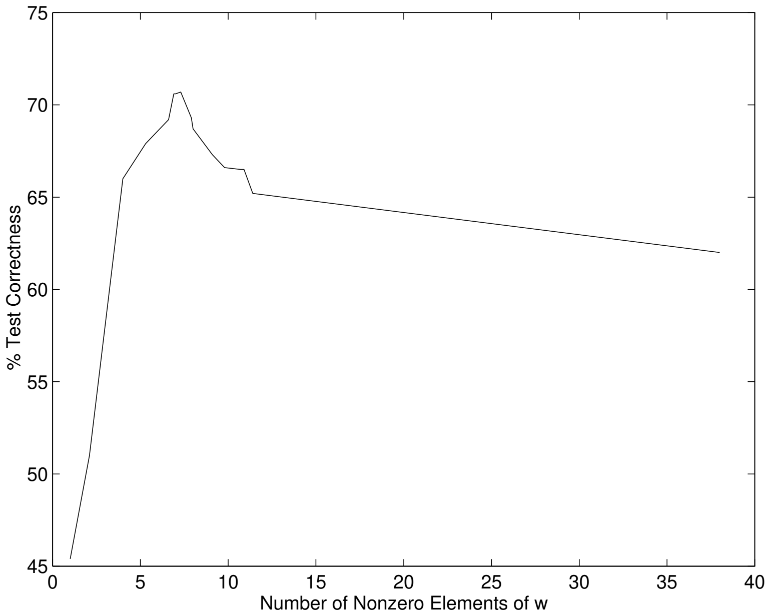

Figure 4: **Prognosis Problem: Tenfold cross-validation correctness versus number of features selected by the OBD Algorithm 4.1**

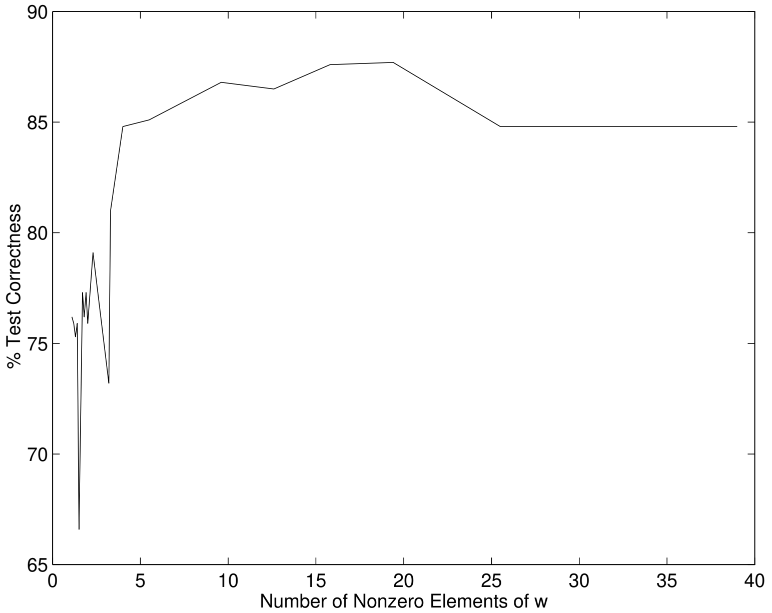

Figure 5: **Ionosphere Problem: Tenfold cross-validation correctness versus number of features selected by the FSS Algorithm (10)**

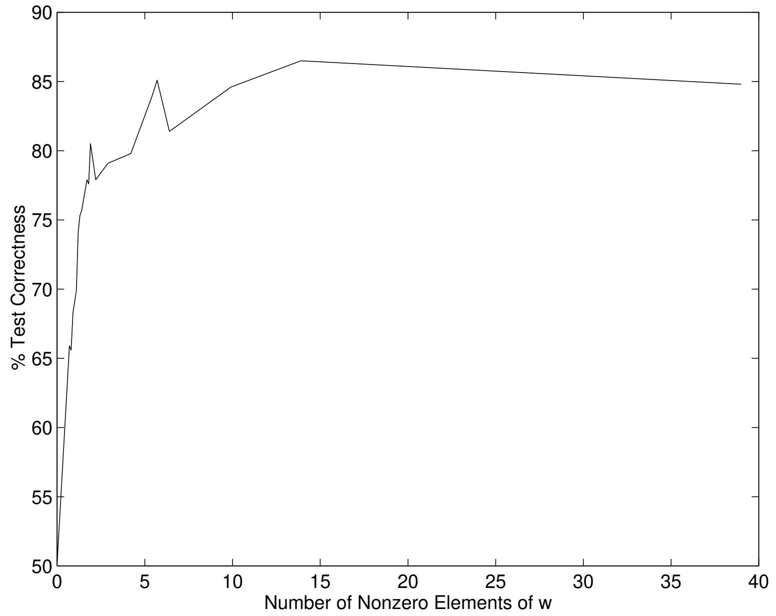

Figure 6: **Ionosphere Problem: Tenfold cross-validation correctness versus number of features selected by the FSV Algorithm (11)**

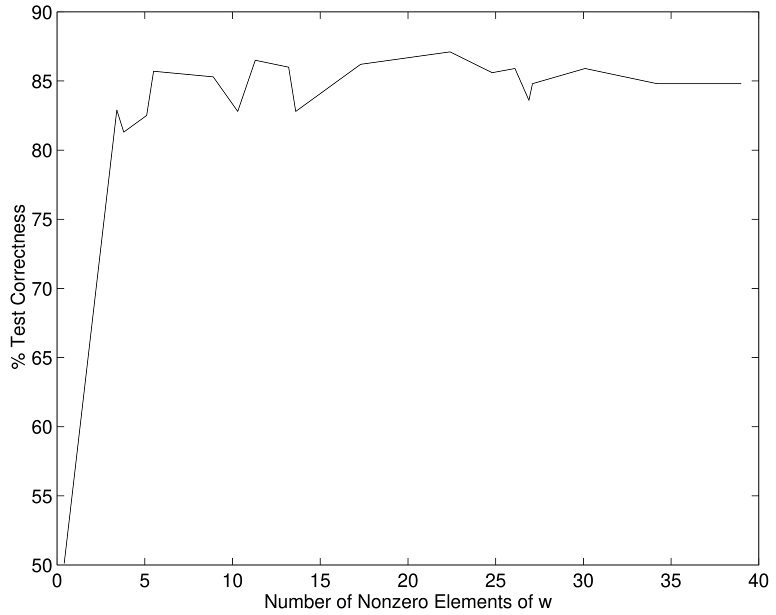

Figure 7: **Ionosphere Problem: Tenfold cross-validation correctness versus number of features selected by the FSB Algorithm (14)**

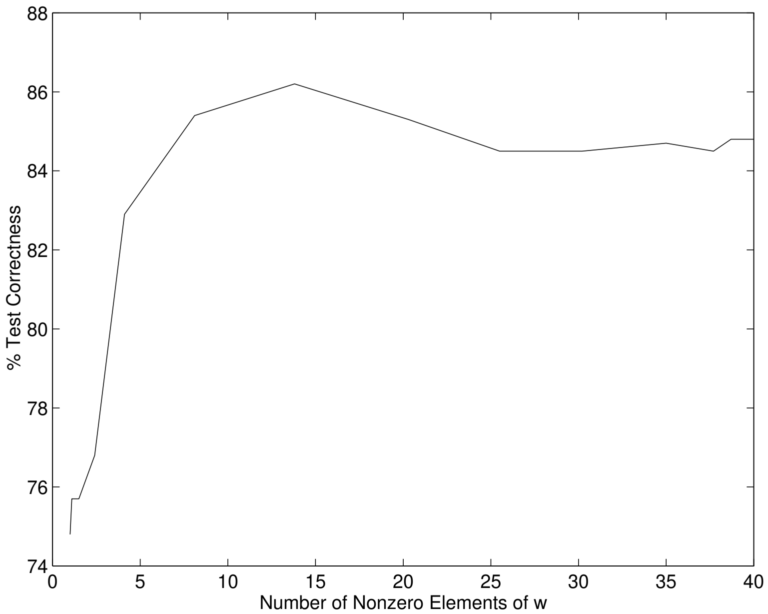

Figure 8: **Ionosphere Problem: Tenfold cross-validation correctness versus number of features selected by the OBD Algorithm 4.1**

	FSS	FSV	FSB	OBD
Max Cross-Validation Error Reduction	33.7%	35.4%	8.5%	22.9%
Corresponding Feature Reduction	87.6%	88.2%	4.4%	80.8%

Table 1: Maximum reduction in cross-validation error and corresponding feature reduction for the Prognosis Problem using each of four algorithms.

	FSS	FSV	FSB	OBD
Max Cross-Validation Error Reduction	19.1%	11.2%	15.1%	9.2%
Corresponding Feature Reduction	50.2%	64.4%	42.6%	65.5%

Table 2: Maximum reduction in cross-validation error and corresponding feature reduction for the Ionosphere Problem using each of four algorithms.

	FSS	FSV	FSB	OBD
Prognosis	3.578	3.298	25.965	2.423
Ionosphere	38.571	11.895	23.630	15.946

Table 3: Solution times in seconds for the four feature selection algorithms for the Prognosis and Ionosphere Problems

Acknowledgement

We are indebted to Jude Shavlik for suggesting comparisons with the Optimal Brain Damage method [18]. This material is based on research supported by Air Force Office of Scientific Research Grants F49620-94-1-0036, F49620-97-1-0326, National Science Foundation Grant CCR-9322479 and National Institutes of Health INRSA Fellowship 1 F32 CA 68690-01.

References

1. K. P. BENNETT and O. L. MANGASARIAN, 1992. Neural Network Training via Linear Programming, in *Advances in Optimization and Parallel Computing*, P. M. Pardalos (ed.), North Holland, Amsterdam, 56–67.

2. K. P. BENNETT and O. L. MANGASARIAN, 1992. Robust Linear Programming Discrimination of Two Linearly Inseparable Sets, *Optimization Methods and Software 1*, 23–34.

3. K. P. BENNETT and O. L. MANGASARIAN, 1993. Bilinear Separation of Two Sets in n-space, *Computational Optimization & Applications 2*, 207–227.

4. E. J. BREDENSTEINER and K. P. BENNETT, 1995. Feature Minimization within Decision Trees, Department of Mathematical Sciences Math Report No. 218, Rensselaer Polytechnic Institute, Troy, NY 12180.

5. A. BROOKE, D. KENDRICK, and A. MEERAUS, 1988. *GAMS: A User's Guide*, The Scientific Press, South San Francisco.

6. CHUNHUI CHEN and O. L. MANGASARIAN, 1995. Hybrid Misclassification Minimization, Technical Report 95-05, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin. *Advances in Computational Mathematics,* to appear. Available by ftp://ftp.cs.wisc.edu/math-prog/tech-reports/95-05.ps.Z.

7. S.-J. CHUNG, 1989. NP-Completeness of the Linear Complementarity Problem, *Journal of Optimization Theory and Applications 60*, 393–399.

8. R. W. COTTLE, J.-S. PANG, and R. E. STONE, 1992. *The Linear Complementarity Problem*, Academic Press, New York.

9. U. M. GARCIA PALOMARES and O. L. MANGASARIAN, 1976. Superlinearly Convergent Quasi–Newton Algorithms for Nonlinearly Constrained Optimization Problems, *Mathematical Programming 11*, 1–13.

10. S.-P. HAN, 1976. Superlinearly Convergent Variable Metric Algorithms for General Nonlinear Programming Problems, *Mathematical Programming 11*, 263–282.

11. M. H. HASSOUN, 1995. *Fundamentals of Artificial Neural Networks*, MIT Press, Cambridge, MA.

12. J. HERTZ, A. KROGH, and R. G. PALMER, 1991. *Introduction to the Theory of Neural Computation*, Addison-Wesley, Redwood City, CA.

13. P. J. HUBER, 1981. *Robust Statistics*, John Wiley, New York.

14. IBM OPTIMIZATION SUBROUTINE LIBRARY, 1994. *GAMS - The Solver Manuals: OSL*, GAMS Development Corporation, Washington, D.C.

15. G. H. JOHN, R. KOHAVI, and K. PFLEGER, 1994. Irrelevant Features and the Subset Selection Problem, in *Proceedings of the 11th International Conference on Machine Learning*, Morgan Kaufmann, San Mateo, CA.

16. K. KIRA and L. RENDELL, 1992. The Feature Selection Problem: Traditional Methods and a New Algorithm, in *Proceedings of the Tenth National Conference on Artificial Intelligence*, Morgan Kaufmann, San Mateo, CA, 129–134.

17. J. KITTLER, 1986. Feature Selection and Extraction, in *Handbook of Pattern Recognition and Image Processing*, Young & Fu (eds.), Academic Press, New York.

18. Y. LE CUN, J. S. DENKER, and S. A. SOLLA, 1990. Optimal Brain Damage, in *Advances in Neural Information Processing Systems II (Denver 1989)*, D. S. Touretzky (ed.), Morgan Kaufmann, San Mateo, CA, 598–605.

19. Z.-Q. LUO, J.-S. PANG, D. RALPH, and S.-Q. WU, 1996. Mathematical Programs with Equilibrium Constraints, *Mathematical Programming 75*, 19–76.

20. O. L. MANGASARIAN, 1993. Mathematical Programming in Neural Networks, *ORSA Journal on Computing 5*, 349–360.

21. O. L. MANGASARIAN, 1994. Misclassification Minimization, *Journal of Global Optimization 5*, 309–323.

22. O. L. MANGASARIAN, 1996. Machine Learning via Polyhedral Concave Minimization, in *Applied Mathematics and Parallel Computing - Festschrift for Klaus Ritter*, H. Fischer, B. Riedmueller, and S. Schaeffler (eds.), Physica-Verlag A Springer-Verlag Company, Heidelberg. Available by ftp://ftp.cs.wisc.edu/math-prog/tech-reports/95-20.ps.Z.

23. O. L. MANGASARIAN, W. N. STREET, and W. H. WOLBERG, 1995. Breast Cancer Diagnosis and Prognosis via Linear Programming, *Operations Research 43*, 570–577.

24. L. MATHIESEN, 1987. An Algorithm Based on a Sequence of Linear Complementarity Problems Applied to a Walrasian Equilibrium Model: An Example, *Mathematical Programming 37*, 1–18.

25. MATHWORKS, INC., 1991. *PRO-MATLAB for UNIX Computers*, The Mathworks, Inc., South Natick, MA 01760.

26. S. MUKHOPADHYAY, A. ROY, and S. GOVIL, 1993. A Polynomial Time Algorithm for Generating Neural Networks for Pattern Classification: Its Stability Properties and Some Test Results, *Neural Computation 5*, 317–330.

27. P. M. MURPHY and D. W. AHA, 1992. UCI Repository of Machine Learning Databases, Department of Information and Computer Science, University of California, Irvine, www.ics.uci.edu/AI/ML/MLDBRepository.html.

28. B. A. MURTAGH and M. A. SAUNDERS, 1983, 1992. MINOS 5.0 User's Guide, Technical Report SOL 83.20, Stanford University. MINOS 5.4 Release Notes.

29. B. T. POLYAK, 1987. *Introduction to Optimization*, Optimization Software, Inc., Publications Division, New York.

30. A. ROY, L. S. KIM, and S. MUKHOPADHYAY, 1993. A Polynomial Time Algorithm for the Construction and Training of a Class of Multilayer Perceptrons, *Neural Networks 6*, 535–545.

31. D. E. RUMELHART, G. E. HINTON, and R. J. WILLIAMS, 1986. Learning Internal Representations by Error Propagation, in *Parallel Distributed Processing*, D. E. Rumelhard and J. L. McClelland (eds.), MIT Press, Cambridge, MA.

32. D. E. RUMELHART and J. L. MCCLELLAND, 1986. *Parallel Distributed Processing*, MIT Press, Cambridge, MA.

33. W. SIEDLECKI and J. SKLANSKY, 1988. On Automatic Feature Selection, *International Journal of Pattern Recognition and Artificial Intelligence 2*, 197–220.

34. V. G. SIGILLITO, S. P. WING, L. V. HUTTON, and K. B. BAKER, 1989. Classification of Radar Returns from the Ionosphere Using Neural Networks. APL Technical Digest 10, Johns Hopkins University, Baltimore, MD.

35. M. STONE, 1974. Cross-validatory Choice and Assessment of Statistical Predictions, *Journal of the Royal Statistical Society 36*, 111-147.

36. W. H. WOLBERG, W. N. STREET, and O. L. MANGASARIAN, 1995. WPBC: Wisconsin Prognostic Breast Cancer Database, Computer Sciences Department, University of Wisconsin, Madison. ftp://ftp.cs.wisc.edu/math-prog/cpo-dataset/machine-learn/WPBC/