# Efficient Tuning of SVM Hyperparameters Using Radius/Margin Bound and Iterative Algorithms

*S.S. Keerthi*
Dept. of Mechanical Engineering
National University of Singapore
Singapore 119260

Feb 3, 2001

**Control Division Technical Report CD-01-02**

**Abstract**

*In this short report I discuss implementation issues related to the tuning of the hyperparameters of a Support Vector Machine (SVM) with $L_2$ soft margin, for which the radius/margin bound is taken as the index to be minimized and iterative techniques are employed for computing radius and margin. The implementation is shown to be feasible and efficient even for large problems having more than 10,000 support vectors.*

## 1 Introduction

The basic problem addressed in this paper is the two category classification problem. Let $\{x_i, y_i\}$ be a given set of training examples where $x_i$ is the $i$-th input vector and $y_i$ is the target value; $y_i = 1$ denotes that $x_i$ is in class 1 and $y_i = -1$ denotes that $x_i$ is in class 2. In this paper I consider the SVM problem formulation that uses $L_2$ soft margin. This problem is usually converted (see Keerthi et al, 2000 for details) to the SVM problem with hard margin given by

$$\begin{array}{ll} \min & \frac{1}{2}\|w\|^2 \\ \text{s.t.} & y_i(w \cdot z_i + b) \geq 1 \;\; \forall i \end{array} \qquad (1)$$

where: $z_i$ denotes the transformed vector in the (modified) feature space;

$$z_i \cdot z_j = \tilde{K}(x_i, x_j) = K(x_i, x_j) + \frac{1}{C}\delta_{ij}; \tag{2}$$

$\delta_{ij} = 1$ if $i = j$, 0 otherwise; and $K$ is the kernel function. Popular choices for $K$ are:

$$\text{Gaussian kernel}: \quad K(x, \bar{x}) = \exp(-\frac{\|x - \bar{x}\|^2}{2\sigma^2}) \tag{3a}$$

$$\text{Polynomial kernel}: \quad K(x, \bar{x}) = (1 + \frac{x \cdot \bar{x}}{\sigma^2})^d \tag{3b}$$

In this paper I will focus mainly on the gaussian kernel. The solution of (1) is obtained by solving the dual problem,

$$\begin{array}{ll} \max & \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \tilde{K}(x_i, x_j) \\ \text{s.t.} & \sum_i y_i \alpha_i = 0 \quad \text{and} \quad \alpha_i \geq 0 \forall i \end{array} \tag{4}$$

At optimality, the objective functions in (1) and (4) are equal.

Tuning of hyperparameters such as $C$ and $\sigma$ is usually done by minimizing an estimate of generalization error such as the leave-one-out (LOO) error and the $k$-fold cross validation error. It was shown by Vapnik (Vapnik, 1998) that the following bound holds:

$$\text{LOO Error} \leq f \stackrel{\text{def}}{=} \frac{1}{m} R^2 \|w\|^2 \tag{5}$$

where $w$ is the solution of (1) and $R$ is the radius of the smallest sphere that contains all $z_i$ vectors. $R^2$ can be obtained as the optimal objective function value of the following problem (see Schölkopf et al, 1995 & Vapnik, 1998 for details):

$$\begin{array}{ll} \max & \sum_i \beta_i \tilde{K}(x_i, x_i) - \sum_{i,j} \beta_i \beta_j \tilde{K}(x_i, x_j) \\ \text{s.t.} & \sum_i \beta_i = 1 \quad \text{and} \quad \beta_i \geq 0 \ \forall i \end{array} \tag{6}$$

The right hand side bound in (5), $f$, is usually referred to as the *Radius/Margin* bound. Note that, both $R^2$ as well as $\|w^2\|$ depend on $C$ and $\sigma$ and hence $f$ is also a function of $C$ and $\sigma$. The first experiments on using the Radius/Margin bound for model selection were done by Schölkopf et al (Schölkopf et al, 1995). Recently Chapelle et al (Chapelle et al, 2000) used matrix based quadratic programming solvers for (1) and (6) to successfully demonstrate the usefulness of $f$ for tuning these (and other) hyperparameters.

Since, even for medium size problems with a few thousand examples it is difficult to load the entire kernel matrix of $K(x_i, x_j)$ values to the computer memory and do matrix operations on it, conventional, finitely terminating quadratic programming solvers are not very suitable for solving (1). Hence, specially designed iterative algorithms (Keerthi et al, 2000; Schölkopf et al, 1999) that are asymptotically converging are popular for solving (4) and (6). The use of these algorithms allows the easy tuning of hyperparameters in large scale problems. The main aim of this paper is to discuss implementation issues associated with this, and, use the resulting implementation to study the usefulness of the radius/margin bound on several benchmark problems.

It should be mentioned here that Cristianini et al (Cristianini et al, 1998) carried out the first set of experiments along the above lines. Their experiments were done on the hard margin problem without the $C$ parameter and threshold $b$. To solve (4), they employed the Kernel Adatron algorithm, which is extremely easy to implement, but very slow.[1]

## 2 Implementation Issues

I will assume throughout that $f$ is differentiable with respect to $C$ and $\sigma$.[2] To speed up the tuning it is appropriate to use a gradient based technique such as the BFGS Quasi-Newton algorithm to minimize $f$.

### 2.1 Evaluation of $f$

I have employed the nearest point algorithm given in Keerthi et al, 2000, for solving (4) and evaluating $\|w\|^2$. The numerical experiments of that paper show that this algorithm is very efficient for solving the hard margin problem in (1) and (4). To determine $R^2$ via (6), the SMO algorithm discussed in section 4 of Schölkopf et al, 1999, is very suitable. This algorithm was modified along the lines outlined in Keerthi et al, 1999, so that it runs very fast.

[1] In a personal communication, Colin Campbell has informed me that later he did some experiments using the $L_2$ soft margin, but he did not publish them.

[2] The contour plots given later in Figures 1-3 seem to indicate that this is a valid assumption.

## 2.2 Evaluation of gradient of $f$

The computation of gradient of $f$ requires the knowledge of the gradients of $\|w\|^2$ and $R^2$. Recently Chapelle et al (Chapelle et al, 2000) gave a very useful result (see Lemma 2 of their paper) which makes these gradient computations extremely easy once (4) and (6) are solved. It is important to appreciate the usefulness of their result, particularly from the viewpoint of this paper that, iterative, non-matrix-based techniques are used for solving (4) and (6). Clearly, $\|w\|^2$ depends on $\alpha$, and, $\alpha$ in turn depends on $C$ and $\sigma$. Yet, because $\|w\|^2$ itself is computed via an optimization problem (i.e., (4)), it turns out that the gradient of $\alpha$ with respect to the hyperparameters does not enter into the computation of the gradient of $\|w\|^2$. Since $R^2$ is also solved via an optimization problem (i.e, (6)), a similar result holds for $\beta$ and $R^2$.

**Remark 1.** The easiest way to appreciate the above result is to consider the function $f(u)$ given by $f(u) = \min_\alpha F(u, \alpha)$. Let $\tilde{\alpha}(u)$ denote the solution of the minimization problem, i.e., $\partial F / \partial \alpha = 0$ at $\alpha = \tilde{\alpha}(u)$. Now, $f(u) = F(u, \tilde{\alpha}(u))$. Hence,

$$\frac{\partial f}{\partial u} = \frac{\partial F}{\partial u} + \frac{\partial F}{\partial \alpha}|_{\alpha=\tilde{\alpha}(u)} \frac{\partial \tilde{\alpha}(u)}{\partial u} = \frac{\partial F}{\partial u} \tag{7}$$

Thus the gradient of $f$ with respect to $u$ can be obtained simply by differentiating $F$ with respect to $u$ as if $\alpha$ has no influence on $u$. The corresponding arguments for the constrained optimization problems in (4) and (6) is a bit more complicated. See Chapelle et al, 2000, for details.

It is easy to see that, if the gradients of $\alpha$ and $\beta$ with respect to the hyperparameters are to be determined, then matrix operations involving the kernel matrix are necessary. *Hence, Chapelle et al's result concerning the avoidance of these gradients in the evaluation of the gradients of $\|w\|^2$ and $R^2$ gives terrific support for the radius/margin criterion when iterative techniques are employed for solving (4) and (6).* For other criteria such as the LOO error, k-fold CV error etc, such an easy evaluation of gradient is ruled out. This issue is particularly very important when a large number of hyperparameters other than $C$ and $\sigma$ are also considered for tuning because, when the number of optimization variables is large, gradient-based optimization methods are many times faster than methods which use function values only.

Application of Chapelle et al's gradient calculations yields the following expressions.

$$\frac{\partial f}{\partial C} = \frac{1}{m}[\frac{\partial \|w\|^2}{\partial C} R^2 + \|w\|^2 \frac{\partial R^2}{\partial C}] \qquad \frac{\partial f}{\partial \sigma^2} = \frac{1}{m}[\frac{\partial \|w\|^2}{\partial \sigma^2} R^2 + \|w\|^2 \frac{\partial R^2}{\partial \sigma^2}] \tag{8}$$

$$\frac{\partial\|w\|^2}{\partial C} = \sum_i \alpha_i^2/C^2 \qquad \frac{\partial\|w\|^2}{\partial\sigma^2} = -\sum_{i,j} \alpha_i\alpha_j y_i y_j \frac{\partial\tilde{K}(x_i,x_j)}{\partial\sigma^2} \tag{9}$$

$$\frac{\partial R^2}{\partial C} = -\sum_i \beta_i(1-\beta_i)/C^2 \qquad \frac{\partial R^2}{\partial\sigma^2} = -\sum_{i,j} \beta_i\beta_j \frac{\partial\tilde{K}(x_i,x_j)}{\partial\sigma^2} \tag{10}$$

$$\frac{\partial\tilde{K}(x_i,x_j)}{\partial\sigma^2} = \tilde{K}(x_i,x_j)\frac{\|x_i-x_j\|^2}{2\sigma^4} \tag{11}$$

Thus, gradient of $f$ is cheaply computed once $f$ has been computed (since $\|w\|^2$, $\alpha$, $R^2$ and $\beta$ are all available).

**Remark 2.** Since iterative algorithms for (4) and (6) converge only asymptotically, a termination criterion is usually employed to terminate them finitely. This termination criterion has to be chosen with care for the following reason. Take, for example, the simplified scenario mentioned in Remark 1. Suppose $f(u)$ is to be evaluated at some given $u$. If, during the solution of $\min_\alpha F(u,\alpha)$, we use a termination criterion and only obtain $\bar{\alpha}$, which is an approximation of $\tilde{\alpha}(u)$. Since $\partial F/\partial\alpha|_{\alpha=\bar{\alpha}} \neq 0$, the last equality in (7) does not hold and hence, $\partial\alpha/\partial u$ is needed to compute $\partial f/\partial u$. If the effect of $\partial\alpha/\partial u$ has to be ignored then it is important to ensure that the termination criterion used in the solution of $\min_\alpha F(u,\alpha)$ is stringent enough to ensure that $\partial F/\partial\alpha|_{\alpha=\bar{\alpha}}$ is sufficiently small.

### 2.3 Variable Transformation

As suggested by Chapelle et al (Chapelle et al, 2000) I use

$$u_1 = \ln(C) \quad u_2 = \ln(\sigma^2) \tag{12}$$

as the variables for optimization instead of $C$ and $\sigma^2$. This is a common transformation usually suggested elsewhere in the literature too.

### 2.4 Choosing Initial Values of $C$ and $\sigma$

Unless we have some good knowledge about the problem, it is not easy to choose good initial values for $C$ and $\sigma$. I have experimented with two pairs of different initial conditions. The first one is:

$$C = 1\ , \quad \sigma^2 = 1 \tag{13}$$

Let $\rho$ denote the radius of the smallest sphere that contains all examples, i.e., the $x_i$'s. The second pair of initial conditions is

$$C = 1 \quad \sigma^2 = \rho^2 \tag{14}$$

In all the datasets tried in this paper, each component of the $x_i$'s is normalized to lie between -1 and 1. Hence, for all the numerical experiments, I have simply used $\rho^2 = n$ where $n$ is the dimension of $x_i$.[3] Detailed testing shows that, (14) gives better results than (13). There was one dataset (Splice) for which (13) actually failed. See Figure 3 for details.

### 2.5 Issues Associated With the Gradient Descent Algorithm

To minimize $f$ there are many choices for optimization methods. In this work, the BFGS Quasi-Newton algorithm (Shanno and Phua, 1980) has been used. A conjugate-gradient method was also tried, but it required many more $f$ evaluations than the BFGS algorithm. Since each $f$ evaluation is expensive (it requires the solution of (4) and (6)), the BFGS method was preferred.

Each optimization iteration involves the determination of a direction $d$ using the BFGS method. Then a line search is performed along that direction to look for a point that satisfies certain approximate conditions associated with the following problem:

$$\min_{\gamma \geq 0} f(u + \gamma d) \tag{15}$$

Since gradient of $f$ is easily computed once $f$ is obtained, it is effective to use a line search technique that uses both function values and gradients. The code in Shanno and Phua, 1980, employs such a technique. For the BFGS algorithm, $\gamma = 1$ is a natural choice to try as the first step size in each line search. This choice is so good that, the line search usually attempts only one or two values of $\gamma$ before successfully terminating an optimization iteration. Truly, the goodness of the choice of $\gamma = 1$ is expected to hold strongly as the minimizer of $f$ is approached since the BFGS direction $d$ approaches the Newton direction. However, this does not happen in our case for the following reason. As the minimizer is approached, the gradient values are small and the effect of errors associated with the solution of (4) and (6) on the gradient evaluation and hence on the choice of $d$ is strong. Thus, the line search sometimes requires many evaluations of $f$ and $\nabla f$ in the end steps. *In numerical experiments it was observed that reaching the minimizer of $f$ too closely is not important for arriving at good values of hyperparameters.* Hence, it is a

[3] In the case of Adult-7 dataset, each $x_i$ has only 15 non-zero entries. Hence $\rho^2$ is set to 15 for that example.

good idea to terminate the line search (as well as the optimization process) if more than ten values of $\gamma$ have been attempted in that line search.

The optimization process generates a sequence of points in the space of hyperparameters. Successive points attempted by the process are usually located 'not-so-far-off' from each other. It is important to take this factor to advantage in the solution of (4) and (6). Thus if $\alpha$ and $\beta$ denote the solution of (4) and (6) at some $u$ and the optimization process next tries a new point, $\tilde{u}$, then $\alpha$ and $\beta$ are used to obtain good starting points for the solution of (4) and (6) at $\tilde{u}$. This gives significant gains in computational time. Since the constraints in (6) do not depend on the hyperparameters, $\beta$ can be directly carried over for the solution of (6) at $\tilde{u}$. For (4), we already said that the nearest point formulation in Keerthi et al, 2000, is employed. Since the constraints in the nearest point formulation are also independent of the hyperparameters, carrying over the variables for solution at $\tilde{u}$ is easy for the nearest point algorithm too.

The choice of criterion for terminating the optimization process is also very important. As already said above, reaching the minimizer of $f$ too closely is not crucial. Hence the criterion used can be loose. The following choice has worked quite well. Suppose BFGS starts an optimization iteration at $u$, then successfully completes a line search and reaches the next point $\tilde{u}$. Optimization is terminated if the following holds:

$$|f(\tilde{u}) - f(u)| \leq 10^{-5} f(u) \tag{16}$$

Typically, the complete optimization process uses only 5 to 20 $f$ evaluations.

## 3 Computational Experiments

An experimental version of my code, running on Matlab interface through the mex facility, is available from the following homepage:

*http://guppy.mpe.nus.edu.sg/˜mpessk/nparm.shtml*

I have numerically tested it on several benchmark datasets given in Rätsch, 1999. To test the usefulness of the code for solving large scale problems, I have also tested it on the Adult-7 dataset in (Platt, 1998). Some important quantities associated with these datasets and which describe the code's performance are given in Table 1. For all datasets except Image, Figures 1, 2 and

Table 1: Performance of the code on the data sets considered. Here: $n$ = number of input variables; $m$ = number of examples; $m_{\text{test}}$ is the number of test examples; $C$ and $\sigma^2$ are the final hyperparameter values; $m_{\text{supp}}$ is the final number of support vectors; $f$ is the optimal value of $R^2\|w\|^2/m$; and TestErr is the percentage error on the test set.

| DATA | $n$ | $m$ | $m_{\text{test}}$ | $C$ | $\sigma^2$ | $m_{\text{supp}}$ | $100f$ | TestErr |
|---|---|---|---|---|---|---|---|---|
| ADULT-7 | 123 | 16100 | 30956 | 0.077 | 5.98 | 11282 | 46.38 | 14.97 |
| BREAST CANCER | 9 | 200 | 77 | 0.122 | 8.93 | 200 | 75.13 | 28.57 |
| DIABETIS | 8 | 468 | 300 | 0.135 | 6.64 | 447 | 71.86 | 24.33 |
| FLARE-SOLAR | 9 | 666 | 400 | 0.057 | 0.533 | 666 | 88.07 | 35.00 |
| GERMAN | 20 | 700 | 300 | 0.118 | 12.4 | 696 | 75.99 | 19.67 |
| HEART | 13 | 170 | 100 | 0.618 | 34.7 | 147 | 56.92 | 19.00 |
| THYROID | 5 | 140 | 75 | 0.871 | 1.27 | 81 | 27.50 | 2.67 |
| TITANIC | 3 | 150 | 2051 | 0.111 | 2.49 | 150 | 69.77 | 22.91 |
| SPLICE | 60 | 1000 | 2175 | 1348 | 21.6 | 864 | 60.10 | 10.16 |
| BANANA | 2 | 400 | 4900 | 0.404 | 0.195 | 325 | 45.57 | 11.45 |
| IMAGE | 18 | 1300 | 1010 | 1.56 | 0.747 | 853 | 44.45 | 2.97 |

the left hand side of Figure 3 show the sequence of points generated by the BFGS optimization method on plots in which contours with equal $f$ values are drawn for various values of $f$. In the case of Splice and Banana datasets, for which the sizes of test sets are large, the right hand side plots of Figure 3 show contours of test set error. These are given to point out how good the radius/margin criterion is.

### 3.1 Danger of Using the $R = 1$ Approximation

When the gaussian kernel function is used, to simplify computations, the approximation $R = 1$ is sometimes tried. I did some experiments to check the usefulness of this approximation. For four datasets, Figure 4 shows the variation of $R^2\|w\|^2$, $\|w\|^2$ and test set error with respect to $\sigma^2$ for fixed values of $C$. It is clear that all three functions are quite well correlated and hence, as far as the tuning of $\sigma^2$ is concerned, using $R = 1$ seems to be a good approximation to make. This agrees with the observation made by Cristianini et al (Cristianini et al, 1998).

However, using $R = 1$ for tuning $C$ is dangerous. Figure 5 shows the variation of $R^2\|w\|^2$, $\|w\|^2$ and test set error with respect to $C$ for fixed values of $\sigma^2$. (Note using (9) that $\|w\|^2$ is always increasing with $C$.) Clearly $\|w\|^2$ alone is inadequate for the determination of $C$.

## 4 Conclusion

In this paper I have discussed various implementation issues associated with the determination of $C$ and $\sigma^2$ for the SVM $L_2$ soft margin problem (with gaussian kernel function) by minimizing the radius/margin criterion and employing iterative techniques for obtaining radius and margin. The experiments indicate the usefulness of the radius/margin criterion. In the near future I plan to extend the implementation to the tuning of hyperparameters associated with feature selection, along the lines indicated by Chapelle et al (Chapelle et al, 2000).

## References

O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, (2000). *Choosing kernel parameters for support vector machines*, Submitted to Machine Learning. Online version is at: *http://www.ens-lyon.fr/˜ochapell/*

C. Cortes and V. Vapnik, (1995). *Support vector networks*, Machine Learning, Vol 20, pp. 273-297.

N. Cristianini, C. Campbell, and J. Shawe-Taylor, (1999). *Dynamically adapting kernels in support vector machines*, In *Advances in Neural Information Processing Systems*, Vol 10. Online version: *http://lara.enm.bris.ac.uk/cig/pubs/1999/nips98.ps.gz*

S.S. Keerthi, S.K. Shevade, C. Bhattacharyya, and K. R. K. Murthy, (2000). *A fast iterative nearest point algorithm for support vector machine classifier design*, IEEE Transactions on Neural Networks, Vol 11, pp. 124-136.

S.S. Keerthi, S.K. Shevade, C. Bhattacharyya, and K. R. K. Murthy, (1999). *Improvements to Platt's SMO algorithm for SVM design.* Technical Report, CD-99-14, Dept. of Mechanical and Production Engineering, National University of Singapore, Singapore. Online version is at: *http://guppy.mpe.nus.edu.sg/˜mpessk/smo_mod.shtml*

J. Platt, (1998). *http://www.research.microsoft.com/˜jplatt/smo.html*

G. Rätsch, (1999). *http://ida.first.gmd.de/˜raetsch/data/benchmarks.htm*

B. Schölkopf, C. Burges, and V. Vapnik, (1995). *Extracting support data for a given task*,

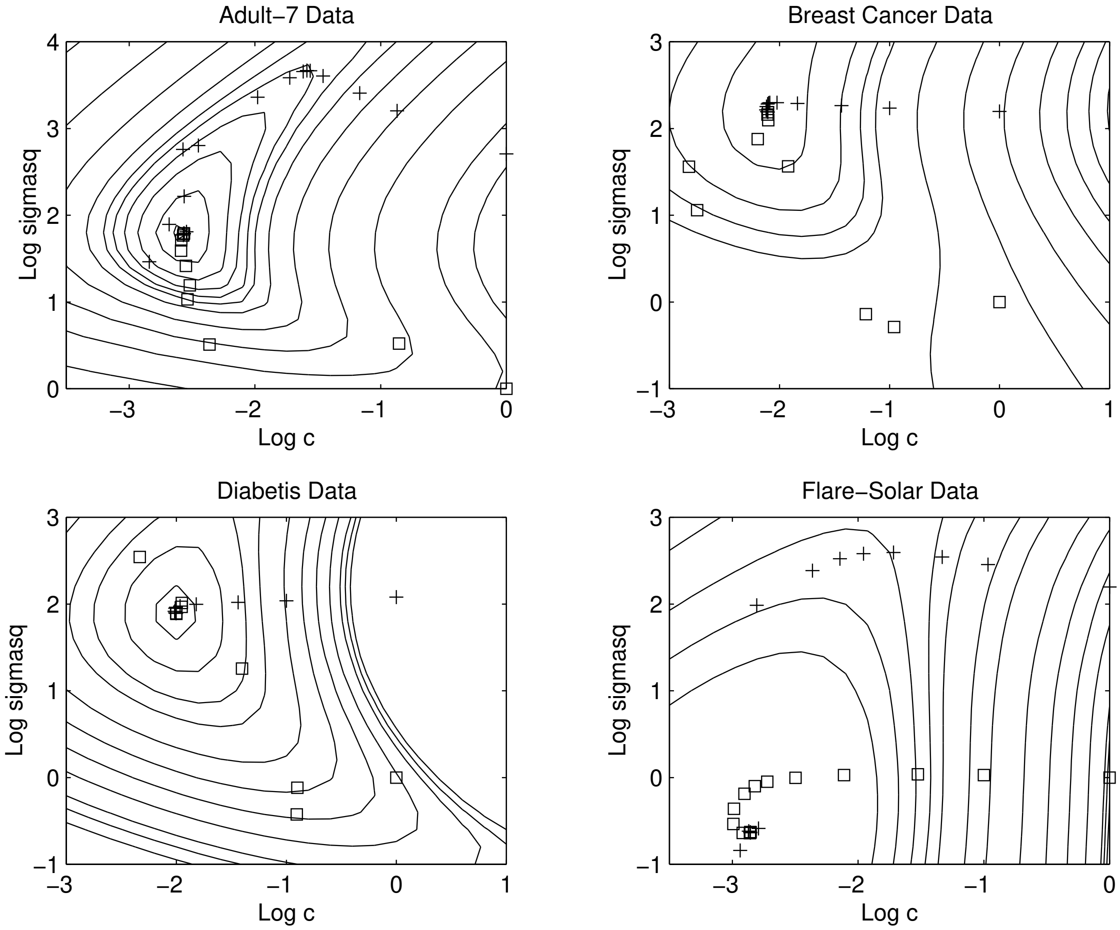


Figure 1: Radius/Margin contour plots for Adult-7, Breast Cancer, Diabetis, and Flare-Solar datasets. □ and + respectively denote points generated by the BFGS algorithm starting from the initial conditions in (13) and (14).

In U. M. Fayyad and R. Uthurusamy, editors, Proceedings, First International Conference on Knowledge Discovery & Data Mining, Menlo Park. AAAI Press.

B. Schölkopf, J.C. Platt, J. Shawe-Taylor, and A.J. Smola, (1999). *Estimating the support of a high dimensional distribution*, Technical Report MSR-TR-99-87, Microsoft Research, Redmond, WA, USA. Online version: *http://www.kernel-machines.org/papers/oneclass-tr.ps.gz*

D.F. Shanno and K.H. Phua, (1980). *Minimization of unconstrained multivariate functions*, ACM Transactions on Mathematical Software, Vol 6, pp. 618-622.

V. Vapnik, (1998). *Statistical Learning Theory*, John Wiley & Sons.

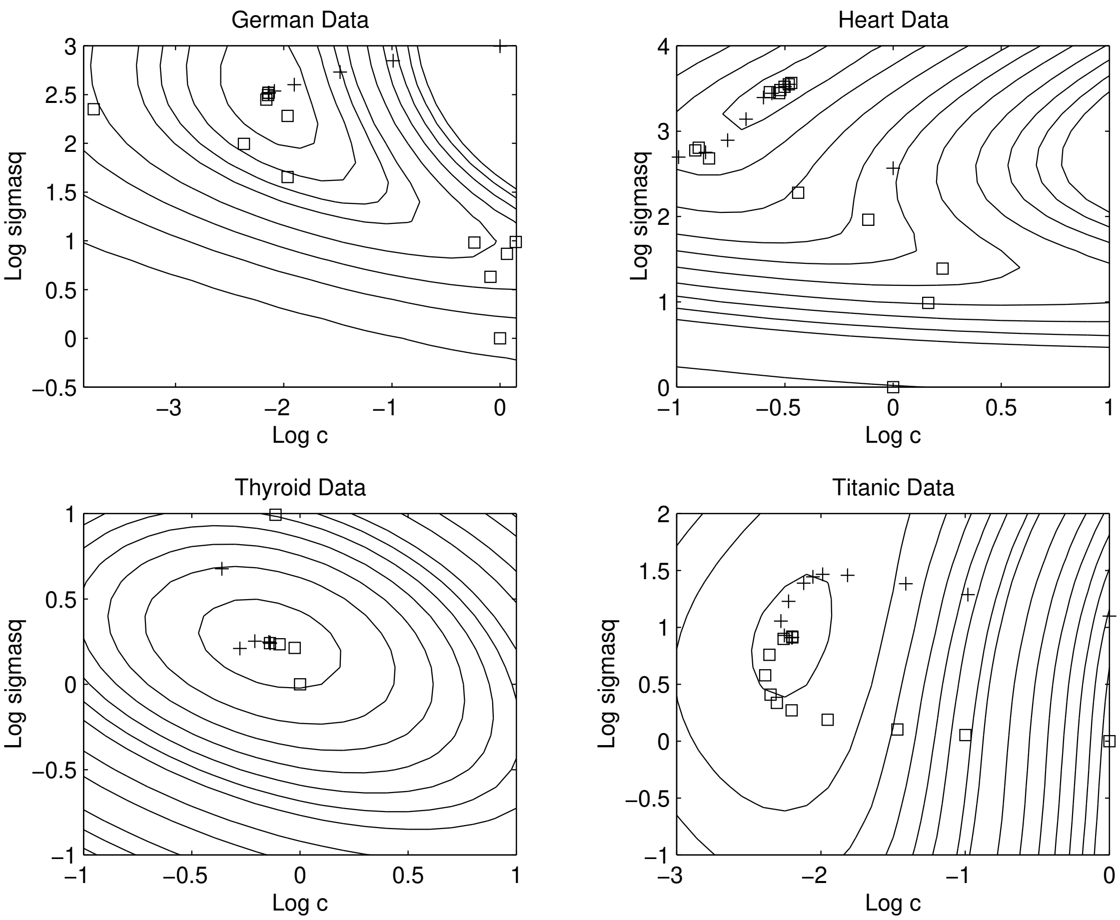


Figure 2: Radius/Margin contour plots for German, Heart, Thyroid, and Titanic datasets. □ and + respectively denote points generated by the BFGS algorithm starting from the initial conditions in (13) and (14).

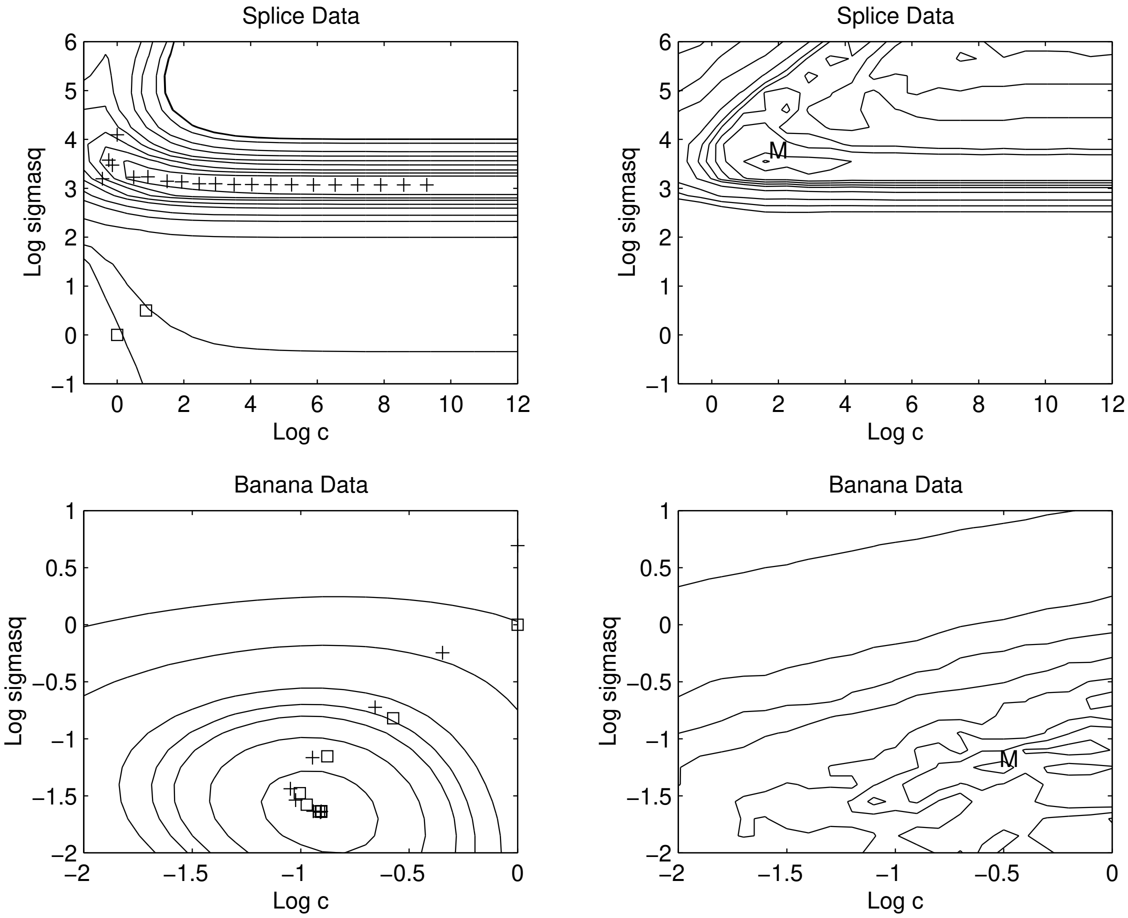


Figure 3: The two figures on the left hand side give Radius/Margin contour plots for Splice and Banana datasets. □ and + respectively denote points generated by the BFGS algorithm using the initial conditions in (13) and (14). In the case of Splice dataset, for initial condition (13), optimization was terminated after two $f$ evaluations since a very large $C$ value was attempted at the third $f$ evaluation point. The two figures on the right hand side give contour plots of test set error. In these two plots M denotes the location of the point of least test set error.

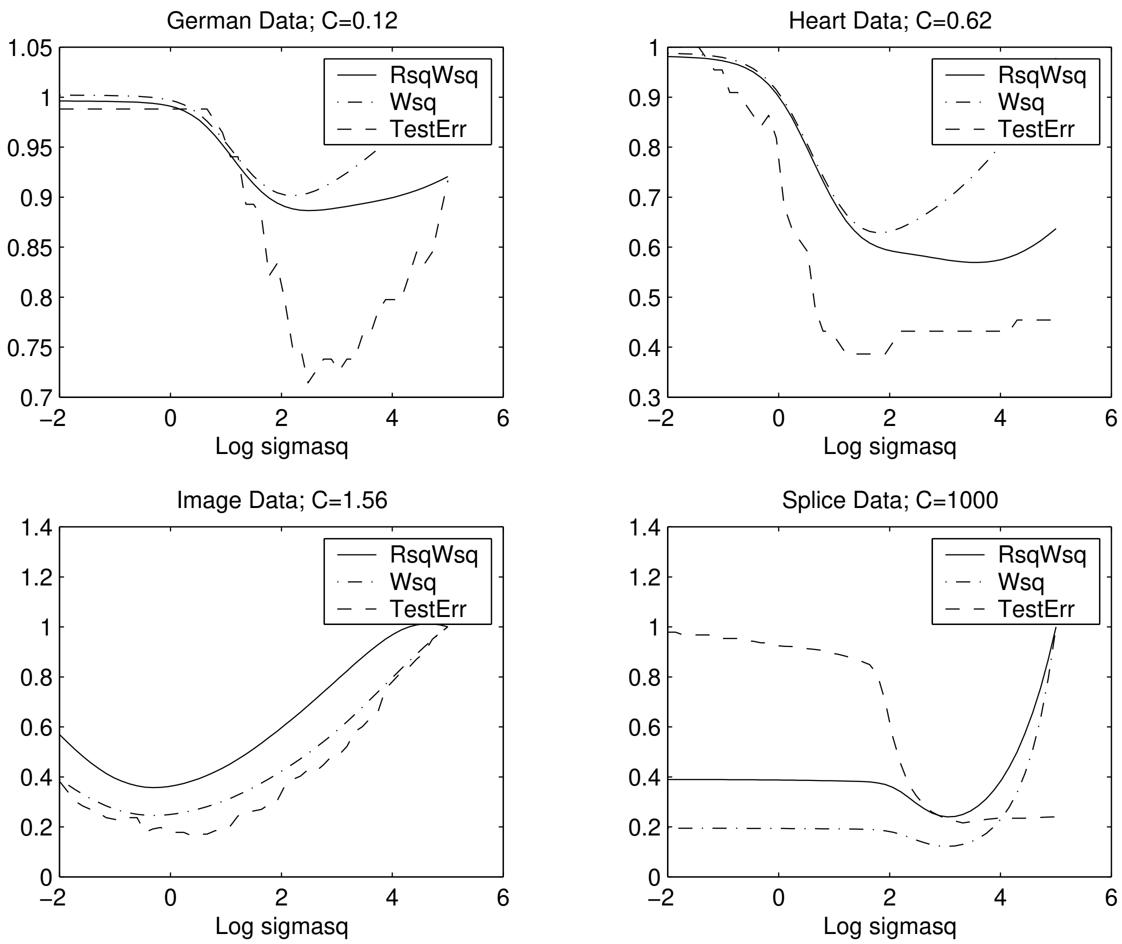


Figure 4: Variation of $R^2\|w\|^2$, $\|w\|^2$ and TestErr with respect to $\sigma^2$ for fixed $C$ values. In each graph, the vertical axis is normalized differently for $R^2\|w\|^2$, $\|w\|^2$ and TestErr.

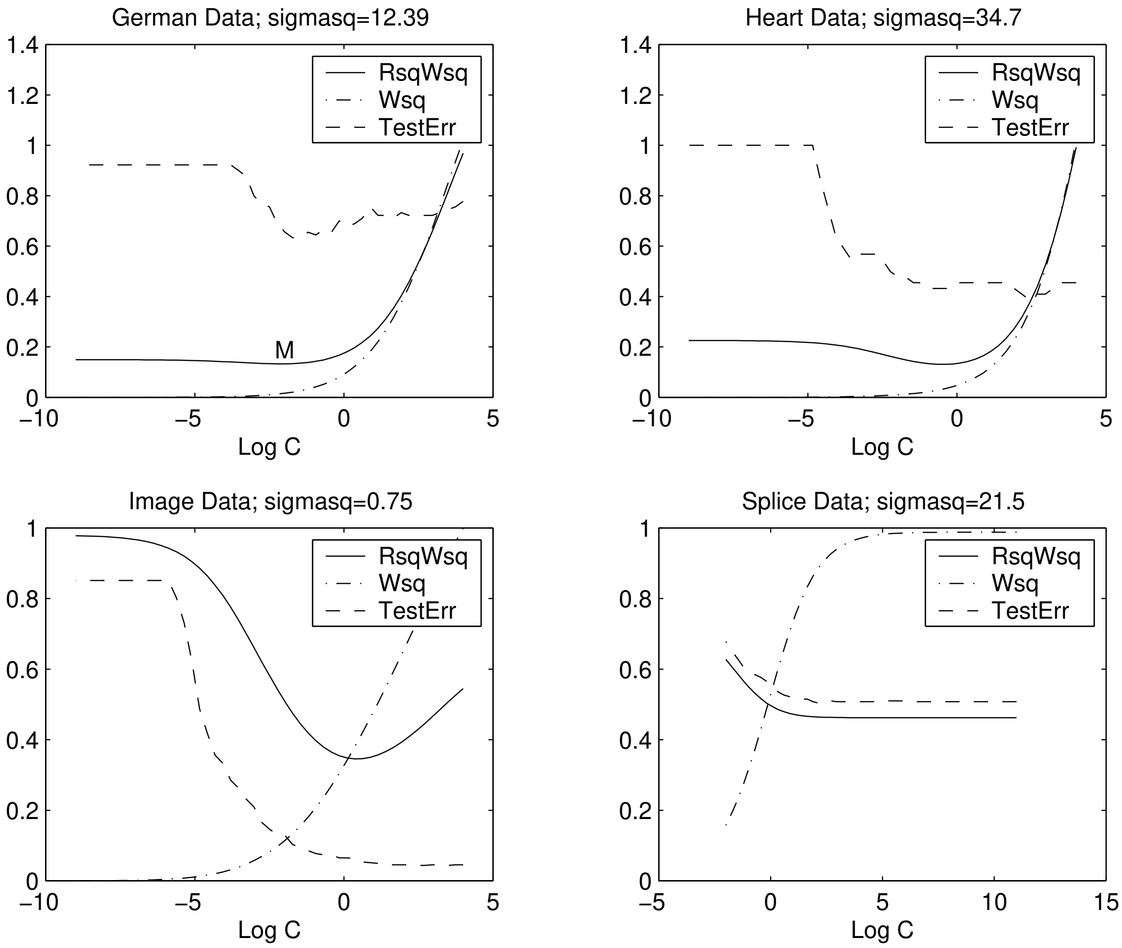


Figure 5: Variation of $R^2\|w\|^2$, $\|w\|^2$ and TestErr with respect to $C$ for fixed $\sigma^2$ values. In the case of German dataset, the minimum of the $R^2\|w\|^2$ function is marked as M since it is not visible. In each graph, the vertical axis is normalized differently for $R^2\|w\|^2$, $\|w\|^2$ and TestErr.