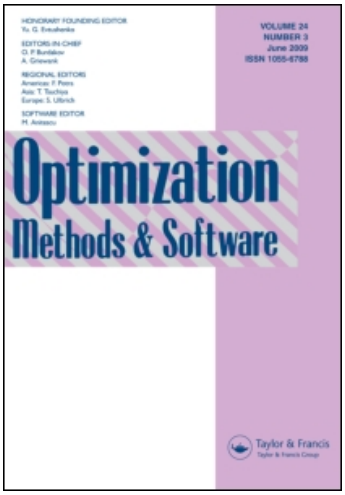

Optimization Methods and Software
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713645924

Feature selection combining linear support vector machines and concave optimization

F. Rinaldi [a]; M. Sciandrone [a]
[a] Dipartimento di Informatica e Sistemistica, Sapienza Università di Roma, Roma, Italy

To cite this Article Rinaldi, F. and Sciandrone, M.(2010) 'Feature selection combining linear support vector machines and concave optimization', Optimization Methods and Software, 25: 1, 117 — 128
To link to this Article: DOI: 10.1080/10556780903139388
URL: http://dx.doi.org/10.1080/10556780903139388

Optimization Methods & Software
Vol. 25, No. 1, February 2010, 117–128

Taylor & Francis
Taylor & Francis Group

Feature selection combining linear support vector machines and concave optimization

F. Rinaldi[a]* and M. Sciandrone[a]

[a]*Dipartimento di Informatica e Sistemistica, Sapienza Università di Roma, via Ariosto, 25, 00185 Roma, Italy;* [b]*Dipartimento di Sistemi e Informatica, Università di Firenze, Via di S. Marta 3, 50139 Firenze, Italy*

(*Received 28 November 2008; final version received 3 June 2009*)

In this work we consider feature selection for two-class linear models, a challenging task arising in several real-world applications. Given an unknown functional dependency that assigns a given input to the class to which it belongs, and that can be modelled by a linear machine, we aim to find the relevant features of the input space, namely we aim to detect the smallest number of input variables while granting no loss in classification accuracy. Our main motivation lies in the fact that the detection of the relevant features provides a better understanding of the underlying phenomenon, and this can be of great interest in important fields such as medicine and biology. Feature selection involves two competing objectives: the prediction capability (to be maximized) of the linear classifier and the number of features (to be minimized) employed by the classifier. In order to take into account both the objectives, we propose a feature selection strategy based on the combination of support vector machines (for obtaining good classifiers) with a concave optimization approach (for finding sparse solutions). We report results of an extensive computational experience showing the efficiency of the proposed methodology.

Keywords: support vector machines; zero-norm; concave programming; Frank-Wolfe method

1. Introduction

Obtaining a reliable predictor is a key aspect when dealing with machine learning problems. If data are noisy or there is much irrelevant and redundant information, then knowledge discovery during the training phase becomes more difficult. Data preprocessing, which represents a fundamental step in data mining, transforms raw data into a more suitable format for easing the training task. There are a number of different tools and methods used for preprocessing, including: data denoising, integration, transformation, normalization, feature extraction, and feature selection. In this work, we focus on a specific class of feature selection problems.

Given an unknown process that assigns a given input to the class to which it belongs, feature selection basically consists of selecting a subset of relevant features (components of the vectors) that permits us to build a reliable model of the underlying process.

*Corresponding author. Email: rinaldi@dis.uniroma1.it

ISSN 1055-6788 print/ISSN 1029-4937 online

DOI: 10.1080/10556780903139388
http://www.informaworld.com

The aims of feature selection are substantially the following: improving the generalization performance of the predictive model, reducing the computational time to construct the model, and better understanding the underlying process. A complete survey on the topic can be found in [8].

In this work, we focus on feature selection problems of two-class linear models, which are common in several applications [4].

Given a finite set (the training set) of data (not necessarily linearly separable)

$$TS = \{(x^i, y^i),\ x^i \in R^n,\ y^i \in \{-1, 1\}, i = 1, \ldots, N\},$$

where the label y^i denotes the class of the vector x^i, we make predictions about points $x \in R^n$ by a linear machine defined by the decision function

$$y(x) = \text{sgn}(w^{\mathrm{T}}x + b), \tag{1}$$

where $x \in R^n$ is the input vector, $w \in R^n$ the vector of weights, $b \in R$ the threshold, $\text{sgn}: R \to \{-1, 1\}$ the *sign function* such that $\text{sgn}(t) = 1$ for $t \geq 0$ and $s(t) = -1$ for $t < 0$. We denote by $H(w, b)$ the separating hyperplane associated with the decision function (1).

We are interested in finding the relevant features of the input space R^n, namely we want to reduce the dimensionality of the data by selecting those variables that enable us to model the unknown process by a linear classifier in a subspace of R^n. Our motivation lies in the fact that, as mentioned earlier, the detection of the relevant features provides a better understanding of the underlying phenomenon, and this can be of great interest in important fields such as medicine and biology. For instance, feature selection in data concerning healthy patients and patients affected by a given pathology may help to better understand the considered pathology from a medical point of view.

In feature selection two objectives have to be considered:

(a) the goodness (to be maximized) of the data fitting of the linear classifier modelling the process;
(b) the number of input variables (to be minimized) of the classifier.

A reasonable and often adopted approach is that of formalizing the feature selection problem as an optimization problem whose objective function consists of two terms, the first related to the error on the training data (its inverse gives a measure of the goodness of the training data fitting), the second to the sparsity of the solution to be determined. The second term is also introduced to prevent *overtraining* (the machine learns too specifically the training data), a phenomenon that could lead to a linear classifier with poor generalization capabilities, namely poor capabilities to correctly classify input data never used in the training process.

The two formulations presented in [3] take the form

$$\begin{aligned} \min_{w,b,\xi} \quad & (1-\lambda) f(\xi) + \lambda g(w) \\ \text{s.t.} \quad & y^i(w^{\mathrm{T}}x^i + b) \geq 1 - \xi^i, \quad i = 1, \ldots, N, \\ & \xi^i \geq 0, \quad i = 1, \ldots, N, \end{aligned} \tag{2}$$

where $\lambda \in [0, 1)$ is a parameter, ξ^i for $i = 1, \ldots, N$ are slack variables, the first term f in the objective function is an upper bound of the average training error, and the second term g penalizes nonzero components of w. In both the formulations, the term f is the sum (possibly weighted) of the slack variables ξ^i, and hence measures the (average) violation of the constraints (thus giving an upper bound of the training error). Concerning the term g, in the first formulation it is a concave smooth approximation of the so-called zero-norm of w, denoted by $\|w\|_0$, and indicating the number of nonzero components of w. The second formulation, related to the support vector

machine (SVM) approach [12], uses as function g, three different functions: the 1-norm, the ∞-norm, and the 2-norm of w. The numerical results reported in [3] show that the two approaches lead to predictive models comparable in terms of classification accuracy. However, the concave-based approach performs better as a feature selection technique, that is, the classifiers obtained select fewer features than those determined by SVM.

In this work, we present a feature selection strategy that combines SVM and a concave optimization-based approach. Differently from [3], where the two objectives (a) and (b) are embedded in the objective function of Equation (2) and assessed by means of a parameter λ, our method operates separately on the two objectives (a) and (b). The basic ideas of the strategy are the following:

(i) to compute in a given subspace (which is the original input space at the beginning) a good linear classifier (according to the statistical learning theory [12]) by means of SVM;
(ii) to determine a sparse linear classifier using concave optimization and checking that the current solution is not 'too far' from the SVM solution in order to preserve the generalization capability of the classifier.

The method is designed in such a way as to avoid sparse solutions determined by concave minimization diverging from SVM classifiers, thus deteriorating the quality of data fitting. It involves convex quadratic and linear programming problems (which can be efficiently solved by available solvers), and this makes possible its application, as shown by the experiments, to large dimensional problems.

We tested the method on several problems with different sizes both in the number n (ranking from 34 to 7129) of features and in the number N (ranking from 40 to 13,086) of training data. The obtained results clearly show that the proposed feature selection technique is very efficient and robust.

2. Linear support vector machines

Consider the training set

$$TS = \{(x^i, y^i), x^i \in R^n, y^i \in \{-1, 1\}, i = 1, \ldots, N\}$$

and assume it is linearly separable, that is, there exists a separating hyperplane

$$H(w, b) = \{x \in R^n : w^{\mathrm{T}}x + b = 0\}$$

such that

$$\begin{aligned} w^{\mathrm{T}}x^i + b \geq 1, \quad \forall x^i : y^i = 1, \\ w^{\mathrm{T}}x^i + b \leq -1, \quad \forall x^i : y^i = -1. \end{aligned} \tag{3}$$

The *margin* $\rho(w, b)$ of a separating hyperplane $H(w, b)$ is the distance from the hyperplane to the closest training points, i.e.

$$\rho(w, b) = \min_{x^i,\, i=1,\ldots,N} \frac{|w^{\mathrm{T}}x^i + b|}{\|w\|}.$$

The SVM approach picks out, among the linear classifiers, the optimum separating hyperplane (i.e. the hyperplane having maximum margin). The basic training principle of SVM, motivated by the statistical learning theory [12], is that the expected classification error for unseen test samples is minimized, so that SVM defines a good predictive model.

The optimum hyperplane can be determined by solving the following quadratic programming:

$$\begin{aligned} &\min_{w\in R^n,\ b\in R} \ \frac{1}{2}\|w\|^2 \\ &y^i(w^{\mathrm{T}}x^i + b) \geq 1, \quad i = 1, \ldots, N. \end{aligned} \tag{4}$$

In the case that the training set is not linearly separable, the system of inequalities (3) does not admit solution. By introducing slack variables ξ^i, for $i = 1, \ldots, N$, the SVM classifier is determined by solving

$$\begin{aligned} &\min_{w\in W,\ b\in R,\ \xi\in R^N} \ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi^i, \\ &y^i(w^{\mathrm{T}}x^i + b) \geq 1 - \xi^i, \quad i = 1, \ldots, N \\ &\xi^i \geq 0, \quad i = 1, \ldots, N, \end{aligned} \tag{5}$$

where the term $\sum_{i=1}^{N}\xi^i$ is an upper bound on the training error. The regularization parameter $C > 0$ trades off margin size and training error, and is usually determined by standard cross-validation tuning procedures. More specifically, discrete values of C are defined, and for each value a k-fold cross validation on the training set is performed. The value of C which gives the lowest cross-validation error rate is selected. Recently, some algorithms have been developed to efficiently solve large-scale problems of the form (5) (see [5] and the references therein).

We observe that SVM provides a good linear classifier, but we cannot expect that the obtained separating hyperplane utilizes as few of the features as possible, since the minimization of the Euclidean norm favours solutions with many small nonzero components. We will induce sparsity by suitably exploiting the concave approach described in the next section.

3. Concave minimization for finding sparse solutions

Consider the general problem of finding a vector belonging to a polyhedron and having the minimum number of nonzero components, that is

$$\begin{aligned} &\min_{y\in R^n} \ \|y\|_0, \\ &y \in P, \end{aligned} \tag{6}$$

where $\|y\|_0$ is the zero-norm of y defined as $\|y\|_0 = \mathrm{card}\{y_j : y_j \neq 0\}$, $P \subset R^n$ is a nonempty polyhedral set. As shown in [2], this combinatorial optimization problem is NP-hard.

In order to make the problem tractable, a concave optimization-based approach has been originally proposed in [3] and successively developed in [11,14]. It is necessary to solve a separable problem of the form

$$\begin{aligned} &\min_{y,z\in R^n} \ F(z;u) = \sum_{j=1}^{n} f_j(z_j;u), \\ &y \in P, \\ &-z_j \leq y_j \leq z_j, \quad j = 1, \ldots, n, \end{aligned} \tag{7}$$

where $f_j : R \to R$ are continuously differentiable concave functions that depend on a vector $u \in R^m$ of parameters (in the proposed formulations, the number m of parameters is either one or two).

We say that an approximating problem (7) is *equivalent* to the given nonsmooth problem (6) if for suitable values of the vector u there exists a solution of Equation (7) which yields a solution of the original problem (6).

We briefly illustrate the ideas underlying the concave approach described above. We can write

$$\|y\|_0 = \sum_{j=1}^{n} s(|y_j|),$$

where $s : R \to R^+$ is the *step function* such that $s(t) = 1$ for $t > 0$ and $s(t) = 0$ for $t \leq 0$. Then, following the approach described in [3], we replace the discontinuous step function by a continuously differentiable concave function $v(t) = 1 - \mathrm{e}^{-\alpha t}$, with parameter $\alpha > 0$, thus obtaining a problem of the form

$$\begin{aligned} &\min_{y,z \in R^n} \; F(z;\alpha) = \sum_{j=1}^{n} (1 - \mathrm{e}^{-\alpha z_j}), \\ &y \in P, \\ &-z_j \leq y_j \leq z_j, \quad j = 1, \ldots, n. \end{aligned} \tag{8}$$

It has been shown in [10] that the approximating problem (8) is equivalent to the given nonsmooth problem (6).

Another concave formulation has been proposed in [14]

$$\begin{aligned} &\min_{y,z \in R^n} \; F(z;\epsilon) = \sum_{j=1}^{n} \ln(\epsilon + z_j), \\ &y \in P, \\ &-z_j \leq y_j \leq z_j, \quad j = 1, \ldots, n, \end{aligned} \tag{9}$$

with $0 < \epsilon \ll 1$. Formulation (9) is practically motivated by the fact that, due to the form of the logarithm function, it is better to increase one variable z_j while setting to zero another, rather than doing some compromise between both, and this should facilitate the computation of a sparse solution. The following two concave formulations, related to the ideas underlying Equations (8) and (9) respectively, have been proposed in [11]

$$\begin{aligned} &\min_{y \in R^n, z \in R^n} F(z;\epsilon,p) = \sum_{j=1}^{n} (z_j + \epsilon)^p, \\ &y \in P, \\ &-z_j \leq y_j \leq z_j, \quad j = 1, \ldots, n \end{aligned} \tag{10}$$

with $0 < p < 1$, and $0 < \epsilon$;

$$\begin{aligned} &\min_{y \in R^n, z \in R^n} F(z;\epsilon,p) = -\sum_{j=1}^{n} (z_j + \epsilon)^{-p}, \\ &y \in P, \\ &-z_j \leq y_j \leq z_j, \quad j = 1, \ldots, n \end{aligned} \tag{11}$$

with $1 \leq p$, and $0 < \epsilon$.

The following result (see Propositions 3–6 in [11]) shows the equivalence between the approximating smoooth problems (8)–(11) and the original nonsmooth problem (6).

PROPOSITION 3.1 *Formulations* (8)–(11) *are equivalent to Equation* (6).

The above result well motivates, from a theoretical point of view, the replacement of problem (6) with suitable smooth concave problems of the form (7).

In order to solve a concave problem of the form (7), the Frank–Wolfe algorithm [6] with unitary stepsize can be applied. The algorithm is described below, where for notational convenience we omit the dependence on the vector of parameter u of the objective function, and we set

$$\Omega = \left\{ \begin{pmatrix} y \\ z \end{pmatrix} \in R^{2n} : y \in P,\ -z_j \leq y_j \leq z_j,\ j = 1, \ldots, n \right\}. \tag{12}$$

Frank–Wolfe – Unitary Stepsize (**FW1**) Algorithm

(1) Let $(y^0, z^0)^{\mathrm{T}} \in R^{2n}$ be the starting point.
(2) For $k = 0, 1, \ldots,$
if $(y^k, z^k)^{\mathrm{T}} \notin \arg \min_{(y,z)^{\mathrm{T}} \in \Omega} \nabla F(z^k)^{\mathrm{T}} z$ then compute a vertex solution $(y^{k+1}, z^{k+1})^{\mathrm{T}}$ of

$$\min_{(y,z)^{\mathrm{T}} \in \Omega} \nabla F(z^k)^{\mathrm{T}} z \tag{13}$$

else exit.

The algorithm only involves the solution of linear programming problems. The following result, proved in [10] for a general class of concave problems, states that the algorithm generates a finite sequence and that it terminates at a stationary point, provided suitable assumptions are satisfied. These assumptions hold for problems (8)–(11).

PROPOSITION 3.2 *Let us consider problem* (7) *and assume that F is a concave, continuously differentiable function, bounded below on the feasible set that contains no straight lines going to infinity in both directions. The Frank–Wolfe algorithm with unitary stepsize converges to a vertex stationary point of problem (7) in a finite number of iterations.*

The computational experiments described in [3,11,14] show that the FW1 Algorithm, applied to concave formulations, is able to find very sparse solutions. In particular, among the theoretically equivalent concave formulations (8)–(11), slightly better results in terms of sparsity were obtained in [11] by means of formulation (11), which have been used in the computational experiments of the present work.

However, in a feature selection context where besides the objective of finding sparse solutions there is the objective of ensuring a sufficiently high classification accuracy of the separating hyperplane, the concave-based approach must be suitably adapted, and this will be shown in the next section.

4. A feature selection algorithm

In this section, we present a feature selection technique based on the combination of SVM with concave optimization. The approach exploits the good predictive capabilities of SVM classifiers and the effectiveness of concave optimization to determine sparse solutions.

The method is designed in such a way as to avoid sparse solutions determined by concave minimization diverging from SVM classifiers, thus deteriorating the quality of data fitting. More specifically, starting from the SVM solution in a given subspace, we perform a single iteration of the Frank–Wolfe algorithm on a problem whose objective function is a concave approximation of the zero-norm (that favours sparse solutions), and the constraints are the linear inequalities related to the training data correctly classified by SVM. The nonzero components of the computed separating hyperplane define a subspace of a lower dimension, and we repeat the two-phase procedure in this subspace.

In the sequel, we refer to an unspecified concave function of the form

$$F(z) = \sum_{j=1}^{n} f_j(z_j),$$

where $f_j : R \to R$ are concave smooth functions, aimed to approximate the zero-norm problem. Specific concave functions that can be practically employed are the objective functions of formulations (8)–(11).

Formally, the feature selection algorithm, based on the combination of SVM with concave programming, is reported below.

4.1 *Algorithm FS-SVMCP*

Let $W = R^n$.

(1) Solve the SVM problem

$$\begin{aligned} &\min_{w\in W, b\in R, \xi\in R^N} \frac{1}{2}\|w\|^2 + C\sum_{i\in I}\xi^i, \\ &y^i(w^{\mathrm{T}}x^i + b) \geq 1 - \xi^i, \quad i = 1, \ldots, N, \\ &\xi^i \geq 0, \quad i = 1, \ldots, N \end{aligned} \tag{14}$$

and let $(w^\star, b^\star, \xi^\star)$ be the solution obtained.

(2) Set $I = \{1, \ldots, N\}/\{i{:}(\xi^\star)^i \geq 1\}$, $z_j^\star = |w_j^\star|$ for $j = 1, \ldots, n$, and compute a vertex solution $(\hat{w}, \hat{b}, \hat{z})^{\mathrm{T}}$ of the linear programming problem

$$\begin{aligned} &\min_{w\in W, b\in R, z\in R^n} \nabla F(z^\star)^{\mathrm{T}} z, \\ &y^i(w^{\mathrm{T}}x^i + b) \geq 1, \quad i \in I, \\ &-z_j \leq w_j \leq z_j, \quad j = 1, \ldots, n. \end{aligned} \tag{15}$$

If $\|\hat{w}\|_0 < \|w^\star\|_0$ then remove features corresponding to null components of $\hat{w}$, i.e. set

$$W = \{w \in R^n : w_h = 0,\ \forall h : \hat{w}_h = 0\},$$

and go to step 1, otherwise let $H(w^\star, b^\star)$ be the separating hyperplane and exit.

At Step 1 a good classifier in a given subspace of R^n (defined by W) is determined by means of the standard SVM technique (as said in Section 2, the value of parameter C can be computed by means of a cross-validation tuning procedure). The misclassified training points x^i are those corresponding to slack variables $(\xi^*)^i$ greater than 1. The index set I, defined at Step 2, identifies the well-classified training points.

At Step 2, starting from the point $(w^\star, b^\star, z^\star)^{\mathrm{T}}$ provided by the SVM classifier, a single iteration of FW1 Algorithm is applied to the problem

$$\begin{aligned} &\min_{w\in W, b\in R, z\in R^n} F(z), \\ &y^i(w^T x^i + b) \geq 1, \quad i \in I, \\ &-z_j \leq w_j \leq z_j, \quad j = 1, \ldots, n, \end{aligned} \tag{16}$$

where, as said above, the set I identifies the inequalities corresponding to the training points well-classified by SVM classifier. The inequalities defined by the set I should induce the good behaviour of SVM classifier in terms of separation, in other words, the constraints defined by the set I impose that the hyperplane provides a decision function yielding, on the training data, the same outputs of the SVM classifier. The aim of the single-iteration of Step 2 is to determine a separating hyperplane that utilizes fewer features and is not too far from the SVM solution. The nonzero components of the separating hyperplane so determined define a subspace of lower dimension, and we repeat the two-phase procedure in this subspace. The algorithm terminates whenever no dimension reduction is obtained by the concave minimization phase.

We remark that the algorithm is very simple and that involves the use of only one parameter (the parameter C at Step 1). Furthermore, the optimization problems that must be solved at steps 1 and 2 are convex quadratic programming and linear programming problems respectively, so that they can be efficiently solved even when their dimensions are extremely large.

5. Computational experiments

In this section we describe the numerical results obtained by Algorithm FS-SVMCP on nine data sets (See *appendix* for further information) usually employed in linear classifier testing.

5.1 *Implementation details*

At Step 1, SVM problem (14) has been solved by *LIBLINEAR A Library for Large Linear Classification*, developed by the Machine Learning Group at National Taiwan University (see [5] for the details). At each iteration, the parameter C has been determined by a standard cross-validation procedure. At Step 2, we have used the concave function

$$F(z; \epsilon, p) = -\sum_{j=1}^{n} (z_j + \epsilon)^{-p} \tag{17}$$

corresponding to formulation (11) approximating the zero-norm problem, with $\epsilon = 10^{-6}$ and $p = 1$. This choice is motivated by the fact that, as previously mentioned, formulation (11) gave, in (11), results slightly better than those of the other formulations.

At each iteration, the linear programming problem (15) has been solved using *GLPK* (4.9). The experiments were carried out on Intel Pentium 4 3.2 GHz 512 MB RAM.

5.2 *Experiments and results*

For each problem, we randomly divided the available data into training set and test set, thus generating 10 instances. We grouped the nine problems into two sets:

- the first set is made of four well-known problems in Bioinformatics (breast cancer, colon cancer, leukemia, and lymphoma) having many features (of the order of thousands) and a small number of training and test data (of the order of tens);
- the second set is made of five miscellaneous problems (Ionosphere, Sonar, Musk, Sylva, and Gina) with number n of features ranking from 34 to 970, and number N of training data ranking from 180 to 13,056.

The results obtained on the two sets of test problems are shown in Tables 1 and 2 respectively, where for each problem we report

- the number Tr of training data, the number Ts of test data, the number n of features;
- the average (on the 10 instances) classification accuracy on the training set (TR%) and on the test set (TS%), and the average number of selected features $\|w^\star\|_0$ obtained by the standard linear SVM classifier (SVM);
- the average (on 10 instances) classification accuracy on the training set (TR%), on the test set (TS%), and the average number of selected features $\|w^\star\|_0$ obtained by Algorithm FS-SVMCP (FS-SVMCP).

The results of Tables 1 and 2 clearly show the effectiveness of the proposed feature selection technique. Indeed, the proposed algorithm seems to provide a good trade-off between the classification accuracy and the number of selected variables.

Note that the data sets of the first table are very different (in the structure) from those of the second table, and that the performance of the proposed algorithm on the two sets of problems are very similar. This points out a good robustness of the method (thanks to its simplicity), which is useful whenever different kind of applications must be tackled.

5.3 *Comparison with an existing feature selection method*

In order to better assess the performance of our algorithm, we have compared the obtained results with those of a well-known and deeply used feature selection method, the so-called recursive

Table 1. Results obtained by Algorithm FS-SVMCP on the first set of test problems.

				SVM			FS-SVMCP		
Data set	Tr	Ts	n	TR%	TS%	$\|w^*\|_0$	TR%	TS%	$\|w^*\|_0$
Breast cancer	40	4	7129	100	95.00	7128.8	100	90.00	10.10
Colon cancer	50	12	2000	100	88.33	2000.0	100	85.00	9.30
Leukemia	58	14	7129	100	98.57	7123.3	100	97.14	7.30
Lymphoma	85	11	4026	100	97.27	4025.0	100	96.36	10.50

Table 2. Results obtained by Algorithm FS-SVMCP on the second set of test problems.

				SVM			FS-SVMCP		
Data set	Tr	Ts	n	TR%	TS%	$\|w^*\|_0$	TR%	TS%	$\|w^*\|_0$
Ionosphere	300	51	34	100	94.51	33.0	99.83	93.92	15.90
Sonar	180	28	60	91.22	79.99	60.0	89.73	81.07	21.60
Musk	430	46	166	96.28	89.77	166.0	94.95	88.91	48.2
Sylva	13086	1308	216	98.15	97.95	181.2	98.85	98.70	17.9
Gina	3153	315	970	93.85	85.17	969.1	91.95	85.78	156.10

feature selection (RFE-SVM) method proposed in [9]. This method is based on a standard linear SVM classifier and tries to determine the best subset of r features, $r < n$ being a prefixed number of variables. The strategy is that of choosing the r features that yield the largest margin of class separation. To this aim, a sequential backward selection is performed removing some variables (or only one variable) at each iteration until r features remain. The removed features are that corresponding to the smallest values of the components of the weight vector w obtained by SVM training. In the experiments we used the Matlab implementation of RFE-SVM provided by SPIDER Toolbox [15].

For each problem, and for each of the 10 instances, we fixed r equal to the number of features selected by our algorithm. Thus, we compare the performance of the two algorithms in terms of classification accuracy on the test set. The results of the comparison are shown in Tables 3 and 4, where we report for each algorithm and for each problem the average (on the 10 instances) classification accuracy on the test set (TS%).

From the results of Tables 3 and 4 we get that our algorithm outperforms RFE-SVM. Indeed, the classification accuracy of the linear model generated by our algorithm is always better than that of the linear machine determined by RFE-SVM.

We have performed further experiments with Algorithm RFE-SVM on breast cancer and colon cancer problems, where we have detected the biggest differences, in terms of classification accuracy, between Algorithm FS-SVMCP and the standard linear SVM.

Concerning Algorithm RFE-SVM, we performed several tests, varying the number r of selected features. The obtained results are presented in Table 5 where we show the classification accuracy on the test set for different values of r. By comparing the results of Tables 1 and 5, we may observe that, even after a substantial increase in the number of selected features, RFE-SVM does not outperform our method in terms of classification accuracy. This confirms the good trade-off between the classification accuracy and the number of employed variables obtained by the algorithm proposed in this work.

Table 3. Comparison between Algorithm RFE-SVM and Algorithm FS-SVMCP on the first set of test problems.

	TS%	
Data Set	RFE-SVM	FS-SVMCP
Breast cancer	82.5	90.00
Colon cancer	81.0	85.00
Leukemia	90.0	97.14
Lymphoma	86.0	96.36

Table 4. Comparison between Algorithm RFE-SVM and Algorithm FS-SVMCP on the second set of test problems.

	TS%	
Data Set	RFE-SVM	FS-SVMCP
Ionosphere	91.76	93.92
Sonar	76.11	81.07
Musk	86.73	88.91
Sylva	98.65	98.70
Gina	83.59	85.78

Table 5. Results of Algorithm RFE-SVM on two test problems with different values of the number r of selected features.

Data set	r	TS%
Breast cancer	20	85.0
	40	87.5
	80	87.5
Colon cancer	20	81.7
	40	80.9
	80	85.9

Acknowledgements

The authors are grateful to Professor Fabio Schoen of University of Florence for his useful comments and suggestions.

References

[1] U. Alon, N. Barkai, D.A. Notterman, K. Gish, S. Ybarra, D. Mack, and A.J. Levine, *Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays*, Cell Biol. 96 (1999), pp. 6745–6750.
[2] E. Amaldi and V. Kann, *On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems*, Theore. Comput. Sci. 209 (1998), pp. 237–260.
[3] P.S. Bradley and O.L. Mangasarian, *Feature selection via concave minimization and support vector machines*, in *Machine Learning Proceedings of the Fifteenth International Conference*(*ICML '98*), J. Shavlik, ed., Morgan Kaufmann, San Francisco, California, 1998, pp. 82–90.
[4] K.-W. Chang, C.-J. Hsieh, and C.-J. Lin, *Coordinate descent method for large-scale L2-loss linear SVM*, J. Mach. Learn. Res. 9 (2008), pp. 1369–1398.
[5] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, *LIBLINEAR: A library for large linear classification*, J. Mach. Learn. Res. 9 (2008), pp. 1871–1874.
[6] M. Frank and P. Wolfe, *An algorithm for quadratic programming*, Naval Res. Logist. Quart. 3 (1956), pp. 95–110.
[7] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, and E.S. Lander, *Molecular classification of cancer: class discovery and class prediction by gene expression monitoring*, Science 531, 286(5439) 1999, p. 531.
[8] I. Guyon and A. Elisseeff, *An introduction to variable and feature selection*, J. Mach. Learn. Res. 3 (2003), pp. 1157–1182.
[9] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, *Gene selection for cancer classification using support vector machines*, Mach. Learn. 46 (2002), pp. 389–422.
[10] O.L. Mangasarian, *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, Germany, 1996, pp. 175–188.
[11] F. Rinaldi, F. Schoen, and M. Sciandrone, *Concave programming for minimizing the zero-norm over polyhedral sets*, Tech. Rep. RT 2/2008, Dipartimento Sistemi e Informatica, Università di Firenze, Comput. Optim. Appl., 2008, DOI: 10.1007/s10589-008-9202-9.
[12] V.N. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, New York, 1995.
[13] M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H. Zuzan, J.A. Olson, Jr., J.R. Marks, and J.R. Nevins, *Predicting the clinical status of human breast cancer by using gene expression profiles*, Proc. Natl Acad. Sci. 17, 98 (2001), pp. 11462–11467.
[14] J. Weston, A. Elisseef, B. Schölkopf, and M. Tipping, *Use of the zero-norm with linear models and kernel methods*, J. Mach. Learn. Res. 3 (2003), pp. 1439–1461.
[15] J. Weston, A. Elisseef, G. Baklr, and F. Sinz, The spider. Available at http://www.kyb.tuebingen.mpg.de/bs/people/spider/.

Appendix 1. The test problems

Breast cancer [13]: The duke breast-cancer data set contains 44 breast cancer tissues described by 7129 genes expression values extracted from DNA microarray data. The data are available at http://www.csie.ntu.edu.tw/ ∼ cjlin/libsvmtools/datasets/.

Colon cancer [1]: The colon cancer data set contains 22 normal and 40 colon cancer tissues described by 2000 genes expression values extracted from DNA microarray data. The data are available at http://www.csie.ntu.edu.tw/ ∼ cjlin/libsvmtools/datasets/.

Leukemia [7]: The leukemia data set contains information on gene-expression in samples from human acute myeloid (AML) and acute lymphoblastic leukemias (ALL). It contains 25 ALL examples and 47 AML examples described by 7129 genes. The data are available at http://www.csie.ntu.edu.tw/ ∼ cjlin/libsvmtools/datasets/.

Lymphoma: The gene expression of 96 samples is measured with microarray to give 4026 features; 61 of the samples are malignant and 35 are normal. The data are available at http://llmpp.nih.gov/lymphoma/.

Ionosphere: The Ionosphere data set describes a binary classification task where radar signals target two types of electrons in the ionosphere: those that show some structure (good) and those that do not (bad). The data set contains 351 samples described by 34 attributes. The data are available at http://archive.ics.uci.edu/ml/datasets/.

Sonar: The data set contains 111 sonar signals bounced off a metal cylinder and 97 bounced off a roughly cylindrical rock. Each pattern is a set of 60 numbers in the range 0.0 to 1.0. The data are available at http://archive.ics.uci.edu/ml/datasets/.

Musk: This data set contains 476 conformations of molecules. The goal is to learn to predict whether new molecules will be musks or non-musks. Each sample is described by a set of 166 features. The data are available at http://archive.ics.uci.edu/ml/datasets/.

Sylva: The task of Sylva is to classify forest cover types. This is a two-class classification problem with 216 input variables. Each pattern is composed of four records: two true records matching the target and two records picked at random. Thus half of the features are distracters. This version of the database was prepared for the WCCI 2006 challenge on performance prediction. The data are available at http://clopinet.com/isabelle/Projects/modelselect/.

Gina: The task of Gina is handwritten digit recognition. We chose the problem of separating the odd numbers from even numbers. We used 2-digit numbers. Only the unit digit is informative for that task, therefore at least half of the features are distracters. This version of the database was prepared for the WCCI 2006 challenge on performance prediction. The data are available at http://clopinet.com/isabelle/Projects/modelselect/.