Feature weighting using margin and radius based error bound optimization in SVMs

Huyen Do and Alexandros Kalousis and Melanie Hilario

University of Geneva,
Computer Science Department,
7 Route De Drize, 1227, Carouge, Switzerland
{Huyen.Do,Alexandros.Kalousis,Melanie.Hilario}@unige.ch

Abstract. The Support Vector Machine error bound is a function of the margin and radius. Standard SVM algorithms maximize the margin within a given feature space, therefore the radius is fixed and thus ignored in the optimization.
We propose an extension of the standard SVM optimization in which we also account for the radius in order to produce an even tighter error bound than what we get by controlling only for the margin.
We use a second set of parameters, $\boldsymbol{\mu}$, that control the radius introducing like that an explicit feature weighting mechanism in the SVM algorithm. We impose an l_1 constraint on $\boldsymbol{\mu}$ which results in a sparse vector, thus performing feature selection. Our original formulation is not convex, we give a convex approximation and show how to solve it. We experiment with real world datasets and report very good predictive performance compared to standard SVM.

Key words: Feature Weighting, Support Vector Machine, convex optimization

1 Introduction

Support Vector Machines have been proven one of the most successful machine learning tools over the last decade. Their wide acceptance from the machine learning community has to do by the excellent performance that they achieve over different learning problems—excellent performance that is founded on sound theoretical concepts and namely the fact that they control directly their error bound. In the classification setting they work by establishing a class separating hyperplane in some feature space, typically given by a kernel function. The theory shows that their error bound is a function of both the margin, γ, informally the distance of the nearest data points from the hyperplane, and the radius, R, of the smallest sphere enclosing the data; more precisely, the function depends on the ratio R^2/γ^2.

However, even though this double dependency is well known, all SVM algorithms optimize the error bound by focusing only on the margin and ignoring the radius. One could argue that for a given feature space the radius of the smallest sphere enclosing the data remains fixed and thus can be ignored. However it is quite easy to show that under a very simple scenario, for example by weighting or selecting features, the radius changes. The question that then arises is what is the best way to control that change, i.e. best in the sense of the error bound optimization. This question has been partially explored in the context of feature selection; for example [1] and [2] directly try to exploit this dual dependency of the error bound in order to determine which features to remove and which to retain. SVMRFE [3] also implicitly changes the radius since it removes features; however since the algorithm is based on the recursive application of a standard SVM, its cost function obviously disregards the radius.

Nevertheless it still remains a challenge to exploit all the facets of the generalization bound in the learning process. In this paper we move one step ahead in this direction and propose an SVM algorithm that optimizes the error bound by controlling both for the margin and the radius. To do so we extend the standard margin-based cost function of SVM (which controls the margin by controlling the size of the norm of the normal vector, $\mathbf{w}$, of the maximum margin hyperplane), to include also the radius. We control the latter by introducing a second vector of parameters, $\boldsymbol{\mu}$, which in fact performs feature weighting; we call our algorithm MR-SVM (Margin-Radius SVM). The proposed algorithm includes a sparsity constraint on $\boldsymbol{\mu}$ which is based on the l_1 norm; this forces many features to have a zero weight, thus performing also feature selection.

The paper is organized as follows. In section 2 we briefly review previous work on SVM that deal with both the margin and the radius, mainly in the context of feature selection. In section 3 we discuss the various error bounds that motivate the use of the radius and present the standard SVM framework. We give our main contribution in section 4 and present some experiments on several benchmark datasets in section 5. Finally, we conclude in section 6 where we also present some ideas for future work.

2 Related work

The idea of optimizing the error bound of SVM by controlling both the margin and the radius has received relatively limited attention. This despite the fact that one would expect, at least in principle, to be able to produce tighter error bounds when we control for both, thus achieving better generalization performance. Somehow naturally the only two works of which we are aware that move in that direction fall within the feature selection research domain.

Weston et al. in [1] start from exactly the same idea, i.e. that the generalization performance of SVM depends on the ratio R^2/γ^2 and not only on the margin γ and propose a feature selection algorithm which tries to optimize that ratio. To do so they introduce a binary valued vector $\boldsymbol{\sigma} \in \{0,1\}^d$, where each $\sigma_i, i := 1, \ldots, d$ corresponds to one of the dimensions of the input space. The $\boldsymbol{\sigma}$ controls which features are included or removed through a componentwise multiplication with the learning instances. They then express the ratio as a function of $\boldsymbol{\sigma}$, i.e. $f(\boldsymbol{\sigma}) = \frac{R^2}{\gamma^2}(\boldsymbol{\sigma})$. Their goal is to find that vector $\boldsymbol{\sigma}^*$

which minimizes $f(\boldsymbol{\sigma})$. However the original formulation of the problem requires searching over all possible feature subsets which is a combinatorial problem that is only tractable by greedy search methods. The authors propose an alternative approach in which they relax $\boldsymbol{\sigma}$ to be a real valued vector and formulate the problem as an approximation of integer programming which they solve by using gradient descent by computing the gradient $\partial f(\boldsymbol{\sigma})/\partial \boldsymbol{\sigma}$. There is no guarantee that the algorithm will reach a global minimum. Moreover since they view the problem as a feature selection problem they parametrize the ratio in a manner that includes or removes features on the basis of the original input space, i.e. the parametrization performed by $\boldsymbol{\sigma}$ is not done in the feature space. However the radius is computed in the feature space and obviously tighter error bounds can be achieved if the parametrization is done in the feature space, in the same way that the parametrization of the margin is done in the feature space and not in the original input space.

Rakotomamonjy in [2] examines a number of different SVM-based functions and feature ranking approaches to perform feature selection. Among the different functions he examines we find the radius-margin ratio as it was given in the previous paragraph, i.e. $f(\boldsymbol{\sigma})$. He explores two approaches to establish a ranking of the features from $f(\boldsymbol{\sigma})$, which he calls zero-order and first-order approaches. In the zero-order approach the importance of a feature is given by the value of $f(\boldsymbol{\sigma})$ when that feature has been removed. In other words the importance of feature i is given by $f(\boldsymbol{\sigma}^{(-i)})$, where $\sigma_{j\neq i}^{(-i)} = 1$ and $\sigma_i^{(-i)} = 0$. In the first-order approach the importance of a feature i is given by the value of the partial derivative $\partial f(\boldsymbol{\sigma})/\partial \sigma_i$ calculated at $\sigma_i = 1$. The two approaches are incorporated within a stepwise selection method to determine which features to retain. We note here that this work does not try to optimize $f(\boldsymbol{\sigma})$ but simply uses it as a way to determine the importance of the different features in the input space and decided which ones to retain and which to eliminate. Moreover the initial value of $f(\mathbf{1})$, where $\mathbf{1}$ is the vector of ones, may not be the optimal value and the method removes those features that have the smallest effect on the value of $f(\mathbf{1})$.

In both approaches the parametrization of the radius-margin ratio is achieved via the parametrization of the kernel function which is written in the form $K(\boldsymbol{\sigma}.\mathbf{x}_i, \boldsymbol{\sigma}.\mathbf{x}_j)$, where the operator . denotes the componentwise multiplication. [4] examined the more general problem of tuning any number of parameters of a kernel and proposed a framework that relies on the use of theoretical error bounds to perform that tuning. One of the error bounds they examined was the radius-margin ratio. Obviously the $K(\boldsymbol{\sigma}.\mathbf{x}_i, \boldsymbol{\sigma}.\mathbf{x}_j)$ form can be addressed within their framework, and in fact they did use their method and this kernel parametrization to perform feature selection and scaling. As in the two previous works, feature scaling and weighting is performed in the input space. The framework they proposed is based on a two step iterative optimization procedure. The first step solves a standard SVM problem in which the set of parameters $\boldsymbol{\sigma}$ is kept fixed. The second step uses gradient descent to update the parameters $\boldsymbol{\sigma}$ in the direction that minimizes the theoretical error bound, i.e. the radius-margin ratio. However the radius-margin ratio can have many local minima, see for example [5], and a simple gradient method may easily get trapped in one of them.

As it is apparent from the above discussion, one of the learning tasks in which SVMs have been used often is that of feature selection. Probably the most popular algorithm of this family is SVM-RFE, introduced by [3], which is a backwards feature elimination procedure that removes features based on the values of their weights as these are determined by a linear kernel SVM. In fact the work of [2] extends this work. The linear kernel has been proven one of the most popular kernels for SVM not only because of its very good predictive performance, but also because of its understandability; its weights directly reflect the importance of the corresponding features, provided that features have been normalized. These explain why it is the kernel of choice not only for the feature selection task but also more generally for the task of classification in problems with high dimensionality.

The same trend also appears in regression problems where some of the most successful algorithms for high dimensional problems are algorithms that produce linear models by imposing sparsity constraints on the vector of weights of the coefficients, e.g. Lasso, [6], LARS, [7], and more recently Adaptive Lasso, [8]. All of them use the l_1 norm to control the coefficients of the linear regression models, which is known to produce sparse weight vectors. In an interesting twist Adaptive Lasso uses two sets of weights: a first set of weights controls the importance of the different features and the second is the set of lasso coefficients learned on the weighted feature set. The author proposes to derive the first set of weights from the solution of the Ordinary Least Squares regression [9]. Note here that unlike the different regression algorithms mentioned the standard SVM does not impose strong sparsity constraints such as the one imposed by the l_1 norm.

3 SVM, margin and radius based error bounds

There are a number of theorems in statistical learning that bound the expected classification error of the thresholded linear classifier given by the maximum margin hyperplane by quantities that are related to its margin and the radius of the smallest sphere that encloses the data. Below we give two of them that correspond to the cases of linearly separable and non-separable training sets.

Consider a mapping $\mathbf{x} \mapsto \mathbf{\Phi}(\mathbf{x})$ in which a training instance $\mathbf{x} \in \mathcal{X}$ is mapped to an inner product feature space $\mathcal{H}$. Moreover the inner product of $\mathcal{H}$ is given by $K(.,.)$, a kernel function defined in the original input space $\mathcal{X}$, i.e. $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{\Phi}(\mathbf{x}_i), \mathbf{\Phi}(\mathbf{x}_j) \rangle$.

Theorem 1 *[4], Given a training set* $S = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_l, y_l)\}$ *of size* l*, a feature space* $\mathcal{H}$ *and a hyperplane* $(\mathbf{w}, b)$*, the margin* $\gamma(\mathbf{w}, b, S)$ *and the radius* $R(S)$

are defined by

$$\gamma(\mathbf{w}, b, S) = \min_{(\mathbf{x}_i, y_i) \in S} \frac{y_i(\langle \mathbf{w}, \mathbf{\Phi}(\mathbf{x}_i) \rangle + b)}{\|\mathbf{w}\|}$$
$$R(S) = \min_{\mathbf{a}} \max_{i} \|\mathbf{\Phi}(\mathbf{x}_i) - \mathbf{a}\|$$

The maximum margin algorithm $L_l : (\mathcal{X} \times \mathcal{Y})^l \to \mathcal{H} \times \mathbb{R}$ *takes as input a training set of size* l *and returns a hyperplane in feature space such that the margin* $\gamma(\mathbf{w}, b, S)$ *is maximized. Note that assuming the training set is separable means that* $\gamma > 0$. *Under this assumption, for all probability measures* P *underlying the data* S, *the expectation of the misclassification probability*

$$p_{err}(\mathbf{w}, b) = P(sign(\langle \mathbf{w}, \mathbf{\Phi}(\mathbf{X}) \rangle + b) \neq Y)$$

has the bound

$$E\{p_{err}(L_{l-1}(Z))\} \leq \frac{1}{l} E \left\{ \frac{R^2(Z)}{\gamma^2(L_l(Z), Z)} \right\}$$

The expectation is taken over the random draw of a training set Z *of size* $l-1$ *for the left hand side and* l *for the right hand side.*

The following theorem gives a similar result for the error bound of the linearly non-separable case.

Theorem 2 *[10], Consider thresholding real-valued linear functions* $\mathcal{L}$ *with unit weight vectors on an inner product space* $\mathcal{H}$ *and fix* $\gamma \in \mathcal{R}^+$. *There is a constant* c, *such that for any probability distribution* $\mathcal{D}$ *on* $\mathcal{H} \times \{-\infty, \infty\}$ *with support in a ball of radius* R *around the origin, with probability* $1 - \delta$ *over* l *random examples* S, *any hypothesis* $f \in \mathcal{L}$ *has error no more than:*

$$err(f)_{\mathcal{D}} \leq \frac{c}{l} \left(\frac{R^2 + \|\boldsymbol{\xi}\|_2^2}{\gamma^2} log^2 l + log \frac{1}{\delta} \right), \tag{1}$$

where $\boldsymbol{\xi} = \boldsymbol{\xi}(f, S, \gamma)$ *is the margin slack vector with respect to* f *and* γ.

It is clear from both theorems that the bound on the expected error depends not only on the margin but also on the radius of the data. The expected error is bounded in the linearly separable and non-separable cases by functions of the ratios R^2/γ^2 and $(R^2 + \|\boldsymbol{\xi}\|_2^2)/\gamma^2$ respectively.

The standard soft margin SVM builds exactly on these results, namely theorem 2, by learning maximal margin hyperplanes while controlling for the l_2 norm of the slack vector in an effort to optimize the error bound given in equation 1. Maximizing for the margin is equivalent to minimizing the norm of the normal vector, $\mathbf{w}$, of the separating hyperplane, thus the optimization problem that is solved by the standard soft margin SVM is given by:

$$\min_{\boldsymbol{\xi}, \mathbf{w}, b} \ \frac{1}{2} \langle \mathbf{w}, \mathbf{w} \rangle + \frac{C}{2} \sum_{i=1}^{l} \xi_i^2 \tag{2}$$
$$s.t \ \ y_i(\langle \mathbf{w}, \mathbf{\Phi}(\mathbf{x}_i) \rangle + b) \geq 1 - \xi_i, i = 1, \ldots, l$$

C is a regularization parameter that controls the trade off between the size of the margin and the norm of the slack variables vector; the latter is directly related to the total number of training set missclassifications. Obviously this approach to error bound optimization focuses exclusively on the margin and ignores the radius; under this problem formulation the latter is fixed and optimizing the cost function of equation 2 is equivalent to optimizing the error bound given in equation 1. Usually we solve the dual optimization problem of equation 2; this is given by:

$$\max_{\boldsymbol{\alpha}} W(\boldsymbol{\alpha}) = \sum_{i}^{l} \alpha_i - \frac{1}{2} \sum_{ij}^{l} \alpha_i \alpha_j y_i y_j (K(\mathbf{x}_i, \mathbf{x}_j) + \frac{1}{C} \delta_{ij})$$
$$s.t \quad \sum_{i=1}^{l} y_i \alpha_i = 0, \alpha_i \geq 0, i = 1, ..., l.$$

where $\boldsymbol{a}$ is the vector of Langrange multipliers, and δ_{ij} is the Kronecker δ, defined to be 1 if $i = j$ and 0 otherwise. The decision function of SVM is $f(\mathbf{x}) = sign(\langle \mathbf{w}^*, \boldsymbol{\Phi}(\mathbf{x}) \rangle + b^*) = sign(\sum_i^l y_i \alpha_i^* K(\mathbf{x_i}, \mathbf{x}) + b^*)$ where $\mathbf{w}^*$, b^* and α_i^* are the solutions of 2 and its dual form.

The radius of the smallest sphere that contains all instances $\mathbf{x}_i$ in the $\mathcal{H}$ feature space defined by the $\mathbf{\Phi}(\mathbf{x})$ mapping is computed by the following formula [11]:

$$\min_{R, \mathbf{\Phi}(\mathbf{x}_0)} \quad R^2 \tag{3}$$
$$s.t. \quad \|\mathbf{\Phi}(\mathbf{x}_i) - \mathbf{\Phi}(\mathbf{x}_0)\|^2 \leq R^2, \forall i$$

where $\mathbf{\Phi}(\mathbf{x}_0)$ is the center of the sphere.

4 Margin and Radius based SVM

In this section we will show how it is possible to control not only for the margin but also for the radius in an effort to achieve better error bounds. Consider the feature space $\mathcal{H}$ given by the mapping function $\boldsymbol{\Phi}(\mathbf{x}) = (\Phi_1(\mathbf{x}), \Phi_2(\mathbf{x}), ..., \Phi_d(\mathbf{x})) \in \mathcal{R}^d$, where $\mathbf{x} \in \mathcal{X}$; let $\boldsymbol{\mu} \in R^d$ be a vector of parameters whose role will be to perform feature weighting in the feature space. Then the feature space $\mathcal{H}_{\boldsymbol{\mu}}$ given by the mapping $\boldsymbol{\Phi}_{\boldsymbol{\mu}}(\mathbf{x}) = \boldsymbol{\Phi}(\mathbf{x}).\sqrt{\boldsymbol{\mu}}$ is a feature space the radius of which is no longer constant but can be controlled by the weighting vector $\boldsymbol{\mu}$ ($\sqrt{\boldsymbol{\mu}}$ denotes the componentwise square root of $\boldsymbol{\mu}$). Therefore, if we use the standard SVM, which maximizes only the margin, we will not be fully optimizing the generalization bound given in equation 1. To exploit the full potential of the radius-margin error bound we should also learn the vector $\boldsymbol{\mu}$. Note that in the standard SVM formulation $\boldsymbol{\mu} = \mathbf{1}$, i.e. all features of the feature space have equal importance or weight. For reasons that will become apparent below, and without loss of generality, we will work in the normalized feature space $\mathcal{H}'$, i.e. $\boldsymbol{\Phi}'(\mathbf{x}) = \frac{\boldsymbol{\Phi}(\mathbf{x})}{\|\boldsymbol{\Phi}(\mathbf{x})\|}$; this normalization is also achieved by the kernel $K'(\mathbf{x},\mathbf{y}) = \frac{K(\mathbf{x},\mathbf{y})}{\sqrt{K(\mathbf{x},\mathbf{x})K(\mathbf{y},\mathbf{y})}}$, where $K(\mathbf{x},\mathbf{y})$ is the kernel associated with the $\boldsymbol{\Phi}(\mathbf{x})$ mapping. To simplify notation in the following we will be using $\mathcal{H}$, $\boldsymbol{\Phi}(\mathbf{x})$, and $K(\mathbf{x},\mathbf{y})$, instead of $\mathcal{H}'$, $\boldsymbol{\Phi}'(\mathbf{x})$, and $K'(\mathbf{x},\mathbf{y})$.

We will now discuss a number of inequalities that relate the radius of the $\mathcal{H}_{\boldsymbol{\mu}}$ space to the radii of the spaces defined by each one of its features. Consider the one-dimensional space given by $\mathcal{H}_k$, $\Phi_k(\mathbf{x})$, which we get by projecting $\mathcal{H}$ on its k^{th} dimension. Its radius is given by $R_k = (\max_j(\Phi_k(\mathbf{x}_j)) - \min_j \Phi_k(\mathbf{x}_j))/2$. Let $R^2_{\boldsymbol{\mu}}$ be the radius of the feature space $\mathcal{H}_{\boldsymbol{\mu}}$. The following inequalities hold:

$$\max_k(\mu_k R_k^2) \leq R^2_{\boldsymbol{\mu}} \leq \sum_{k=1}^d \mu_k R_k^2 \leq \max_k(R_k^2) \leq 1 \qquad (4)$$
$$s.t.\ \sum_k^d \mu_k = 1, \mu_k \geq 0,\ \forall k$$

The inequality, $\max_k(R_k^2) \leq 1$, holds because the feature space $\mathcal{H}$ is normalized. The rest of the proof is given in the appendix.

We will now describe how we can optimize the radius-margin error bound given in equation 1 by learning also the $\boldsymbol{\mu}$ vector. A straightforward way to do so is to modify the cost function of the standard SVM so that it also includes the radius. We define the the following optimization problem in the $\mathcal{H}_{\boldsymbol{\mu}}$ feature space:

$$\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\mu}} \ \frac{1}{2}\langle \mathbf{w},\mathbf{w}\rangle R^2_{\boldsymbol{\mu}} + \frac{C}{2}\sum_{i=1}^{l} \xi_i^2 \qquad (5)$$
$$s.t.\ \ y_i(\langle \mathbf{w}, \sqrt{\boldsymbol{\mu}}.\boldsymbol{\Phi}(\mathbf{x}_i)\rangle + b) \geq 1 - \xi_i, \forall i$$

By rewriting $\mathbf{w} := \sqrt{\boldsymbol{\mu}}.\mathbf{w}$ we also have $w_k := \sqrt{\mu_k} w_k$ and then we can rewrite equation 5 as:

$$\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\mu}} \ \frac{1}{2}\sum_k^d \frac{\langle w_k, w_k\rangle}{\mu_k} R^2_{\boldsymbol{\mu}} + \frac{C}{2}\sum_i^l \xi_i^2 \qquad (6)$$
$$s.t.\ \ y_i(\sum_k^d \langle w_k, \Phi_k(\mathbf{x}_i)\rangle + b) \geq 1 - \xi_i,$$
$$\sum_{k=1}^d \mu_k = 1, \mu_k \geq 0, \forall k$$

The constraint $\sum_{k=1}^d \mu_k = 1$ is added so that we get a unique solution. If $\mu_k = 0$ we will see that from the dual form given later we have $w_k = 0$; in this case, we use the convention that $\frac{0}{0} = 0$. Note the similarity of the two sets of parameters, $\mathbf{w}$ and $\boldsymbol{\mu}$, to the two sets of weights used by the Adaptive Lasso algorithm of [8], however here we simultaneously optimize over both sets of parameters within the same optimization problem and not independently as it is done in [8].

We will denote the cost function of equation 6 by $F(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\mu})$. This optimization problem is not a convex optimization problem because the cost function F is not convex. From the set of inequalities given in equation 4 we have

$R^2_{\boldsymbol{\mu}} \leq \sum_k^d \mu_k R_k^2 \leq 1$ and from this we can get:

$$\begin{aligned} F(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\mu}) = \frac{1}{2}\sum_k^d \frac{\langle w_k, w_k\rangle}{\mu_k} R^2_{\boldsymbol{\mu}} + \frac{C}{2}\sum_i^l \xi_i^2 \leq \\ \frac{1}{2}\sum_k^d \frac{\langle w_k, w_k\rangle}{\mu_k} \sum_k^d \mu_k R_k^2 + \frac{C}{2}\sum_i^l \xi_i^2 = \\ (\frac{1}{2}\sum_k^d \frac{\langle w_k, w_k\rangle}{\mu_k} + \frac{C}{2\sum_k^d \mu_k R_k^2}\sum_i^l \xi_i^2)\sum_k^d \mu_k R_k^2 \leq \\ (\frac{1}{2}\sum_k^d \frac{\langle w_k, w_k\rangle}{\mu_k} + \frac{C}{2\sum_k^d \mu_k R_k^2}\sum_i^l \xi_i^2) = \widetilde{F}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\mu}) \end{aligned} \tag{7}$$

The tighter the bound $R^2_{\boldsymbol{\mu}} \leq \sum_k^d \mu_k R_k^2$ is, the closer $\widetilde{F}$ will be to the original error bound given by F. $\widetilde{F}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\mu})$ is a convex function so instead of the original soft margin optimization problem given in formula 6 we propose to solve the following upper bounding convex optimization problem:

$$\begin{aligned} \min_{\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\mu}} \quad & \frac{1}{2}\sum_k^d \frac{\langle w_k, w_k\rangle}{\mu_k} + \frac{C}{2\sum_k^d \mu_k R_k^2}\sum_i^l \xi_i^2 \\ s.t. \quad & y_i(\sum_k^d \langle w_k, \Phi_k(x_i)\rangle + b) \geq 1 - \xi_i, \\ & \sum_{k=1}^d \mu_k = 1, \mu_k \geq 0, \forall k \end{aligned} \tag{8}$$

The l_1 norm constraint on $\boldsymbol{\mu}$ has the potential to result in a sparser solution, i.e. many of the μ_i could be zero, which can not be obtained using standard SVM. The dual function of the optimization problem of equation 8 is:

$$\begin{aligned} W_s(\boldsymbol{\alpha}, \boldsymbol{\mu}) &= -\frac{1}{2}\sum_{ij}^l \alpha_i\alpha_j y_i y_j \sum_k^d \mu_k \langle \Phi_k(\mathbf{x}_i), \Phi_k(\mathbf{x}_j)\rangle \\ &\quad + \sum_i^l \alpha_i - \frac{\sum_k^d \mu_k R_k^2}{2C}\langle \boldsymbol{\alpha}, \boldsymbol{\alpha}\rangle \\ &= -\frac{1}{2}\sum_{ij}^l \alpha_i\alpha_j y_i y_j (\sum_k^d \mu_k \langle \Phi_k(\mathbf{x}_i), \Phi_k(\mathbf{x}_j)\rangle \\ &\quad + \frac{\sum_k^d \mu_k R_k^2}{C}\delta_{ij}) + \sum_i^l \alpha_i \end{aligned} \tag{9}$$

The dual optimization problem is:

$$\begin{aligned}
&\max_{\boldsymbol{\alpha},\boldsymbol{\mu}} \; W_s(\boldsymbol{\alpha},\boldsymbol{\mu}) \\
&s.t. \; \sum_i^l \alpha_i y_i = 0, \\
&\quad \alpha_i \geq 0, \forall i \\
&\quad \sum_{ij}^l \alpha_i \alpha_j y_i y_j \langle \Phi_k(\mathbf{x}_i), \Phi_k(\mathbf{x}_j) \rangle = \frac{R_k^2 C \sum_i^l \xi_i^2}{(\sum_k \mu_k R_k^2)^2}, \forall k,
\end{aligned} \tag{10}$$

As it is obvious the cost function and the constraints are not expressed in the form of a kernel function on the feature space $\mathcal{H}$ but instead require access to its explicit representation. This limits for the moment the application of the method that we propose here only to features spaces for which we have access to their explicit form, e.g. linear or polynomial feature spaces. In the next section we will show how we can solve this optimization problem.

4.1 Algorithm

The dual function 9 is quadratic with respect to $\boldsymbol{\alpha}$ and linear with respect to $\boldsymbol{\mu}$. One way to solve the optimization problem 8 is by using a two step iterative algorithm such as the ones described in [4, 12]. Following such a two step approach, in the first step we will solve a quadratic problem that optimizes over $(\mathbf{w}, b)$, while keeping $\boldsymbol{\mu}$ fixed; as a consequence the resulting dual function is a simple quadratic function of $\boldsymbol{\alpha}$ which can be optimized easily. In the second step we will solve a linear problem that optimizes over $\boldsymbol{\mu}$. More precisely the formulation of the optimization problem with the two-step approach takes the following form:

$$\begin{aligned}
&\min_{\boldsymbol{\mu}} \; J(\boldsymbol{\mu}) \\
&s.t. \; \sum_{k=1}^d \mu_k = 1, \mu_k \geq 0, \forall k
\end{aligned} \tag{11}$$

where

$$J(\boldsymbol{\mu}) = \begin{cases} \min_{\mathbf{w},b} \; \frac{1}{2} \sum_k^d \frac{\langle w_k, w_k \rangle}{\mu_k} + \frac{C}{\sum_k^d \mu_k R_k^2} \sum_i^l \xi_i^2 \\ s.t. \quad y_i(\sum_k^d \langle w_k, \Phi_k(\mathbf{x}_i) \rangle + b) \geq 1 - \xi_i \end{cases} \tag{12}$$

To solve the outer optimization problem, i.e. $\min_{\boldsymbol{\mu}} J(\boldsymbol{\mu})$, we use gradient descent.

At each iteration, we fix $\boldsymbol{\mu}$, compute the value of $J(\boldsymbol{\mu})$ and then compute the gradient of $J(\boldsymbol{\mu})$ with respect to $\boldsymbol{\mu}$. The dual function of equation 12 is the $W_s(\boldsymbol{\alpha}, \boldsymbol{\mu})$ function already given in equation 9. Since $\boldsymbol{\mu}$ is fixed we now optimize only over $\boldsymbol{\alpha}$ (the resulting dual optimization problem is much simpler compared to the original soft margin dual optimization problem given in formula 10):

$$\max_{\boldsymbol{\alpha}} \; W_s(\boldsymbol{\alpha}, \boldsymbol{\mu})$$
$$s.t. \; \sum_i^l \alpha_i y_i = 0, \alpha_i \geq 0, \forall i$$

which has the same form as the SVM quadratic optimization problem, the only difference is that the C parameter here is equal to $\frac{C}{\sum_k^d \mu_k R_k^2}$.

For the strong duality, at the optimal solution $\boldsymbol{\alpha}^*$, the value of $W_s(\boldsymbol{\alpha}, \boldsymbol{\mu})$, and thus the $J(\boldsymbol{\mu})$ value, is given by:

$$W_s(\boldsymbol{\alpha}^*, \boldsymbol{\mu}) = -\frac{1}{2} \sum_{ij}^l \alpha_i^* \alpha_j^* y_i y_j (\sum_k^d \mu_k \langle \varPhi_k(\mathbf{x}_i), \varPhi_k(\mathbf{x}_j) \rangle + \frac{\sum_k^d \mu_k R_k^2}{C} \delta_{ij}) + \sum_i^l \alpha_i^*$$

The last step of the algorithm is to compute the gradient of the $J(\boldsymbol{\mu})$ function, formula 12, with respect to $\boldsymbol{\mu}$. As [4] and [12] have pointed out, we can use the theorem of Bonnans and Shapiro, [13], to compute gradients of such functions. Hence:

$$\frac{\partial J(\boldsymbol{\mu})}{\partial \mu_k} = -\frac{1}{2} \sum_{ij}^l \alpha_i^* \alpha_j^* y_i y_j (\langle \varPhi_k(\mathbf{x}_i), \varPhi_k(\mathbf{x}_j) \rangle + \frac{R_k^2}{C} \delta_{ij})$$

To compute the optimal step in the gradient descent we used line search. The complete two-step procedure is given in algorithm 1.

Algorithm 1 MR-SVM

```
Initialize $\mu_k^1 = \frac{1}{d}$ for $k = 1, ..., d$
repeat
    Set $R_{\boldsymbol{\mu}}^2 = \sum_k^d \mu_k^t R_k^2$
    compute $J(\boldsymbol{\mu}^t)$ as the solution of a quadratic optimization problem with given $\mu_k^t$
    compute $\frac{\partial J}{\partial \mu_k}$ for $k = 1, ..., d$
    compute optimal step $\gamma_t$
    $\boldsymbol{\mu}^{t+1} \leftarrow \boldsymbol{\mu}^t + \gamma_t \frac{\partial J(\boldsymbol{\mu})}{\partial \boldsymbol{\mu}}$
until stopCriterion is true
```

4.2 Computational complexity

At each step of the iteration we have to compute the solution of a standard SVM, with a fixed $\boldsymbol{\mu}$ and C equal to $\frac{C}{\sum_k^d \mu_k R_k^2}$, which has a complexity $O(n^3)$ where n

is number of instances. Moreover when $\boldsymbol{\mu}$ is updated we have to recompute the approximation of $R^2_{\boldsymbol{\mu}}$, a computation that is linear in the number of features, $O(d)$, where d is number of features.

5 Experiments

We experimented with 12 different datasets. Six of them, *Ionosphere, Liver, Sonar, Wdbc, Wpbc, Musk1*, were taken from the UCI repository, and six, *Colon-Cancer, CentralNervousSystem, FemaleVsMale, Leukemia, stroke, ovarian*, [14], from the domain of genomics and proteomics. A short description of the datasets is given in Table 1. In the experiments we limit ourselves to the linear feature space, although as we mentioned previously any feature space for which we have access to its explicit form can be used. We compare the performance of the standard SVM with the linear kernel, *STD*-SVM, to that of *MR*-SVM. We tuned

Table 1. Datasets Description

Dataset	#Inst.	#Features	#Class1	#Class2
Ionosphere	351	34	126	225
Liver	345	6	145	200
Sonar	208	60	97	111
Wdbc	569	32	357	212
Wpbc	198	34	151	47
Musk1	476	166	269	207
ColonCancer	62	2000	40	22
CentralNervous	60	7129	21	39
FemaleVSMale	134	1524	67	67
Leukemia	72	7128	25	47
Stroke	208	171	101	107
Ovarian	253	385	62	91
Prostate	322	390	253	69

the hyperparameter C by inner cross-validation choosing from the set of values $\{0.1, 1, 10, 100, 500, 1000\}$. We terminate *MR*-SVM when the duality gap is smaller than 0.01. We estimated the classification error using 10-fold cross validation. In table 2 we give the results including: the classification errors of both algorithms, the number of non-zero weight features selected by *MR*-SVM, the percentage that these non-zero weight features represent with respect to the total number of features, and the result of McNemar's test of significance with a significance level of 0.05 (if *MR*-SVM is significantly better than standard SVM we denote that by +, if it is significantly worse by -, and if they are equivalent by =).

As it is obvious from the results the performance of *MR*-SVM is much better than that of standard SVM. Its classification performance is significantly better

Table 2. Results of the experiments. The average errors of standard SVM and MR-SVM are reported, together with the average number of non-zero weight features established by MR-SVM and the respective number of total features. The last column gives the results of the McNemar's significance test, + means that MR-SVM is significantly better than the standard SVM.

Dataset	STD-SVM	MR-SVM	$\#\mu_i \neq 0$	$\frac{\#\mu_i \neq 0}{\#Features}$	Sig.
Ionosphere	11.71	11.14	28.3	83.23	+
Liver	32.35	32.35	5.9	98.33	=
Sonar	24.5	23.00	44.3	73.88	+
Wdbc	1.96	2.50	16.7	52.18	-
Wpbc	18.95	17.37	24.8	72.94	+
Musk1	13.62	13.83	68.1	41.02	=
ColonCancer	15.00	16.67	2000	100	=
CentralNervous	38.33	31.67	6442	90.36	+
FemaleVSMale	13.08	11.54	1524	100	+
Leukemia	1.43	1.43	6922	97.10	=
Stroke	28.00	26.00	76.8	44.91	+
Ovarian	04.80	3.60	53.2	13.81	+
Prostate	18.44	20.00	93.07	23.86	-

in seven out of the 12 datasets and significantly worse only in two of them. Remember here that the two algorithms operate in exactly the same space, i.e. the one that corresponds to the linear kernel. Their only difference is that MR-SVM directly optimizes for both the radius and the margin and not just the margin as the standard SVM does. This has the potential, at least in theory, to produce lower error bounds than those we would get by optimizing only for the margin—a fact that seems to be confirmed by the performance results we get.

Apart from the very good classification performance that MR-SVM achieves we also get for many of the datasets, though not for all of them, a significant reduction of the number of features actually used in the learned model, many of them are assigned a μ_i that has a value of zero due to the use of the l_1 constraint. The mean value of features retained is around 68.6% of the total number of features over the different datasets.

6 Discussion and future work

In this paper we present an extension of the standard SVM that incorporates in its cost function not only the margin but also the radius of the smallest sphere that encloses the data, thus implementing directly well known error bounds from statistical learning theory. Our experimental results show that indeed optimizing the error bound accounting for both the radius and the margin leads to much better classification performance than when we optimize only for the margin as it is typically done in the standard SVM algorithms. We want to further verify the preliminary results reported here by experimenting with more datasets; if these are verified, and provided that we are able to provide a kernelized version, the proposed algorithm has a potential of very wide application.

A possible way of producing a kernelized version of our algorithm is by using kernel PCA [15] and expressing the original feature space by projecting it on its PCA components. In this case we do not need the explicit representation of the feature space, and we will work on the representation given by the projections on the PCA dimensions of the feature space. Since these can be computed explicitly we will be able to directly apply our algorithm on the transformed space.

Apart from the remarkable predictive performance, a further advantage of the algorithm comes as a result of the way we control the radius through the introduction of the $\boldsymbol{\mu}$ vector of parameters. This vector weights the different features in the feature space; moreover due to the incorporated sparsity constraint on $\boldsymbol{\mu}$ the algorithm has the potential to produce sparser linear models than those of the standard SVM. The result is that we do not only get a feature weighting mechanism but we also perform direct feature selection. However we would like to explore further the l_1 constraint defined on the $\boldsymbol{\mu}$ vector in order to see whether it is possible to control the desired level of sparsity in the final models. Remember that we set that constraint to one so that we get a unique solution. For some of the datasets this resulted in quite sparse solutions retaining as few as 14% or 24% of the original features, yet in others it retained all of them. We want to understand better the conditions under which this constraint becomes active and removes a significant number of features. Moreover we would like to explore the option of regularizing this norm as it is done in other methods, such as Lasso, in order to get even sparser solutions.

Finally we should mention that it is straightforward to use our algorithm, in its present form, within the SVM-RFE algorithm in order to replace the standard linear kernel SVM used there. Its advantage over the standard SVM will be the fact that at each iteration we can potentially remove a larger number of features the weight of which will be zero due the sparsity constraint and with a better predictive performance as our results indicate.

Acknowledgments The work reported in this paper was partially funded by the European Commission through EU projects DropTop (FP6-037739), DebugIT (FP7-217139) and e-LICO (FP7-231519).

References

1. Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, J., Vapnik, V.: Feature selection for svms. Advances in Neural Information Processing Systems **13** (2000) 668–674
2. Rakotomamonjy, A.: Variable selection using svm-based criteria. Journal of Machine Learning Research **3** (2003) 1357–1370
3. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using suppor vector machine. Machine Learning **46** (2002) 389–422
4. Chapelle, O., Vapnik, V., Bousquet, O., Mukherjee, S.: Choosing multiple parameters for support vector machines. Machine Learning **46**(1-3) (2002) 131–159

5. Duan, K., Keerthi, S.S., Poo, A.N.: Evaluation of simple performance measures for tuning svm hyperparameters. Neurocomputing **51** (2002) 41–59
6. Tibshirani, R.: Regression shrinkage and selection via the lasso. Roal statistics **58** (1996) 276–288
7. Efron, B., Hastie, T., Tibshirani, R.: Least angle regression. Annals of statistics (2003)
8. Zou, H.: The adaptive lasso and its oracle properties. Journal of the American statistical association **101** (2006) 1418–1429
9. Hastie, T., Tibshirani, R., Friedman, J.: The elements of statistical learning theory. Springer (2001)
10. Cristianini, N., Shawe-Taylor, J.: An introduction to Support Vector Machines. Cambridge University Press (2000)
11. Vapnik, V.: Statistical learning theory. Wiley Interscience (1998)
12. Bach, F., Rakotomamonjy, A., Canu, S., Grandvalet, Y.: SimpleMKL. Journal of Machine Learning Research (2008)
13. Bonnans, J., Shapiro, A.: Optimization problems with perturbation: A guided tour. SIAM Review **40**(2) (1998) 202–227
14. Kalousis, A., Prados, J., Hilario, M.: Stability of feature selection algorithms: a study on high dimensional spaces. Knowledge and Information Systems **12**(1) (2007) 95–116
15. Shawe-Taylor, J., Cristianini, N.: Kernel methods for pattern analysis. Cambridge University Press (2004)
16. Leo Liberti, N.M.: Global OPtimization - From Theory to Implementation. Springer (2006)
17. R. Collobert, J.W., Bottou., L.: Trading convexity for scalability. Proceedings of the 23th Conference on Machine Learning (2006.)
18. Stephen Boyd, L.V., ed.: Convex optimization. Cambrige University Press (2004)

Appendix

Proof of inequality (4) If $K_{\boldsymbol{\mu}}(\mathbf{x}, \mathbf{x}')$ is the kernel function associated with the $\mathbf{\Phi}_{\boldsymbol{\mu}}(\mathbf{x})$ mapping then the computation of the radius in the dual form is given by [15]:

$$\max_{\beta_i \beta_j} \; R^2 = \sum_{i}^{l} \beta_i K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{ij}^{l} \beta_i \beta_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{13}$$

$$s.t. \; \sum_{i}^{l} \beta_i = 1, \beta_i \geq 0$$

We also introduce the kernels $K_{\boldsymbol{\mu}}$ and K_k as follows: $K_{\boldsymbol{\mu}}(\mathbf{x}_i, \mathbf{x}_j) = \langle \boldsymbol{\Phi}_{\boldsymbol{\mu}}(\mathbf{x}_i), \boldsymbol{\Phi}_{\boldsymbol{\mu}}(\mathbf{x}_j) \rangle = \langle \sqrt{\boldsymbol{\mu}}.\boldsymbol{\Phi}(\mathbf{x}_i), \sqrt{\boldsymbol{\mu}}\boldsymbol{\Phi}(\mathbf{x}_j) \rangle = \sum_{k=1}^{d} \mu_k \langle \Phi_k(\mathbf{x}_i), \Phi_k(\mathbf{x}_j) \rangle = \sum_{k=1}^{d} \mu_k K_k(\mathbf{x}_i, \mathbf{x}_j)$, i.e., $K_{\boldsymbol{\mu}}$ is the kernel, inner product, in the weighted feature space, $\mathcal{H}_{\boldsymbol{\mu}}$, and $K_k(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi_k(\mathbf{x}_i), \Phi_k(\mathbf{x}_j) \rangle$, is the trivial kernel, inner product, on the k^{th} dimension, $\mathcal{H}_k$, of the non-weighted feature space $\mathcal{H}$. Remember that the solution $\boldsymbol{\mu}$ satisfies: $\sum_{k=1}^{d} \mu_k = 1$.

If β^* is the optimal solution of (13) when $K = K_{\boldsymbol{\mu}}$, and $\hat{\beta}^k$ is the optimal solution of (13) when $K = K_k$, i.e. :

$$R^2_{\boldsymbol{\mu}} = \sum_{k=1}^{d} \mu_k (\sum_{i=1}^{l} \beta^*_i K_k(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j=1}^{l} \beta^*_i \beta^*_j K_k(\mathbf{x}_i, \mathbf{x}_j))$$
$$R^2_k = \sum_{i=1}^{l} \hat{\beta}^k{}_i K_k(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j=1}^{l} \hat{\beta}^k{}_i \hat{\beta}^k{}_j K_k(\mathbf{x}_i, \mathbf{x}_j)$$

then

$$\sum_{i=1}^{l} \beta^*_i K_k(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j=1}^{l} \beta^*_i \beta^*_j K_k(\mathbf{x}_i, \mathbf{x}_j) \leq$$
$$\sum_{i=1}^{l} \hat{\beta}^k{}_i K_k(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j=1}^{l} \hat{\beta}^k{}_i \hat{\beta}^k{}_j K_k(\mathbf{x}_i, \mathbf{x}_j)$$

Therefore: $R^2_{\boldsymbol{\mu}} \leq \sum_{k=1}^{d} \mu_k R^2_k$. This still obviously holds true when K is linear kernel.

Proof of convexity of R-MKL (Eq.8) To prove that 8 is convex, it is enough to show that functions $\frac{x^2}{\mu}$, where $x \in \mathbb{R}$, $\mu \in \mathbb{R}^+$; and $\frac{\xi^2}{\sum_k^M \alpha_k \mu_k}$, where $\xi \in \mathbb{R}, \mu_k, \alpha_k \in \mathbb{R}^+$, are convex. The former is quadratic-over-linear function which is convex. The later is convex because its epigraph is a convex set [18].