DIMACS Technical Report 2007-08
July 2007

SADDLE POINT FEATURE SELECTION IN SVM REGRESSION

by

Yuri Goncharov
Artezio LLC
Moscow, Russia
ygoncharov@artezio.ru

Ilya Muchnik
DIMACS
Rutgers University
New Brunswick, New Jersey 08903
muchnik@dimacs.rutgers.edu

Leonid Shvartser
Ness TSG,
Atidim 10
Tel-Aviv
leonid.shvartser@ness.com

DIMACS is a collaborative project of Rutgers University, Princeton University, AT&T Labs–Research, Bell Labs, NEC Laboratories America and Telcordia Technologies, as well as affiliate members Avaya Labs, HP Labs, IBM Research, Microsoft Research, Stevens Institute of Technology, Georgia Institute of Technology and Rensselaer Polytechnic Institute. DIMACS was founded as an NSF Science and Technology Center.

ABSTRACT

SVM wrapper method, proposed in our previous work "SIMULTANEOUS FEATURE SELECTION AND MARGIN MAXIMIZATION USING SADDLE POINT APPROACH" is investigated and examined for the SVM regression. The method simultaneously maximizes margin and minimizes feature space with help of a modification of the standard criterion by adding to the basic objective function a third term, which directly penalizes a chosen set of variables. Our examination of the proposed min-max saddle point algorithm for the SVM regression case on a SAR Benchmark proves the ability of the introduced approach both to select small subspaces of features and to improve the regression prediction quality.

1 Introduction

This work is an addition to our previous work [1]. In [1] a new type of SVMs - saddle point SVM (SP-SVM) was introduced and investigated in rather full way for the SVM learning classification case. It was shown that the problem of simultaneous feature selection and optimization of prediction quality could be solved in the same way for the SVM regression case. However, a proposed convex-concave saddle point algorithm was not fully defined and implemented for regression and so this option had not been examined. Therefore we complete this analysis in the present work. The paper is organized into 4 sections. In the second section we repeat the basics of exact wrapped methods of feature selection for regression problem in order to make this paper autonomously for reading. In the third section we describe our saddle-point algorithm for the regression case. In the section 4 we describe and analyze this algorithm using a regression Benchmark for inhibition of dihydrofolate reductase by triazines Qualitative Structure Activity Relationships (QSARs) [2], [3]. Obtained results prove the concept of using min-max convex-concave techniques for feature selection in regression problems.

2 Exact Wrapped Method for Feature Selection in Regression

This section repeats section 5 from [1].

Following [4] suppose that we have a training sample $\{x_i, y_i\}, x_i \in R^M, y_i \in R, i = 1, \ldots, N$. Margin maximization (VC-dimension minimization) now reflects an attempt to approximate as more as possible experimental points with y-axe ϵ-tube:

$$\begin{aligned} \frac{1}{2}\|w\|^2 &\to \min_{w,b} \\ y_i - (w^T x_i - b) &\le \epsilon \\ y_i - (w^T x_i - b) &\ge -\epsilon \\ i &= 1, \ldots, N \end{aligned} \tag{1}$$

The goal is to approximate all the points with ϵ precision can be infeasible task, and the regression problem transformed to a following one:

$$\begin{aligned} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}(\delta_i + \delta_i^*) &\to \min_{w,b,\delta_i,\delta_i^*} \\ y_i - (w^T x_i - b) &\le \epsilon + \delta_i \\ y_i - (w^T x_i - b) &\ge -\epsilon - \delta_i^* \\ \delta_i, \delta_i^* &\ge 0, i = 1, \ldots, N \end{aligned} \tag{2}$$

It means that the points into the ϵ-tube do not penalized and the points outside the ϵ-tube penalized linearly. This problem has the following dual:

$$\sum_{j=1}^{N} y_j \left(\lambda_j - \lambda_j^*\right) - \frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{N} x_j^T x_k \left(\lambda_j - \lambda_j^*\right)\left(\lambda_k - \lambda_k^*\right) - \varepsilon \sum_{j=1}^{N}\left(\lambda_j + \lambda_j^*\right) \to \max_{\lambda,\lambda^*}$$
$$\sum_{j=1}^{N}\left(\lambda_j - \lambda_j^*\right) = 0, \tag{3}$$
$$0 \le \lambda_j, \lambda_j^* \le C, j = 1, \ldots, N$$

In [1] we introduced a feature selection problem as follows:

$$F(Q) \to \min_{Q \in 2^M}$$
$$F(Q) = \min_{w_Q, b_Q, \delta_Q^i, {\delta_Q^i}^*} \frac{1}{2}\left\|w_Q\right\|^2 + C\sum_{i=1}^{N}\left(\delta_Q^i + {\delta_Q^i}^*\right) + A|Q|$$
$$y_i - (w_Q^T x_i^Q - b_Q) \le \epsilon + \delta_Q^i \tag{4}$$
$$y_i - (w_Q^T x_i^Q - b_Q) \ge -\epsilon - {\delta_Q^i}^*$$
$$\delta_Q^i, {\delta_Q^i}^* \ge 0, i = 1, \ldots, N$$

and nest this problem into a continuous one

$$\min_z \max_{\lambda,\lambda^*} L(z,\lambda,\lambda^*) = \sum_{j=1}^{N} y_j\left(\lambda_j - \lambda_j^*\right) - \frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{N}\left(\left(\lambda_j - \lambda_j^*\right)\left(\lambda_k - \lambda_k^*\right)\sum_{l=1}^{M} z_l x_j^l x_k^l\right) -$$
$$-\varepsilon\sum_{j=1}^{N}\left(\lambda_j + \lambda_j^*\right) + A\sum_{l=1}^{M} z_l \tag{5}$$
$$\sum_{j=1}^{N}\left(\lambda_j - \lambda_j^*\right) = 0, \qquad 0 \le \lambda_j, \lambda_j^* \le C, j = 1, \ldots, N, \qquad 0 \le z_j \le 1, j = 1, \ldots, M$$

Lets take function $L(z,\lambda,\lambda^*)$ from (5) and consider the saddle point problem - find $(\hat{z}, \hat{\lambda}, \hat{\lambda}^*)$ that satisfy the following inequalities:

$$L(\hat{z},\lambda,\lambda^*) \le L(\hat{z},\hat{\lambda},\hat{\lambda}^*) \le L(z,\hat{\lambda},\hat{\lambda}^*), \forall z \in Z, \forall(\lambda,\lambda^*) \in \Lambda. \tag{6}$$

where

$$Z = \{z | 0 \le z_j \le 1, j = 1, \ldots, M\} \tag{7}$$

$$\Lambda = \{(\lambda, \lambda^*) | \sum_{j=1}^{N} \left(\lambda_j - \lambda_j^*\right) = 0, \qquad 0 \leq \lambda_j, \lambda_j^* \leq C, j = 1, \ldots, N\} \tag{8}$$

This problem has the following properties:

1) There exists a saddle point $(\hat{z}, \hat{\lambda}, \hat{\lambda}^*)$ of problem (6)- (8) and this saddle point is a solution of problem (5):

$$(\hat{z}, \hat{\lambda}, \hat{\lambda}^*) = \arg \min_{z} \max_{\lambda, \lambda^*} L(z, \lambda, \lambda^*);$$
$$(\hat{z}, \hat{\lambda}, \hat{\lambda}^*) = \arg \max_{\lambda, \lambda^*} \min_{z} L(z, \lambda, \lambda^*); \tag{9}$$

$$\sum_{j=1}^{N} \left(\lambda_j - \lambda_j^*\right) = 0, \qquad 0 \leq \lambda_j, \lambda_j^* \leq C, j = 1, \ldots, N, \qquad 0 \leq z_j \leq 1, j = 1, \ldots, M$$

2) If

$$\sum_{j=1}^{N} \sum_{k=1}^{N} \left(\left(\hat{\lambda_j} - \hat{\lambda_j^*}\right) \left(\hat{\lambda_k} - \hat{\lambda_k^*}\right) x_j^l x_k^l \right) > 2A$$

then $\hat{z}_l = 1$ and if

$$\sum_{j=1}^{N} \sum_{k=1}^{N} \left(\left(\hat{\lambda_j} - \hat{\lambda_j^*}\right) \left(\hat{\lambda_k} - \hat{\lambda_k^*}\right) x_j^l x_k^l \right) < 2A$$

then $\hat{z}_l = 0$.

3) A component $\hat{z}_l$ of a saddle point $(\hat{z}, \hat{\lambda}, \hat{\lambda}^*)$ can be different from 0 or 1 only if

$$\sum_{j=1}^{N} \sum_{k=1}^{N} \left(\left(\hat{\lambda_j} - \hat{\lambda_j^*}\right) \left(\hat{\lambda_k} - \hat{\lambda_k^*}\right) x_j^l x_k^l \right) = 2A$$

3 SVM Regression Saddle Point Algorithm

The flow of algorithm is the same as one described in [1], but with the calculation formulas applicable to the regression case. Below we use vector $\mu = (\lambda, \lambda^*)$ and use (z, μ) notation instead of (z, λ, λ^*) , $L(z, \mu)$ notation for function $L(z, \lambda, \lambda^*)$ and $(\hat{z}, \hat{\mu})$ notation for saddle point $(\hat{z}, \hat{\lambda}, \hat{\lambda}^*)$. The algorithm realizes separate calculation of z-component and μ-component of saddle point $(\hat{z}, \hat{\mu})$. Lets consider functions $\psi(z)$ and $\varphi(\mu) = \varphi(\lambda, \lambda^*)$ defined below:

$$\psi(z) = \max_{\lambda,\lambda^*} L(z,\lambda,\lambda^*) = \sum_{j=1}^{N} y_j \left(\lambda_j - \lambda_j^*\right) - \frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{N}\left(\left(\lambda_j - \lambda_j^*\right)\left(\lambda_k - \lambda_k^*\right)\sum_{l=1}^{M} z_l x_j^l x_k^l\right) - \tag{10}$$

$$-\varepsilon \sum_{j=1}^{N}\left(\lambda_j + \lambda_j^*\right) + A\sum_{l=1}^{M} z_l$$

$$\sum_{j=1}^{N}\left(\lambda_j - \lambda_j^*\right) = 0, 0 \le \lambda_j, \lambda_j^* \le C,\ j = 1,\dots,N$$

$$\varphi(\mu) = \min_{z} L(z,\lambda,\lambda^*) = \sum_{j=1}^{N} y_j \left(\lambda_j - \lambda_j^*\right) - \frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{N}\left(\left(\lambda_j - \lambda_j^*\right)\left(\lambda_j - \lambda_j^*\right)\sum_{l=1}^{M} z_l x_j^l x_k^l\right) \tag{11}$$

$$-\varepsilon \sum_{j=1}^{N}\left(\lambda_j + \lambda_j^*\right) + A\sum_{l=1}^{M} z_l$$

$$0 \le z_l \le 1,\ l = 1,\dots,M$$

According to [8]:
Let I -arbitrary set,
If $f_i(x)$-convex function for $\forall i \in I$ then $f(x) = \sup\{f_i(x)|i \in I\}$ is convex function.
If $f_i(x)$-concave function for $\forall i \in I$ then $f(x) = \inf\{f_i(x)|i \in I\}$ is concave function.

In our case $L(z,\lambda,\lambda^*)$ is linear and hence convex by z for fixed λ,λ^*.
So taking $I = \{\lambda,\lambda^*|\sum_{j=1}^{N}(\lambda_j - \lambda_j^*) = 0, 0 \le \lambda_j, \lambda_j^* \le C,\ j = 1,\dots,N\}$ we see that $\psi(z)$ is convex function.

On the other hand $L(z,\lambda,\lambda^*)$ is concave by λ,λ^* for fixed z.
Taking $I = \{z|0 \le z_l \le 1,\ l = 1,\dots,M\}$ we see that $\varphi(\mu) = \varphi(\lambda,\lambda^*)$ is concave function.
Any solution of the problem:

$$\min_{z}\{\psi(z)|0 \le z_l \le 1, l = 1,\dots,M\} \tag{12}$$

gives z-component of saddle point $(\hat{z},\hat{\mu})$.

Any solution of the problem:

$$\max_{\lambda,\lambda^*}\{\varphi(\lambda,\lambda^*)|\sum_{j=1}^{N}(\lambda_j - \lambda_j^*) = 0, 0 \le \lambda_j, \lambda_j^* \le C, j = 1,\dots,N\} \tag{13}$$

gives μ-component of saddle point $(\hat{z}, \hat{\mu})$.

We realized algorithm of non-differentiable convex (concave) optimization for solving problems (12) and (13).

The algorithm use the following scheme for generating sequences of points converging to solution of (12) and (13) [1]:

$$z^{t+\frac{1}{2}} = z^t - \alpha_t^z \partial\psi(z^t),\ \ z^{t+1} = \pi_Z(z^{t+\frac{1}{2}}), t = 0, 1 \ldots\ldots \tag{14}$$

$$\mu^{t+\frac{1}{2}} = \mu^t + \alpha_t^\mu \partial\varphi(\mu^t),\ \ \mu^{t+1} = \pi_\Lambda(\mu^{t+\frac{1}{2}}), t = 0, 1 \ldots\ldots \tag{15}$$

where t is a iteration number, $\partial\psi(z^t)$ is subgradient of function $\psi(z)$ in point z^t, $\pi_Z(z)$ is projection of point $z \in R^M$ on set $Z = \{0 \le z_l \le 1, l = 1, \ldots, M\}$,

$\partial\varphi(\mu^t)$ is subgradient of function $-\varphi(\mu)$ (supergradient) in point $\mu^t, \pi_\Lambda(\mu)$ is projection of point $\mu = (\lambda, \lambda^*) \in R^{2N}$ on set

$$\Lambda = \{\sum_{j=1}^{N}(\lambda_j - \lambda_j^*) = 0, 0 \le \lambda_j, \lambda_j^* \le C, j = 1, \ldots, N\},\ \ \alpha_t^z \text{ and } \alpha_t^\mu \text{ are step-sizes.}$$

We realized several well-known methods of setting values for the step-sizes. The best result gives a method that uses bounds on the optimal value of optimized function [7]. Lets describe computation of step-size α_t^z.

If we know value $\psi(\hat{z})$, where $\hat{z}$ is a solution of (12), then we can set step value to

$$\alpha_t^z = \frac{\psi(z^t) - \psi(\hat{z})}{\|\partial\psi(z^t)\|^2}$$

Such choice of step-size gives convergence of sequence (14) to $\hat{z}$, see, for example theorem 7.2 from [7].

The value of $\psi(\hat{z})$ is unknown, we just looking for it. As it is shown in paragraphs 5.3 and 7.2 of [7] a low estimate of $\psi(\hat{z})$ can be used for practical computations.

Obviously $\psi(z) \ge \varphi(\mu)$ for all $z \in Z, \mu \in \Lambda$. Assign φ^{rec}as maximal record value of $\varphi(\mu^t)$ achieved during optimization process to iteration t. φ^{rec} is a low estimate of $\psi(\hat{z})$, $\psi(\hat{z}) \ge \varphi^{rec}$, and we have step-size formula:

$$\alpha_t^z = \frac{\psi(z^t) - \varphi^{rec}}{\|\partial\psi(z^t)\|^2} \tag{16}$$

In the same way we get formula for value of step-size by μ :

[1] Each step of iterative procedure consists of two substeps: step by subgradient and projection to the set of constraints. This is the reason for usage of indexes t, $t + \frac{1}{2}$ and $t + 1$ in the definition of the procedure.

$$\alpha_t^{\mu} = \frac{\psi^{rec} - \varphi(\mu^t)}{\left\|\partial\varphi(\mu^t)\right\|^2} \tag{17}$$

To calculate $\partial\psi(z)$ - subgradient of $\psi(z)$ in point z we should get $\bar{\mu} = (\bar{\lambda}, \bar{\lambda}^*)$ - solution of the problem:

$$\begin{aligned}\arg\max_{\lambda,\lambda^*} L(z,\lambda,\lambda^*) = \sum_{j=1}^{N} y_j \left(\lambda_j - \lambda_j^*\right) - \frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{N}\left(\left(\lambda_j - \lambda_j^*\right)\left(\lambda_k - \lambda_k^*\right)\sum_{l=1}^{M} z_l x_j^l x_k^l\right) \\ -\varepsilon\sum_{j=1}^{N}\left(\lambda_j + \lambda_j^*\right) + A\sum_{l=1}^{M} z_l \quad (18) \\ \sum_{j=1}^{N}(\lambda_j - \lambda_j^*) = 0, 0 \leq \lambda_j, \lambda_j^* \leq C,\ j = 1,\ldots,N\end{aligned}$$

and calculate derivative of function $L(z,\bar{\mu})$ by z. The formula for $l-$component of subgradient $\partial\psi(z)$ will be: [2]:

$$\frac{\partial\psi(z)}{\partial z_l} = -\frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{N}\left(\bar{\lambda}_j - \bar{\lambda}_j^*\right)\left(\bar{\lambda}_k - \bar{\lambda}_k^*\right) x_j^l x_k^l + A \tag{19}$$

To calculate $\partial\varphi(\mu)$ - supergradient of $\varphi(\mu)$ in point μ we should get $\bar{z}$ - solution of the **linear programming** problem:

$$\begin{aligned}\arg\min_{z} L(z,\lambda,\lambda^*) =& \sum_{j=1}^{N} y_j \left(\lambda_j - \lambda_j^*\right) - \frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{N}\left(\left(\lambda_j - \lambda_j^*\right)\left(\lambda_k - \lambda_k^*\right)\sum_{l=1}^{M} z_l x_j^l x_k^l\right) \\ &- \varepsilon\sum_{j=1}^{N}\left(\lambda_j + \lambda_j^*\right) + A\sum_{l=1}^{M} z_l \qquad (20) \\ 0 \leq & z_l \leq 1,\ l = 1,\ldots,M\end{aligned}$$

and calculate derivative of function $-L(\bar{z},\mu)$ by μ. The formula for $j-$component of supergradient $\partial\varphi(\mu)$ will be:

$$\begin{aligned}\frac{\partial\varphi(\mu)}{\partial\lambda_j} &= -y_j + \sum_{k=1}^{N}\left(\left(\lambda_k - \lambda_k^*\right)\sum_{l=1}^{M}\bar{z}_l x_j^l x_k^l\right) + \varepsilon \qquad (21) \\ \frac{\partial\varphi(\mu)}{\partial\lambda_j^*} &= y_j - \sum_{k=1}^{N}\left(\left(\lambda_k - \lambda_k^*\right)\sum_{l=1}^{M}\bar{z}_l x_j^l x_k^l\right) + \varepsilon\end{aligned}$$

[2] Functions $\psi(z)$ and $\varphi(\mu)$ are non-differentiable. That's why $\frac{\partial\psi(z)}{\partial z_l}$, $\frac{\partial\varphi(\mu)}{\partial\lambda_j}$ and $\frac{\partial\varphi(\mu)}{\partial\lambda_j^*}$ are only the notations for the corresponding components of sub- and super- gradient vectors.

To find $\bar{z}$ we calculate coefficients of linear by z function $L(z, \mu)$:

$$p_l = -\frac{1}{2}\sum_{j=1}^{N}\sum_{k=1}^{N}\left(\lambda_j - \lambda_j^*\right)\left(\lambda_k - \lambda_k^*\right) x_j^l x_k^l + A \tag{22}$$

and calculate

$$\bar{z}_l = \begin{cases} 1, & p_l < 0 \\ 0, & p_l \geq 0 \end{cases} \tag{23}$$

Projection $z^{pr} = \pi_Z(\tilde{z}),\ Z = \{0 \leq z_l \leq 1, l = 1, \ldots, M\}, \tilde{z} \in R^M$ calculated by simple formula:

$$z_l^{pr} = \begin{cases} 0, & \tilde{z}_l < 0 \\ \tilde{z}_l, & 0 \leq \tilde{z}_l \leq 1\ , l = 1, \ldots, M \\ 1, & \tilde{z}_l > 1 \end{cases} \tag{24}$$

Projection $\lambda^{pr} = \pi_\Lambda(\tilde{\mu}),\ \Lambda = \{\sum_{j=1}^{N}(\lambda_j - \lambda_j^*) = 0, 0 \leq \lambda_j, \lambda_j^* \leq C, j = 1, \ldots, N\}$, $\tilde{\mu} = (\tilde{\lambda}, \tilde{\lambda^*}) \in R^{2N}$ is given by solution of **quadratic programming** problem:

$$\min_{\lambda,\lambda^*}\left\{\sum_{j=1}^{N}(\lambda_j - \tilde{\lambda}_j)^2 + \sum_{j=1}^{N}(\lambda_j^* - \tilde{\lambda}_j^*)^2 | \sum_{j=1}^{N}(\lambda_j - \lambda_j^*) = 0, 0 \leq \lambda_j, \lambda_j^* \leq C, j = 1, \ldots, N\right\} \tag{25}$$

Now we give a pseudocode description of the algorithm:

Initialization.

Choose $z^0 \in Z$, $\mu^0 \in \Lambda$,

$$Z = \{z | 0 \leq z_j \leq 1, j = 1, \ldots, M\}$$

$$\Lambda = \{(\lambda, \lambda^*) | \sum_{j=1}^{N}\left(\lambda_j - \lambda_j^*\right) = 0, \qquad 0 \leq \lambda_j, \lambda_j^* \leq C, j = 1, \ldots, N\}$$

Set solution accuracy $\sigma > 0$

Set $t = 0$.

Init record values $\psi^{rec} = \psi(z^0)$, $\varphi^{rec} = \varphi(\mu^0)$, $z^{rec} = z^0, \mu^{rec} = \mu^0$, where ψ, φ calculated using formulas (10) and (11).

Stopping test.

if $\frac{\psi^{rec}-\varphi^{rec}}{(1+\psi^{rec})} \leq \sigma$
then $\hat{z} = z^{rec}, \hat{\mu} = \mu^{rec}$, Stop.
Step by z.
Calculate $\partial\psi(z^t)$ using procedure (18),(19) and $\alpha_t^z = \frac{\psi(z^t)-\varphi^{rec}}{\|\partial\psi(z^t)\|^2}$ using (16).
Calculate $z^{t+\frac{1}{2}} = z^t - \alpha_t^z \partial\psi(z^t)$ (see (14))
Project $z^{t+1} = \pi_Z(z^{t+\frac{1}{2}})$ using (24)

Step by μ.
Calculate $\varphi(\mu^t)$ using (18)-(23) and $\alpha_t^\mu = \frac{\psi^{rec}-\varphi(\mu^t)}{\|\partial\varphi(\mu^t)\|^2}$ using (17).
Calculate $\mu^{t+\frac{1}{2}} = \mu^t + \alpha_t^\mu \partial\varphi(\mu^t)$ (see (15)).
Project $\mu^{t+1} = \pi_\Lambda(\mu^{t+\frac{1}{2}})$ by solving a quadratic programming problem (25)
Update records
If $\psi(z^{t+1}) < \psi^{rec}$ then $\psi^{rec} = \psi(z^{t+1}), z^{rec} = z^{t+1}$
If $\varphi(\mu^{t+1}) > \varphi^{rec}$ then $\varphi^{rec} = \varphi(\mu^{t+1}), \mu^{rec} = \mu^{t+1}$
$t = t + 1$
go to Stopping test

4 Experimental Results

4.1 Experimental Framework

As we already mentioned in our previous work, the proposed algorithm is still time consuming. It is working rather fast on data arrays with ∼30 observations (calculation time < 1 sec), but it takes to it more than hour to process real data arrays with more than 1000 observations. This is an experimental framework that was realized. Let M is overall number of objects in a Benchmark set.

1. Any given matrix was used for generation of submatrices of objects of length L, which is taken randomly with uniform distribution from interval 15-30. Thus each matrix contained L objects and all N variables.

2. Parameter C was chosen according to [5]: taken SVM light, chosen default $C = (average\|x_i\|)^{-1}$, extracted support vectors and calculated default C only with support vectors (x_i is a vector of i-th object in the feature space of features, average is taken over all the objects).

3. Parameter A chosen randomly with uniform distribution from a predefined interval. The boundaries for A were chosen experimentally in such a way that the solutions would be non-trivial (small A remain all the variables, large A - delete a lot, may be all the variables). The boundaries were chosen as a function of the hyperplane coefficients w for a case $A = 0$.

4. The regression tolerance ϵ (see (5)) was taken as a default value 0.05.

5. Saddle point training algorithm found $(\hat{z}, \hat{\mu})$ - approximation to saddle point.

6. Three resulted sets of variables are stored:

   a. "Deleted" - those, which obtained weight 0 (assigned as not important) and were deleted by the Saddle Point algorithm;

   b. "Assigned" - those, which obtained weight 1 (assigned as highly important) by the Saddle Point algorithm;

   c. "Weighted" - those, which obtained weight $0 \leq z_i \leq 1$ (see section 2).

7. All the complementary objects $(M \backslash L)$ were used as examined objects.

8. Prediction quality evaluator for SVM regression is $\epsilon Losses$ [4]:

$$\epsilon Losses = \frac{1}{|T|} \sum_{i \in T, |y_i - (w^T x_i - b| \geq \epsilon} |y_i - (w^T x_i - b|,$$

   where T is a testing set of examples.

9. The evaluator was calculated for the SP-SVM regression.

4.2 Benchmark

We made experiments with a Benchmark described in [2],[3]: dihydrofolate reductase by triazines Qualitative Structure Activity Relationships (QSARs). The data is organized for 6 fold cross-validation series of 186 compounds. We took all the series together and obtained a dataset contains 1116 points, corresponding to the different activities of the compound. Each drug has 6 positions of possible substitution. There are 10 attributes for each substitution position: polar, size, flex, h_donor, h_acceptor, pi_donor, pi_acceptor, polarisable, sigma and branch. It means that each point contains its activity level and 60 attributes. All the attribute values are real numbers. They are rescaled between 0.1 and 0.9. Dataset is available by FTP from ftp://ftp.ics.uci.edu/pub/machine-learning-databases/qsar/.

4.3 Results

There were generated 10 submatrices. First of all the regular linear SVM regression was evaluated using each one of these submatrices for training and correspondingly their additions for the full matrix for recognition. The first important observation was that the SVM regression uses almost all the features as significant ones.
Then 31 random uniform selections of parameter A from interval [3]

[3] w^0 is a normal of hyperplane for $A = 0$

$P = [\min_{i=1,\ldots,M} w_i^0, \max_{i=1,\ldots,M} w_i^0]$ were done for each sub matrix.
The results presented in Tables 1, 2. In Table 1 presented results of comparison of Minimal $\epsilon Losses$ of saddle point in the examined interval P with SVM results for a same data submatrix. Minimum of $\epsilon Losses$ is taken over all the 31 choices of parameter A from the interval P. These results show possibilities to decrease the feature space and to increase prediction quality simultaneously. It was the first set of experiments.
As we already mentioned in [1] these results prove only the existence of an optimal value of parameter A, which gives Minimal $\epsilon Losses$ on the prediction stage. The choice of this optimal value is a plan of our future investigations. This is the reason that we tried to choose this parameter in a very simple way: to choose such A that gives an average value of the optimized saddle point criterion. Once more, average is taken over all the 31 choices of parameter A from the interval P. The results of this second set of experiments presented in Table 2.
One can see from the Table 1 that the Minimal $\epsilon Losses$ of saddle point is a little better than SVM $\epsilon Losses$. At the same time approximately 75-80% of the variables are eliminated (the average number of eliminated variables over all the 10 submatrices is 46.7 from 60). Average SVM $\epsilon Losses$ is 0.101; average saddle point $\epsilon Losses$ is 0.097. The Accuracy obtained with saddle point algorithm is better than the SVM $\epsilon Losses$ in 7 cases from 10. In case 8 saddle point and SVM $\epsilon Losses$ are equal. The only case of saddle point accuracy greater than SVM $\epsilon Losses$ is case number 18.
Obviously the results of the second set of experiments are worse but still good from the $\epsilon Losses$ point of view and very good from the point of view of feature selection. Average SVM $\epsilon Losses$ is 0.101; average saddle point $\epsilon Losses$ is 0.1019. For the choice of result corresponding to average of saddle point values on interval P only in 6 cases from 10 the saddle point $\epsilon Losses$ is better than the SVM $\epsilon Losses$. At the same time the number of deleted variables is rather big. In average 44.6 variables from 60 were eliminated.
This analysis of Table 2 shows that even in case of such simple choice of parameter A the algorithm does feature selection without big decreasing in $\epsilon Losses$. Analysis of Table 1 shows that to find a method of searching for an optimal A is a very important task.

Table 1: Minimal Losses.

Number of Submatrix	Number of Objects in Submatrix	$\epsilon Losses$ SVM	Min $\epsilon Losses$ Saddle Point	Number of deleted variables in Min $\epsilon Losses$ SP $(z_i = 0)$	Number of definitely assigned variables in Min $\epsilon Losses$ SP $(z_i = 1)$
1	76	0.103	0.1	49	1
2	88	0.113	0.101	47	1
3	88	0.095	0.092	45	1
4	90	0.105	0.097	53	1
5	74	0.105	0.099	45	1
6	82	0.088	0.089	45	6
7	84	0.105	0.098	51	0
8	66	0.094	0.096	53	1
9	70	0.11	0.108	43	1
10	64	0.093	0.096	36	9

Table 2: Average value of optimized criterion in experimental interval.

Number of Submatrix	Number of Objects in Submatrix	$\epsilon Losses$ SVM	Average criteria $\epsilon Losses$ Saddle Point	Number of deleted variables in Average criteria SP ($z_i = 0$)	Number of definitely assigned variables in Average criteria SP ($z_i = 1$)
1	76	0.103	0.101	46	1
2	88	0.113	0.106	36	4
3	88	0.095	0.094	45	3
4	90	0.105	0.107	43	4
5	74	0.105	0.102	40	2
6	82	0.088	0.099	52	2
7	84	0.105	0.103	47	1
8	66	0.094	0.099	51	1
9	70	0.11	0.109	43	1
10	64	0.093	0.099	43	3

5 Acknowledgements

Ilya Muchnik thanks to NSF for partial support of this work with grant # 0325398. Thanks to Thorsten Joachims for letting us use his program and to make small changes in it for our needs.

References

[1] Goncharov Yu., Muchnik I., Shvartser L. ,*Simultaneous Feature Selection and Margin Maximization Using Saddle Point Approach*, DIMACS Technical Report 2004-08, April 2004

[2] King, Ross .D., Hurst, Jonathan. D., and Sternberg, Michael.J.E.,*A comparison of artificial intelligence methods for modelling QSARs*, Applied Artificial Intelligence, 1994

[3] Hurst, Jonathan. D., King, Ross .D. and Sternberg, Michael.J.E.,*Quantitative Structure-Activity Relationships by neural networks and inductive logic programming: 2. The inhibition of dihydrofolate reductase by triazines.*, Journal of Computer Aided Molecular Design, 1994.

[4] V. N. Vapnik, *The Nature of Statistical Learning Theory*, 1995,New York

[5] Joachims, T.,*Estimating the Generalization Performance of an SVM Efficiently*, Proceedings of ICML-00, 17th International Conference on Machine Learning, 1999.

[6] Andrei V. Anghelescu and Ilya B. Muchnik *Optimization of SVN in a Space of Two Parameters: Weak Margin and Intercept*, DIMACS Working Group on Monitoring Message Streams, May 2003.

[7] B.T.Polyak *Introduction to optimization*, Moscow (in Russian), 1983.

[8] R.T. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, 1970