2007 International Symposium on Nonlinear Theory and its Applications
NOLTA'07, Vancouver, Canada, September 16-19, 2007

A New Working Set Selection for Decomposition-Type SVM Learning Algorithms

Norikazu Takahashi†, Masashi Kuranoshita‡, Yusuke Kawazoe‡, Jun Guo† and Jun'ichi Takeuchi†

†Faculty of Information Science and Electrical Engineering, Kyushu University
‡Graduate School of Information Science and Electrical Engineering, Kyushu University
744 Motooka, Nishi-ku, Fukuoka 819-0395 Japan
Email: norikazu@csce.kyushu-u.ac.jp

Abstract—Decomposition methods are efficient iterative techniques for solving large quadratic programming (QP) problems arising in support vector machines. In each step, the decomposition method chooses a set of a small number of variables called the working set and then solves the QP problem with respect to those selected variables. In this paper, we propose a new working set selection method based on the conjugate gradient method and evaluate its effectiveness by using benchmark data sets on both pattern classification and regression problems.

1. Introduction

Since the training of a support vector machine (SVM) [1] is formulated as a quadratic programming (QP) problem, computational cost becomes very high for large training samples. In order to overcome this difficulty, decomposition methods have been proposed by several researchers [2]-[5]. A decomposition method tries to find an optimal solution of the original QP problem by executing the following two operations iteratively: 1) selecting a set of a small number of variables called the working set, and 2) solving the QP problem with respect to the selected variables. Since each subproblem is small, decomposition methods spend much less amount of memory than the traditional QP solvers. In addition, it is often the case that even though sufficiently large amount of memory is available, decomposition methods are faster than the traditional QP solvers.

Each decomposition method is characterized by three components: how to select the working set, how to solve subproblems, and the termination criterion. For example, in the sequential minimal optimization (SMO) algorithm proposed by Platt [2] only two variables are selected in each step and subproblems are solved analytically. On the other hand, in SVM$^{\text{light}}$ proposed by Joachims [3] a fixed number, which can be any even number less than the number of training samples, of variables are selected for the working set in a systematic way based on the steepest descent (SD) method, and subproblems are solved by using one of the traditional QP solvers.

In this paper, we concentrate our attention on how to select the working set, which is the most important factor to determine the computation time of the decomposition method, and propose a novel working set selection method based on the conjugate gradient (CG) method. We will first show that the decomposition method with the proposed working set selection always stops within a finite number of iterations after finding an optimal solution. We will then show experimental results on several benchmark data sets from which we see that the proposed method is sometimes much faster than SVM$^{\text{light}}$.

2. QP Problems in SVM Learning

2.1. Pattern Classification

Let $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{l}$ be the given training samples, where $\boldsymbol{x}_i \in \mathbb{R}^n$ and $y_i \in \{1, -1\}$ represent the i-th pattern and its class label, respectively. Then the training of an SVM leads to the following QP problem [1].

Problem 1 Find $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_l]^T$ which minimizes

$$W(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \sum_{i=1}^{l}\alpha_i \tag{1}$$

under constraints:

$$\sum_{i=1}^{l}\alpha_i y_i = 0 \tag{2}$$

$$0 \le \alpha_i \le C, \quad i = 1, 2, \ldots, l \tag{3}$$

where $K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is a kernel function and C is a positive constant.

Throughout this paper, we will assume both C and the kernel function K are fixed a priori. We will also assume that K satisfies Mercer's condition. It is thus guaranteed that Problem 1 is a convex QP problem.

As the kernel function satisfying Mercer's condition, RBF (radial basis function) kernel defined by

$$K(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2\sigma^2}\right) \tag{4}$$

and the polynomial kernel defined by

$$K(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x}^T\boldsymbol{x}' + 1)^p \tag{5}$$

are well known and widely used in SVMs.

Let $\boldsymbol{\alpha}^* = \left[\alpha_1^*, \alpha_2^*, \ldots, \alpha_l^*\right]^T$ be any optimal solution of Problem 1. Then the decision function of the SVM is expressed as

$$f(\boldsymbol{x}) = \mathrm{sgn}\left(\sum_{i=1}^{l} \alpha_i^* y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b^*\right) \tag{6}$$

where

$$b^* = y_{j_0} - \sum_{i=1}^{l} \alpha_i^* y_i K(\boldsymbol{x}_i, \boldsymbol{x}_{j_0})$$

and j_0 is any j satisfying $0 < \alpha_j^* < C$. Pattern $\boldsymbol{x}$ is classified into Class 1 and Class -1 if $f(\boldsymbol{x}) = 1$ and $f(\boldsymbol{x}) = -1$, respectively.

Since Problem 1 is a QP problem, a feasible solution is optimal iff Karush-Kuhn-Tucker (KKT) condition is satisfied. Let us define two sets $I_{\mathrm{up}}(\boldsymbol{\alpha})$ and $I_{\mathrm{low}}(\boldsymbol{\alpha})$ as

$$\begin{aligned} I_{\mathrm{up}}(\boldsymbol{\alpha}) &= \{i \,|\, \alpha_i < C,\, y_i = 1\} \cup \{i \,|\, \alpha_i > 0,\, y_i = -1\} \\ I_{\mathrm{low}}(\boldsymbol{\alpha}) &= \{i \,|\, \alpha_i < C,\, y_i = -1\} \cup \{i \,|\, \alpha_i > 0,\, y_i = 1\} \end{aligned}$$

and the function $F_i(\boldsymbol{\alpha})$ as

$$F_i(\boldsymbol{\alpha}) = y_i \frac{\partial W(\boldsymbol{\alpha})}{\partial \alpha_i} = y_i \left(\sum_{j=1}^{l} \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) - 1\right). \tag{7}$$

Then the KKT condition can be rewritten as follows [4]:

$$\min_{i \in I_{\mathrm{up}}(\boldsymbol{\alpha})} F_i(\boldsymbol{\alpha}) \geqq \max_{i \in I_{\mathrm{low}}(\boldsymbol{\alpha})} F_i(\boldsymbol{\alpha}) \tag{8}$$

2.2. Regression

Suppose that the relationship between two variables $\boldsymbol{x} \in \mathbb{R}^n$ and $y \in \mathbb{R}$ is expressed as $y = f^*(\boldsymbol{x})$ where $f^*(\boldsymbol{x})$ is a certain nonlinear function. Regression problem is to estimate the function $f^*(\boldsymbol{x})$ from given l training samples $(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_l, y_l)$ where y_i usually contains noise independent of $\boldsymbol{x}_i$.

Basic principle of SVMs for pattern classification can easily be applied to regression and this technique is called the support vector regression (SVR). When we use the ε-insensitive loss function defined by

$$\xi_i = \begin{cases} 0, & \text{if } |y_i - f(\boldsymbol{x}_i)| \leq \varepsilon \\ |y_i - f(\boldsymbol{x}_i)| - \varepsilon, & \text{otherwise} \end{cases}$$

SVR is formulated as the following QP problem [1].

Problem 2 Find $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_{2l}]^T$ which minimizes

$$\begin{aligned} W(\boldsymbol{\alpha}) = \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} (\alpha_i - \alpha_{i+l})(\alpha_j - \alpha_{j+l}) K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ - \sum_{i=1}^{l} y_i (\alpha_i - \alpha_{i+l}) + \varepsilon \sum_{i=1}^{l} (\alpha_i + \alpha_{i+l}) \end{aligned} \tag{9}$$

under constraints:

$$\sum_{i=1}^{l} (\alpha_i - \alpha_{i+l}) = 0 \tag{10}$$

$$0 \leq \alpha_i \leq C, \quad i = 1, 2, \ldots, 2l \tag{11}$$

3. Decomposition Methods

In this section, we focus our attention on the decomposition method for Problem 1, and explain the basic algorithm and the termination criterion. The following techniques can be easily applied to Problem 2.

3.1. Basic Algorithm of Decomposition Method

The algorithm of a general decomposition method is expressed as follows:

Algorithm 1
Step 1: Set $\boldsymbol{\alpha}(0) = \mathbf{0}$ and $k = 0$.
Step 2: If $\boldsymbol{\alpha} = \boldsymbol{\alpha}(k)$ satisfies the termination criterion then stop.
Step 3: Choose $q\,(\leq n)$ elements $i_1, i_2, \ldots, i_q$ from the set $L = \{1, 2, \ldots, l\}$ and set $L_B(k) = \{i_1, i_2, \ldots, i_q\}$.
Step 4: Find $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_l]^T$ which minimizes $W(\boldsymbol{\alpha})$ under the constraints (2), (3) and $\alpha_i = \alpha_i(k)$, $\forall i \in L_N$ where $L_N(k) = L \backslash L_B(k)$.
Step 5: Let $\boldsymbol{\alpha}^*$ be an optimal solution of the QP subproblem in Step 4 and set $\boldsymbol{\alpha}(k+1) = \boldsymbol{\alpha}^*$.
Step 6: Add 1 to k and go to Step 2.

For example, SMO algorithm is a decomposition method with $q = 2$ and thus QP subproblems can be solved analytically. In general, q is set to an integer greater than or equal to 2 and QP subproblems are solved by using one of traditional QP solvers. Also, as the termination criterion, a relaxed version of the optimality condition (8) is used as described in the next subsection.

3.2. Termination Criterion of Algorithm 1

It is apparent that $\boldsymbol{\alpha}(k)$ in Algorithm 1 belongs to the feasible region of Problem 1 for all k and that $W(\boldsymbol{\alpha}(k))$ is non-increasing with respect to k. Hence the value of $W(\boldsymbol{\alpha}(k))$ converges to a certain constant. However, this does not imply that $\boldsymbol{\alpha}(k)$ converges to an optimal solution. In fact, if the working set selection in Step 3 is not appropriate then $\boldsymbol{\alpha}(k)$ may not change. We will show below some definitions introduced in [5] and a new convergence theorem for Algorithm 1.

Definition 1 ([5]) The condition

$$\min_{i \in I_{\mathrm{up}}^{\delta}(\boldsymbol{\alpha})} F_i(\boldsymbol{\alpha}) > \max_{i \in I_{\mathrm{low}}^{\delta}(\boldsymbol{\alpha})} F_i(\boldsymbol{\alpha}) - \tau \tag{12}$$

is called the (τ, δ)-optimality condition, where

$$I_{\mathrm{up}}^{\delta}(\boldsymbol{\alpha}) = \{i \,|\, \alpha_i \leq C - \delta, y_i = 1\} \cup \{i \,|\, \alpha_i \geq \delta, y_i = -1\}$$

$$I^{\delta}_{\mathrm{low}}(\boldsymbol{\alpha}) = \{i \,|\, \alpha_i \le C - \delta, y_i = -1\} \cup \{i \,|\, \alpha_i \ge \delta, y_i = 1\}$$

τ is any positive number, and δ is any positive number smaller than $C/2$. A feasible solution $\boldsymbol{\alpha}$ is called a (τ, δ)-optimal solution if it satisfies (12).

Definition 2 ([5]) Given a feasible solution $\boldsymbol{\alpha}$, a pair of indices (i, j) satisfying $i \in I^{\delta}_{\mathrm{up}}(\boldsymbol{\alpha})$, $j \in I^{\delta}_{\mathrm{low}}(\boldsymbol{\alpha})$, and $F_i(\boldsymbol{\alpha}) \le F_j(\boldsymbol{\alpha}) - \tau$ is called a (τ, δ)-violating pair at $\boldsymbol{\alpha}$.

Theorem 1 Suppose that the (τ, δ)-optimality condition is used as the termination criterion of Algorithm 1. Suppose also that there exists a nonnegative integer M such that for any k there exists an r satisfying $k \le r \le k + M$ and the condition that $L_B(r)$ contains a (τ, δ)-violating pair at $\boldsymbol{\alpha}(r)$. Then Algorithm 1 stops within a finite number of iterations for any $\tau > 0$ and any $\delta \in (0, C/2)$.

We will omit the proof of Theorem 1 because it is similar to the proof of Theorem 4 in [5].

4. Proposed Working Set Selection Method

We now propose a novel working set selection based on the conjugate gradient (CG) method. The CG method is an effective technique for solving QP problems without constraints. In particular, if the objective function is strictly convex then the CG method converges to the unique optimal solution after l steps, where l is the number of variables. Let $h(\boldsymbol{\alpha})$ be the objective function. Then the search direction $\boldsymbol{d}(k)$ at the k-th step of the CG method is determined by

$$\boldsymbol{d}(k) = -\nabla h(\boldsymbol{\alpha}(k)) + \beta(k)\boldsymbol{d}(k-1), \quad k = 0, 1, 2, \ldots$$

where $\beta(k)$ is determined by

$$\beta(0) = 0, \quad \beta(k) = \frac{\|\nabla h(\boldsymbol{\alpha}(k))\|^2}{\|\nabla h(\boldsymbol{\alpha}(k-1))\|^2}, \quad k = 1, 2, \ldots$$

By introducing this idea to the decomposition method, we can obtain the following algorithm.

Algorithm 2
Step 1: Set $k = 0$, $\boldsymbol{\alpha}(0) = \mathbf{0}$ and $m(0) = 0$.
Step 2: If $\boldsymbol{\alpha} = \boldsymbol{\alpha}(k)$ satisfies the (τ, δ)-optimality condition then stop.
Step 3: Compute $\beta(k)$ by

$$\beta(k) = \begin{cases} 0, & \text{if} \quad m(k) = 0 \\ \frac{\|\nabla W(\boldsymbol{\alpha}(k))\|^2}{\|\nabla W(\boldsymbol{\alpha}(k-1))\|^2}, & \text{if} \quad 1 \le m(k) \le l-1 \end{cases}$$

Step 4: Compute $\boldsymbol{d}(k) = -\nabla W(\boldsymbol{\alpha}(k)) + \beta(k)\boldsymbol{d}(k-1)$.
Step 5: Find $L_B(k)$ by Steps 5.1 to 5.4.
Step 5.1: Set $L_B(k) = \emptyset$.
Step 5.2: Compute $y_i d_i(k)$ for all i and sort $\{1, 2, \ldots, l\}$ according to the value of $y_i d_i(k)$ in increasing order.
Step 5.3: Choose $q/2$ i's belonging to $I^{\delta}_{\mathrm{low}}(\boldsymbol{\alpha}(k))$ from the top of the sorted list and add them to $L_B(k)$.
Step 5.4: Choose $q/2$ i's belonging to $I^{\delta}_{\mathrm{up}}(\boldsymbol{\alpha}(k))$ from the bottom of the sorted list and add them to $L_B(k)$.
Step 6: If $L_B(k) = L_B(k-1)$ then set $\boldsymbol{\alpha}(k+1) = \boldsymbol{\alpha}(k)$, set $m(k+1) = 0$, and go to Step 10.
Step 7: Find $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_l]^T$ which minimizes $W(\boldsymbol{\alpha})$ under the constraints (2), (3) and $\alpha_i = \alpha_i(k)$, $\forall i \in L_N(k)$ where $L_N(k) = L \backslash L_B(k)$.
Step 8: Let $\boldsymbol{\alpha}^*$ be an optimal solution of the QP subproblem in Step 7 and set $\boldsymbol{\alpha}(k+1) = \boldsymbol{\alpha}^*$.
Step 9: If $m(k) \le l - 2$ then set $m(k+1) = m(k) + 1$. Otherwise set $m(k+1) = 0$.
Step 10: Add 1 to k and go to Step 2.

The difference between Algorithm 2 and SVM$^{\mathrm{light}}$ is only the computation of the search direction $\boldsymbol{d}(k)$. In SVM$^{\mathrm{light}}$, $\boldsymbol{d}(k)$ is always set to the direction of the steepest descent, that is, $\boldsymbol{d}(k) = -\nabla W(\boldsymbol{\alpha}(k))$ for all k. Therefore, the computational cost for one iteration in Algorithm 2 is a little higher than that in SVM$^{\mathrm{light}}$.

Theorem 2 Algorithm 2 stops within a finite number of iterations for any $\tau > 0$ and $\delta \in (0, C/2)$.

Proof. It is obvious from Step 9 that $m(k)$ is set to zero at least once in every l iterations. Assume that $m(k) = 0$ and $\boldsymbol{\alpha}(k)$ is not a (τ, δ)-optimal solution. In this case, $y_i d_i(k)$ is equal to $-F_i(\boldsymbol{\alpha}(k))$ where $F_i(\boldsymbol{\alpha}(k))$ is defined by (7). Hence $\{1, 2, \ldots, l\}$ are sorted in Step 5.2 according to the value of $F_i(\boldsymbol{\alpha}(k))$ in decreasing order. Let i_1 be the element of $I^{\delta}_{\mathrm{low}}(\boldsymbol{\alpha}(k))$ which is first chosen in Step 5.3. Also let i_2 be the element of $I^{\delta}_{\mathrm{up}}(\boldsymbol{\alpha}(k))$ which is first chosen in Step 5.4. Since there exists at least one (τ, δ)-violating pair at $\boldsymbol{\alpha}(k)$, two indices i_1 and i_2 satisfy

$$i_1 \in I^{\delta}_{\mathrm{low}}(\boldsymbol{\alpha}(k)),\ i_2 \in I^{\delta}_{\mathrm{up}}(\boldsymbol{\alpha}(k)),\ F_{i_1}(\boldsymbol{\alpha}(k)) - \tau \ge F_{i_2}(\boldsymbol{\alpha}(k)),$$

that is, (i_2, i_1) is a (τ, δ)-violating pair at $\boldsymbol{\alpha}(k)$. From these observations, we can conclude that for any k there exists an r such that $k \le r \le k + l - 1$ and $L_B(r)$ contains at least one (τ, δ)-violating pair at $\boldsymbol{\alpha}(r)$. Therefore it follows from Theorem 1 that Algorithm 2 terminates within a finite number of iterations for any $\tau > 0$ and $\delta \in (0, C/2)$. □

5. Experiments

In order to evaluate the effectiveness of the proposed method, we have implemented both Algorithm 2 and SVM$^{\mathrm{light}}$ in Scilab[1] and applied to various benchmark data sets. RBF kernels are used in all experiments. All experiments were carried out on a PC with dual processor Xeon 2.8GHz and 2GB RAM. Comparisons are made in terms of the CPU time and the number of iterations.

We have first applied Algorithm 2 and SVM$^{\mathrm{light}}$ to 13 kinds of benchmark data sets on pattern classification problems [6]. Table 1 shows the overview of the benchmark data sets: n is the dimension of patterns; l the number of

[1] `http://www.scilab.org/`

Table 1: Overview of the benchmark data sets [6]

Data	n	l	C	σ^2
Banana	2	400	316.2	1.0
Breast-Cancer	9	200	15.19	50
Diabetis	8	468	100	20
Flare-Solar	9	666	1.02	30
German	20	700	3.16	55
Heart	13	170	3.16	120
Image	18	300	500	30
Ringnorm	20	400	10^9	10
Splice	60	150	1000	70
Thyroid	5	140	10	3
Titanic	3	150	1.0	1.0
TwoNorm	20	400	3.162	40
Waveform	21	400	1.0	20

Table 2: Comparison between Algorithm 2 (CG) and SVM$^{\text{light}}$ (SD) by CPU time and the number of iterations.

	CPU time [sec]		Iterations	
	CG	SD	CG	SD
Banana	1.828	24.242	229.0	3525.4
Breast-Cancer	0.584	1.294	62.3	137.2
Diabetis	17.975	81.209	936.0	4376.7
Flare-Solar	1.664	1.670	55.1	56.9
German	11.792	12.777	199.3	223.1
Heart	0.158	0.125	14.4	12.0
Image	48.173	142.859	2219.7	6603.0
Ringnorm	3.448	0.714	103.8	23.7
Splice	15.770	6.357	487.9	196.8
Thyroid	0.073	0.064	16.0	12.7
Titanic	0.086	0.070	18.1	17.6
Twonorm	1.830	1.056	55.5	31.8
Waveform	2.486	2.149	64.9	62.4

training samples; C the constant in Problem 1; σ the kernel width; n_t the number of test samples. The values of the hyper-parameters C and σ are set to those specified in the web site [6]. Also, q is set to 20, and τ is set to 0.01. Experimental results are summarized in Table 2 where each value represents the average for ten data sets. One can see that for data called Banana, Breast-Cancer, Diabetis and Image, both the CPU time and the number of iterations are significantly reduced by using CG method. In particular, Algorithm 2 is thirteen times faster than SVM$^{\text{light}}$ for Banana.

Next we have applied Algorithm 2 and SVM$^{\text{light}}$ to eight kinds of benchmark data sets on regression problems [7]. Table 3 shows the overview of the benchmark data where the the meanings of n, l, σ and n_t are same as Table 1, and C is the constant in Problem 2. The values of the hyper-parameters C and σ are determined based on preliminary experiments. Also, q is set to 20, and τ is set to 0.01. Experimental results are summarized in Table 4 where each value represents the average for ten data sets. For data called Abalone, Housing, Mg, Mpg, Pyrim and Trizaines, both the CPU time and the number of iterations are significantly reduced by using CG method. In particular, Algorithm 2 is about sixty times faster than SVM$^{\text{light}}$ for Abalone.

6. Conclusion

A novel working set selection based on the conjugate gradient method was proposed in this paper. Experimental results show that the proposed method is sometimes very effective. However, at the same time, the proposed method is as fast as or slower than the conventional method for some data sets. Making clear the conditions on training samples under which the proposed method effectively works is one of the future problems.

Table 3: Overview of the benchmark data sets [7]

Data	n	l	C	σ^2
Abalone	8	1000	31.62	1.00
Bodyfat	14	200	100.00	1.00
Housing	13	406	100.00	999.82
Mg	6	1000	1.00	1.00
Mpg	7	300	10.00	1000000.00
Pyrim	27	500	10.00	1.00
Space-ga	6	1000	1.00	1.00
Trizaines	60	136	1.00	1.00

Table 4: Comparison between Algorithm 2 (CG) and SVM$^{\text{light}}$ (SD) by CPU time and the number of iterations.

	CPU time[sec]		Iterations	
	CG	SD	CG	SD
Abalone	10.361	645.866	176.3	11147.9
Bodyfat	0.258	0.142	14.1	7.0
Housing	48.303	284.361	1548.6	9287.6
Mg	6.747	75.203	136.2	1539.2
Mpg	0.459	4.048	28.4	258.2
Pyrim	0.241	0.939	31.5	131.2
Space-ga	12.948	6.109	234.2	69.7
Trizaines	2.794	11.313	74.8	310.0

Acknowledgments

This work was supported in part by a research grant from the Okawa Foundation for Information and Telecommunications.

References

[1] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

[2] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods: Support Vector Machines*, B. Schölkopf, C. Burges and A. Smola, Eds. Cambridge, MA: MIT Press, 1998.

[3] T. Joachims, "Making large-scale support vector machine learning practical," in *Advances in Kernel Methods: Support Vector Machines*, B. Schölkopf, C. Burges and A. Smola, Eds. Cambridge, MA: MIT Press, 1998.

[4] S. S. Keerthi and E. G. Gilbert, "Convergence of a generalized SMO algorithm for SVM classifier design," *Machine Learning*, vol.46, pp.351–360, 2002.

[5] N. Takahashi and T. Nishi, "Global Convergence of Decomposition Learning Methods for Support Vector Machines," *IEEE Trans. Neural Networks*, vol.17, no.6, pp.1362–1369, 2006.

[6] `http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm`

[7] `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/regression.html`