Learning from Data – A SVM Based Approach

Cătălina Lucia COCIANU[1], Luminiţa STATE[2], Cristian Răzvan USCATU[1], Raluca ŞTEFAN[1]
[1]Dept. of Economy Informatics, Academy of Economic Studies, Bucureşti, Romania
[2]Dept. of Computer Science, University of Piteşti, Piteşti, Romania
ccocianu@ase.ro, lstate@clicknet.ro, cristiu@ase.ro

Support Vector Machines were introduced by Vapnik within the area of statistical learning theory and structural risk minimization and first applied to classification problems as alternatives to multi-layer neural networks. The high generalization ability provided by Support Vector Classifiers has inspired recent work on computational speedups as well as the fundamental theory of model complexity and generalization. In this paper we present a gradient-type learning algorithm for training a SVM and analyze its performance when applied to linear separable data. In order to evaluate the efficiency of our learning method, several tests were performed, the conclusions being formulated in the final section of the paper.
Keywords: *Support Vector Machines, Supervised Learning, Regression, Classification, Gradient Method, The SMO Algorithm*

1 Introduction

In recent years, neural networks as multilayer perceptrons and radial basis function networks have been frequently used in a wide range of fields, including control theory, signal processing, model selection and parameter tuning in text categorization, bioinformatics, and nonlinear modeling. Support Vector Machines (SVM) were introduced by Vapnik within the area of statistical learning theory and structural risk minimization and first applied to classification problems as alternatives to multi-layer neural networks [16], [17]. The high generalization ability provided by Support Vector Classifiers (SVCs) has inspired recent work on computational speedups as well as the fundamental theory of model complexity and generalization.
According to the theory of SVMs, while traditional techniques for pattern recognition are based on the attempt to optimize the performance in terms of the empirical risk, SVMs minimize the structural risk, that is, the probability of misclassifying yet-to-be-seen patterns for a fixed but unknown probability distribution of data. The most distinguished and attractive features of this classification paradigm are the ability to condense the information contained by the training set and the use of families of decision surfaces of relatively low Vapnik-Chervonenkis dimension.
SVM approaches to classification lead to convex optimization problems, typically quadratic problems in a number of variables equal to the number of examples, these optimization problems becoming challenging when the number of data points exceeds few thousands.
For making SVM more practical, several algorithms have been developed such as Vapnik's chunking, and Osuna's decompositions [9] [19]. They make the training of SVM possible by breaking the large QP-problem into a series of smaller QP-problems and optimizing only a sub-set of training data patterns at each step. Because the subset of training data patterns optimized at each step is called the working set, these approaches are referred as the working set methods.
Recently, a series of works on developing parallel implementation of training SVM's have been proposed [4]. Also, there have been proposed methods to solve the least squares SVM formulations [3] [5] [15] as well as software packages as SVM^{light} [6], mysvm [11] and many others [1] [2] [8].
In this paper we present a heuristic learning algorithm of gradient type for training a SVM and analyze its performance in terms of accuracy and efficiency when applied to linear separable data. In order to evaluate the efficiency of our learning method, several

tests were performed, the conclusions being formulated in the final section of the paper.

2 The Largest Margin Linear Classifier Based on Support Vectors (SVM) for Linear Separable Data

SVM learning is one of the best supervised learning algorithms. The task is to predict on the basis of a finite set of labeled examples $\mathcal{S} = \{(x_i, y_i), x_i \in \mathbf{R}^n, y_i \in \{-1,1\}, 1 \le i \le N\}$ either a test sample comes from one of two classes h_1, h_2, or it is unclassifiable. The first component of each pair $(x_i, y_i) \in \mathcal{S}$ is a particular example and the second one, conventionally denoted either by -1 or by 1, is the label of the provenance class of x_i. The examples coming from h_1 are labeled by 1 and they are referred as positive examples, while the examples coming from h_2 are labeled by -1 and they are referred as negative examples.

The classification (recognition) is performed in terms of a parameterized decision rule $h_{b,w} : \mathbf{R}^n \to \{-1,1\}$,

$$h_{b,w}(x) =, \quad \begin{cases} 1, & w^T x + b \ge 0 \\ -1, & w^T x + b < 0 \end{cases}, \quad \text{where}$$

$w \in \mathbf{R}^n, b \in \mathbf{R}$.

The set $\mathcal{S}$ is called linearly separable if

$$\mathsf{H}_S = \{(w,b) \mid w \in \mathbf{R}^n, b \in \mathbf{R}, (w^T x_i + b) y_i > 0, 1 \le i \le N\} \ne \varnothing.$$

Obviously, for finite size data sets, if $\mathcal{S}$ is linearly separable, then $\mathcal{H}_{\mathcal{S}}$ is an infinite set.

In order to assure the best generalization capability, that is to classify well new examples coming from these two classes, from intuitive point of view, one has to determine the optimal margin classifier $(w^*, b^*) \in \mathcal{H}_{\mathcal{S}}$ that separates the positive and the negative training examples with the largest "gap" between them.

Let $H(w,b)$ be a hyper plane that separates without errors $\mathcal{S}$. For any $(x_i, y_i) \in \mathcal{S}$, let γ_i be the distance of x_i to $H(w,b)$. The value γ_i is the geometric margin of $H(w,b)$ with respect to (x_i, y_i).

Obviously,

$$\gamma_i = y_i \left(\left(\frac{w}{\|w\|} \right)^T x_i + \frac{b}{\|w\|} \right)$$

that is the values of the geometric margins are not modified if the parameters w and b are multiply by a positive constant([14]).

The geometric margin of $H(w,b)$ with respect to $\mathcal{S}$ is $\gamma = \min_{1 \le i \le N} \gamma_i$.

From mathematical point of view, an optimal margin classifier is a solution of the quadratic programming (QP) problem,

$$(1) \begin{cases} \text{minimize } \frac{1}{2} \|w\|^2 \\ y_i (w^T x_i + b) \ge 1, \quad 1 \le i \le N \end{cases}$$

The problem (1) can be solved using the Lagrange multiplier method. Let $L(w, b, \alpha_1, \alpha_2, ..., \alpha_N)$ be the objective function,

$$(2) \, L(w, b, \alpha_1, \alpha_2, ..., \alpha_N) = \frac{1}{2} w^T w + \sum_{i=1}^{N} \alpha_i \left[1 - y_i (w^T x_i + b)\right]$$

where $\alpha_1, \alpha_2, ..., \alpha_N$ are nonnegative Lagrange multipliers.

The dual optimization problem yields to the QP problem on the objective function

$$\theta_d(\alpha) = \min_{w,b} L(w,b;\alpha),$$

$$(3)\begin{cases} \text{maximize}\,\theta_d(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j x_i^T x_j \\ \sum_{\mathrm{i=1}}^{\mathrm{N}}\alpha_{\mathrm{i}} y_i = 0 \\ \alpha_i \ge 0, 1 \le i \le N \end{cases}$$

If $\alpha^* = (\alpha_1^*, \alpha_2^*, ..., \alpha_N^*)^T$ is a solution of (3), then the optimal value of the parameter w is

$$w^* = \sum_{i=1}^{N}\alpha_i^* y_i x_i \,.$$

The parameter b cannot be explicitly computed by solving the SVM problem, a convenient choice of b being derived in terms of the support vectors and w^*.

The Karush-Khun-Tucker (KKT) complementarity conditions are,

$$\alpha_i^*\left(1 - y_i\left(w^{*T} x_i + b\right)\right) = 0, 1 \le i \le N$$

,

that is,

for any $1 \le i \le N$, if $\alpha_i^* > 0$, then $y_\mathrm{i}\left(w^{*T} x_i + b\right) = 1$holds.

The examples x_i for which $\alpha_i^* \ne 0$are called support vectors.

The value b should be taken such that $y_i\left(w^{*T} x_i + b\right) \ge 1$ holds for all examples. A suitable value of b^* should be selected such that,

$$1 - \min_{\substack{i \\ y_i = 1}} w^{*T} x_i \le b^* \le -1 - \max_{\substack{i \\ y_i = -1}} w^{*T} x_i \,,$$

for instance in case we take b^*as the middle of the interval, we get

$$(4)\; b^* = -\frac{1}{2}\left\{\max_{\substack{i \\ y_i = -1}} w^{*T} x_i + \min_{\substack{i \\ y_i = 1}} w^{*T} x_i\right\}.$$

The SVM classifier implements the IF-THEN decision rule $D : \mathbf{R}^n \to \{h_1, h_2, U\}$, where

$$\begin{aligned} &\text{IF } w^{*T} x + b^* > 0 \text{ THEN } D(x) = h_1 \\ &\text{IF } w^{*T} x + b^* < 0 \text{ THEN } D(x) = h_2 \\ &\text{IF } w^{*T} x + b^* = 0 \text{ THEN } D(x) = U \end{aligned}$$

and U means “unclassifiable”.

3 Brief Overview on a Certain Class of Methods for Solving the SVM-QP Problem

There have been proposed a series of methods to solve the corresponding dual QP optimization problems to SVM learning as for instance Sequential Minimal Optimization (SMO), decomposition methods [13] and [8], and methods to solve the least squares SVM formulations [3] [5] [15] as well as software packages as SVM$^{\text{light}}$ [7], mysvm [11] and many others.

In 1982 Vapnik [16] proposed a method to solve the QP problem arising from SVM referred as “chunking”. The large QP problem can be split into smaller QP problems whose ultimate goal is to identify all the non-zero Lagrange multipliers and discard all the zero ones. Chunking reduces significantly the size of the matrix corresponding to the particular QP problem, but still cannot handle any large scale training problem.

A new class of QP algorithms for SVM derived from the developments proposed by Osuna [9]. Osuna proved that the large QP problem can be reduced to a series of smaller QP sub-problems based on the idea that as long as at least one example that violets the KKT conditions is added to the examples used in the previous sub-problem, at each step the overall objective function is reduced and a feasible point that obeys the constraints is maintained.

Sequential minimal optimization (SMO) algorithm proposed by Platt ([10]) is a simple algorithm that allows to solve the SVM-QP problem without extra-matrix storage by decomposing the overall QP problem into simple QP sub-problems using Osuna's theorem [9]. Unlike the previous proposed methods, the SMO algorithm solves the smallest optimization problem at each step. Obviously, in case of the standard SVM-QP problem the smallest possible optimization problem involves two Lagrange multipliers, because the Lagrange multipliers must fulfill a linear equality constraint.

The idea of the SMO algorithm is to use a predefined constant $C > 0$, and the tolerance parameter $\tau > 0$, to express sort of tradeoff between accuracy and efficiency. At each iteration two examples (x_p, y_p) , (x_q, y_q) are looked for such that the following condition holds,

$$(5)\quad \begin{aligned}&\left(f_\alpha(x_p) - y_p + \tau < f_\alpha(x_q) - y_q - \tau\right) \wedge \left((\alpha_p < C \wedge y_p = 1) \vee (\alpha_p > 0 \wedge y_p = -1)\right) \wedge \\ &\wedge \left((\alpha_q < C \wedge y_q = -1) \vee (\alpha_q > 0 \wedge y_q = 1)\right)\end{aligned}$$

In case there exists at least a pair (x_p, y_p) , (x_q, y_q) for which (5) holds, the components α_p and α_q of the current parameter vector $\alpha = (\alpha_1, \alpha_2, ..., \alpha_N)$ are changed such that to increase $f_\alpha(x_p)$ and to decrease $f_\alpha(x_q)$, where

$$f_\alpha(x) = b + x^T\left(\sum_{i=1}^{N} \alpha_i y_i x_i\right).$$

In order to get a new point α fulfilling the constraint $\sum_{\mathrm{i}=1}^{\mathrm{N}} \alpha_{\mathrm{i}} y_i = 0$, the updating rules are,

$$\alpha_p \leftarrow \alpha_p + y_p \eta$$
$$\alpha_q \leftarrow \alpha_q - y_q \eta$$

where

$$(6)\ \eta = \frac{\left(f_\alpha(x_q) - y_q\right) - \left(f_\alpha(x_p) - y_p\right)}{\left\|x_p - x_q\right\|^2}$$

.

In case the conditions $0 \le \alpha_p + y_p \eta \le C$ and $0 \le \alpha_q - y_q \eta \le C$ do not hold, it is recommended that the updating should be performed using a smaller value of the coefficient η.

The search is over when there are no examples (x_p, y_p) , (x_q, y_q) satisfying (5), the optimal margin b being computed as,

$$b^* = -\frac{1}{2}\left\{\max_{\substack{i \\ y_i = -1}} w^{*T} x_i + \min_{\substack{i \\ y_i = 1}} w^{*T} x_i\right\}.$$

The SMO learning algorithm can be described as follows.

INPUT: $\alpha \leftarrow \mathbf{0}_N$, $b \leftarrow 0$, $C > 0$, $\tau > 0$

Continue $\leftarrow 1$

```
DO
    Find two examples $(x_p, y_p)$, $(x_q, y_q)$ such that (5) holds
    IF no such examples can be found $Continue \leftarrow 0$
    ELSE
        Compute $\eta$ using (6)
        $\alpha_p \leftarrow \alpha_p + y_p\eta$
        $\alpha_q \leftarrow \alpha_q - y_q\eta$
        IF needed, reduce $\eta$ such that
        $0 \le \alpha_p + y_p\eta \le C$ and
        $0 \le \alpha_q - y_q\eta \le C$
WHILE (Continue)
```

A long series generalizations and improvements have been recently proposed by many authors. For instance, in [4] a parallel version of SMO is proposed to accelerate the SVM training. Unlike the sequential SMO algorithm, that handles all the training data points using one CPU processor, the parallel version of the SMO algorithm first partitions the entire training data set into smaller subsets and next runs simultaneously more CPU processors to deal separately with each subsets.

4 A New Method for Solving the SVM-QP Problem

Our method for solving the SVM-QP problem is a heuristic variant of the gradient ascent method. For simplicity sake we assume that the first *m* examples come from the first class (that is their labels are 1) and the next $N-m$ examples come from the second class (all of them are labeled by -1).
The entries of the gradient $\nabla_\alpha Q(\alpha)$ of the objective function

$$(7)\ Q(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j x_i^T x_j$$

are ,

$$(8)\ \frac{\partial Q(\alpha)}{\partial \alpha_k} = 1 - x_k^T y_k \sum_{i=1}^{N}\alpha_i x_i y_i, \quad 1 \le i \le N$$

Consequently, the entries of the Hessian matrix $H(Q(\alpha)) = \left\| \frac{\partial^2 Q(\alpha)}{\partial\alpha_k \partial\alpha_p} \right\|_{k,p}$ are,

$$\frac{\partial^2 Q(\alpha)}{\partial\alpha_k\partial\alpha_p} = -\frac{\partial\left(\sum_{i=1}^{N}\alpha_i y_i x_i\right)^T x_k y_k}{\partial\alpha_p} = -y_p x_p^T x_k y_k$$

and

$$\forall\alpha \in \mathbf{R}^{\mathrm{N}} - \{\mathbf{0}\}\ \alpha^T H\alpha = -\sum_{k=1}^{N}\sum_{p=1}^{N}\alpha_k\alpha_p y_k y_p x_p^T x_k = -\left\|\sum_{k=1}^{N}\alpha_k y_k x_k\right\|^2 \le 0$$

that is $H(Q(\alpha))$ is negative semi-defined.

If we denote by α^{old} the current value of the parameter α, by applying the standard gradient ascent updating rule we get

$$(9)\ \alpha = \alpha^{old} + \rho\nabla_\alpha Q(\alpha)\Big|_{\alpha=\alpha^{old}},$$

where $\rho > 0$ is the learning rate.

However, this updating rule can not be applied straightforward because the constraint $\sum_{\mathrm{i=1}}^{\mathrm{N}}\alpha_{\mathrm{i}} y_i = 0$ have to be satisfied, and consequently, a modified learning rule of gradient ascent type should be considered instead of (9). In our approach, the updating step is performed as follows.
Assume that p_1, p_2 are such that

$$1 \le p_1 \le m, \quad m+1 \le p_2 \le N$$

and let $\alpha = (\alpha_1, \alpha_2, ..., \alpha_N)^T$ be such that, $\alpha_i = \alpha_i^{old}$, $1 \le i \le N, i \ne p_1, p_2$

$$\alpha_{p_i} = \alpha_{p_i}^{old} + \rho\left[\rho_1 \frac{\partial Q(\alpha)}{\partial \alpha_{p_1}}\Big|_{\alpha=\alpha^{old}} + (1-\rho_1)\frac{\partial Q(\alpha)}{\partial \alpha_{p_2}}\Big|_{\alpha=\alpha^{old}}\right], \quad i = 1,2, 0 \le \rho_1 \le 1$$

Obviously, if α^{old} satisfy the constraint $\sum_{i=1}^{N} \alpha_i^{old} y_i = 0$ then α also satisfies the constraint $\sum_{i=1}^{N} \alpha_i y_i = 0$. We would like to determine p_1, p_2 such that the difference $Q(\alpha) - Q(\alpha^{old})$ is maximized. Using the first order approximation of the variation of the objective function, we get,

$$Q(\alpha) - Q(\alpha^{old}) \cong (\alpha - \alpha^{old})^T \nabla_\alpha Q(\alpha)\Big|_{\alpha=\alpha^{old}} =$$

$$= \left(\alpha_{p_1} - \alpha_{p_1}^{old}\right)\frac{\partial Q(\alpha)}{\partial \alpha_{p_1}}\Big|_{\alpha=\alpha^{old}} + \left(\alpha_{p_2} - \alpha_{p_2}^{old}\right)\frac{\partial Q(\alpha)}{\partial \alpha_{p_2}}\Big|_{\alpha=\alpha^{old}} =$$

$$= \rho\left[\rho_1 \frac{\partial Q(\alpha)}{\partial \alpha_{p_1}}\Big|_{\alpha=\alpha^{old}} + (1-\rho_1)\frac{\partial Q(\alpha)}{\partial \alpha_{p_2}}\Big|_{\alpha=\alpha^{old}}\right]\left(\frac{\partial Q(\alpha)}{\partial \alpha_{p_1}}\Big|_{\alpha=\alpha^{old}} + \frac{\partial Q(\alpha)}{\partial \alpha_{p_2}}\Big|_{\alpha=\alpha^{old}}\right)$$

Therefore, the indices p_1, p_2 should be taken such that

$$(10)\ 1 \le p_1 \le m, \quad m+1 \le p_2 \le N,$$

$$(11)\ \rho\left[\rho_1 \frac{\partial Q(\alpha)}{\partial \alpha_{p_1}}\Big|_{\alpha=\alpha^{old}} + (1-\rho_1)\frac{\partial Q(\alpha)}{\partial \alpha_{p_2}}\Big|_{\alpha=\alpha^{old}}\right]\left(\frac{\partial Q(\alpha)}{\partial \alpha_{p_1}}\Big|_{\alpha=\alpha^{old}} + \frac{\partial Q(\alpha)}{\partial \alpha_{p_2}}\Big|_{\alpha=\alpha^{old}}\right) > 0$$

and $Q(\alpha) - Q(\alpha^{old})$ is maximized.

In case

$$\rho\left[\rho_1 \frac{\partial Q(\alpha)}{\partial \alpha_{p_1}}\Big|_{\alpha=\alpha^{old}} + (1-\rho_1)\frac{\partial Q(\alpha)}{\partial \alpha_{p_2}}\Big|_{\alpha=\alpha^{old}}\right]\left(\frac{\partial Q(\alpha)}{\partial \alpha_{p_1}}\Big|_{\alpha=\alpha^{old}} + \frac{\partial Q(\alpha)}{\partial \alpha_{p_2}}\Big|_{\alpha=\alpha^{old}}\right) \le 0$$

for all $1 \le p_1 \le m, \quad m+1 \le p_2 \le N$, the search is over and the current parameter α^{old} is taken as an approximation of a local maximum of the objective function.
Putting together these arguments, we arrive to the following learning algorithm.

```
INPUT: $\alpha \leftarrow \mathbf{0}_N$, $\rho > 0$, $0 \le \rho_1 \le 1$
DO
1. Compute the set $S = \{(p_1, p_2)\}$ of pairs of indices for which (10) and (11) hold.
2. IF $S \ne \emptyset$ THEN
    2.1. Select $(p_1, p_2) \in S$ such that $Q(\alpha) - Q(\alpha^{old})$ is maximized
    2.2. Update the parameter:
    $\alpha_i = \alpha_i^{old}$, $1 \le i \le N, i \ne p_1, p_2$
```

$$\alpha_{p_i} = \alpha_{p_i}^{old} + \rho\left[\rho_1 \frac{\partial Q(\alpha)}{\partial \alpha_{p_1}}\Big|_{\alpha=\alpha^{old}} + (1-\rho_1)\frac{\partial Q(\alpha)}{\partial \alpha_{p_2}}\Big|_{\alpha=\alpha^{old}}\right], \quad i = 1,2$$

WHILE ($S \neq \varnothing$)

The learning rates ρ and ρ_1 are taken such that the search process is optimized from both point of views, accuracy and efficiency. Concerning the additional learning rate ρ_1, different heuristically motivated expressions can be proposed. We used two ways to compute ρ_1 aiming to weight the data coming from the classes in terms of sample means and sample covariance matrices. Let us denote by $\hat{\mu}_1, \hat{\mu}_2, \hat{\Sigma}_1, \hat{\Sigma}_2$ the sample means and sample covariance matrices, and by A^+ the Penrose pseudo-inverse of the matrix A,

$$\hat{\mu}_1 = \frac{1}{m}\sum_{i=1}^{m} x_i$$
$$\hat{\mu}_2 = \frac{1}{N-m}\sum_{i=1}^{N-m} x_{m+i}$$
$$\hat{\Sigma}_1 = \frac{1}{m-1}\sum_{i=1}^{m}(x_i - \hat{\mu}_1)(x_i - \hat{\mu}_1)^T$$
$$\hat{\Sigma}_2 = \frac{1}{N-m-1}\sum_{i=1}^{N-m}(x_i - \hat{\mu}_2)(x_i - \hat{\mu}_2)^T$$

The first value of the learning rate ρ_1 is defined in terms of the variation coefficients

$$\theta_i = \hat{\mu}_i^T \left(\hat{\Sigma}_i\right)^+ \hat{\mu}_i,\ i = 1,2$$

by

$$(12)\ \rho_1 = \frac{\theta_2}{\theta_1 + \theta_2}$$

and the value of the second learning rate is set to

$$(13)\ \rho_1 = \frac{\|\hat{\mu}_2\|}{\|\hat{\mu}_1\| + \|\hat{\mu}_2\|}.$$

5 Conclusions and Experimental Results

The main purpose of our experimental developments was to compare the performance of our method, in terms of accuracy and efficiency, for different data sets and the two choices of the learning rate ρ_1 given by (12) and (13)against the Platt's SMO algorithm.

The tests were performed on different data sets, all of them being randomly generated from normal distributions of different variability at the level of the actual distributions and of generated data.

Let $\{x_1, x_2, ..., x_m\}$ and $\{x_{m+1}, x_{m+2}, ..., x_N\}$ be the samples coming from the classes h_1, h_2 respectively. The classes are represented in terms of normal multivariate repartitions $N(\mu_i, \Sigma_i),\ i = 1,2$. The closeness degree between the classes h_1, h_2 is expressed by

$$(14)\ d(h_1, h_2) = (\mu_1 - \mu_2)^T (\Sigma_1 + \Sigma_2)^{-1} (\mu_1 - \mu_2)$$

The MLE of mean vectors and covariance matrices are

$$\hat{\mu}_1 = \frac{1}{m}\sum_{i=1}^{m} x_i$$
$$\hat{\mu}_2 = \frac{1}{N-m}\sum_{i=1}^{N-m} x_{m+i}$$
$$\hat{\Sigma}_1 = \frac{1}{m-1}\sum_{i=1}^{m}(x_i - \hat{\mu}_1)(x_i - \hat{\mu}_1)^T$$
$$\hat{\Sigma}_2 = \frac{1}{N-m-1}\sum_{i=1}^{N-m}(x_i - \hat{\mu}_2)(x_i - \hat{\mu}_2)^T$$

We use two metrics $\hat{d}_1, \hat{d}_2$ to express the closeness degree between the samples $\{x_1, x_2, ..., x_m\}$ and $\{x_{m+1}, x_{m+2}, ..., x_N\}$,

$$(15)\ \hat{d}_1(h_1, h_2) = (\hat{\mu}_1 - \hat{\mu}_2)^T \left(\hat{\Sigma}_1 + \hat{\Sigma}_2\right)^+ (\hat{\mu}_1 - \hat{\mu}_2)$$

and

$$(16)\ \hat{d}_2(h_1, h_2) = \min_{\substack{1 \le i \le m \\ 1 \le j \le N-m}} \|x_i - x_{m+j}\|$$

where $\|\ \|$ is the Euclidean norm. In (15) we used the Penrose pseudo-inverse instead of

the inverse of the matrix $\left(\hat{\Sigma}_1+\hat{\Sigma}_2\right)$ because the matrices $\hat{\Sigma}_1,\hat{\Sigma}_2$ depend on data and it could happen that $\left(\hat{\Sigma}_1+\hat{\Sigma}_2\right)$ is a singular matrix.
We used linear separable test data coming from two dimensional normal distributions and tried to derive conclusions concerning the performance of our learning algorithms.
In the following, we present an experiment where the classes are quite close, the generated sets of examples coming from these two classes happens to be well separated, $\hat{d}_1 > d$ and the distance between the closest pair of examples significantly large.

Test. The examples are randomly generated from two dimensional normal distributions. The set $\mathcal{S}$ consists of 40 examples coming from $N(\mu_1,\Sigma_1)$, and 35 examples coming from $N(\mu_2,\Sigma_2)$, where

$$\mu_1=\begin{pmatrix}11 & 8\end{pmatrix}^T,$$
$$\Sigma_1=\begin{pmatrix}2.89 & 3.825\\ 3.825 & 5.3125\end{pmatrix},$$
$$\mu_2=\begin{pmatrix}5 & -2\end{pmatrix}^T,$$
$$\Sigma_2=\begin{pmatrix}1.4884 & 0\\ 0 & 1.3225\end{pmatrix},$$

therefore $d(h_1,h_2)=15.0970$. The statistical information about $\mathcal{S}$ is

$$\hat{\mu}_1=\begin{pmatrix}11.2169 & 8.2605\end{pmatrix}^T,$$
$$\hat{\Sigma}_1=\begin{pmatrix}2.0805 & 2.5486\\ 2.5486 & 3.3605\end{pmatrix},$$
$$\hat{\mu}_2=\begin{pmatrix}4.8448 & -2.2084\end{pmatrix}^T,$$
$$\hat{\Sigma}_2=\begin{pmatrix}1.0978 & 0.2022\\ 0.2022 & 0.9015\end{pmatrix}.$$
$$\hat{d}_1(h_1,h_2)=25.8224$$
$$\hat{d}_2(h_1,h_2)=5.3680$$

The maximum value of the objective function (6) computed by the SMO algorithm is 0.069406. The values of the objective function (6) computed by our method are represented in Table 1 and Table 2. Note that our method “almost optimized” the objective function Q in only 13 iterations, the value computed by the SMO algorithm in 13 iterations being 0.0563. The number of support vector is 3 and the value of b^* is computed according to (4).
There are a series of similarities and differences between our method and Platt’s SMO algorithm. Concerning the differences, note that, from theoretical point of view, our method does not involve any particular value of the constant C as in case of the SMO algorithm. Also, the value of the parameter η given by (6) does not always assure that the constraints $0\le\alpha_p+y_p\eta\le C$ and $0\le\alpha_q-y_q\eta\le C$ hold, in such a case the SMO algorithm recommend to decrease the value of η.
In case of the SMO algorithm, the examples (x_p,y_p), (x_q,y_q) are looked for in the whole set of examples, while in our method, the examples are searched in each of the two classes respectively. The value of the tolerance parameter τ influences significantly the accuracy and efficiency of the SMO algorithm, small values of τ yielding to better accuracy and increased computational complexity, while larger values of τ degrades the accuracy especially in case of small size data.
The performance of our learning algorithm expressed in terms of the number of iterations required to get a good approximations of the solution of the QP-problem (3) is strongly influenced by the closeness of the sets of examples coming from the two classes. Our algorithm computes a quite accurate approximation in a small number of iterations, only slight adjustments being obtained in case we allow a longer training process.

Table 1. Experimental results of our method in case $\rho_1 = \frac{\|\hat{\mu}_2\|}{\|\hat{\mu}_1\| + \|\hat{\mu}_2\|}$

Number of iterations	ρ	The value of the objective function Q	w^{*T}	b^*
16	**0.005**	**0.066174**	(0.1741 0.3337)	-1.8905
151	0.0005	0.069084	(0.1771 0.3294)	-1.9016
754	0.0001	0.069342	(0.1768 0.3280)	-1.8971
7531	0.00001	0.069400	(0.1769 0.3279)	-1.8973

Table 2. Experimental results of our method in case $\rho_1 = \frac{\theta_2}{\theta_1 + \theta_2}$

Number of iterations	ρ	The value of the objective function Q	w^{*T}	b^*
13	**0.005**	**0.066181**	(0.1728 0.3312)	-1.8761
121	0.0005	0.069085	(0.1769 0.3290)	-1.8994
600	0.0001	0.069342	(0.1769 0.3282)	-1.8980
5987	0.00001	0.069400	(0.1769 0.3280)	-1.8974

References

[1] S. Abe, *Support Vector Machines for Pattern Classification*, Springer, 2010.

E. Alpaydin, *Introduction to Machine Learning*, MIT Press, 2004

[2] G. C. Cawley, N.L.C. Talbot, “Improved sparse least squares support vector machines”,*Neurocomputing*48, pp. 1-4,2002.

[3] L. J. Cao, S. S. Keerthi, C. J. Ong, P. Uvaraj, X. J. Fu, H. P. Lee, “Developing parallel sequential minimal optimization for fast training support vector machine”,*Neurocomputing*70, pp. 93-104, 2006.

[4] S. S. Keerthi, S.K.Shevade,“SMO algorithm for least squares SVM formulations”,*Neural Computing.* 15, pp. 2,2003.

[5] T. Joachims, “Making Large-Scale SVM Learning Practical”, in B. Schőlkopf, C.J. Burges, A.J. Smola, eds. *Advances in Kernel Methods – Support Vector Learning*, Cambridge, MA, MIT Press,1998.

[6] T. Joachims, “A Statistical Learning Model of Text Classification with Support Vector Machines”, *Proceedings of the Conference on Research and Development in Information Retrieval (SIGIR), ACM*, 2001.

[7] P. Laskov, “An improved decomposition algorithm for regression support vector machines”, *Machine Learning*, 46, 2002.

[8] E. Osuna, R. Freund, F.Girosi, “An improved training algorithm for support vector machines”, in *Neural Networks for Signal Processing VII – Proceedings of the 1997 IEEE Workshop*, New York, 1997.

[9] J. Platt, “Fast Training of Support Vector Machines Using Sequential Minimal Optimization”, in: *Advances in Kernel Methods – Support Vector Learning*, Cambridge, MA, MIT Press. (Schőlkopf, B., Burges, C.J., Smola, A.J. eds.),1999.

[10] S. Rueping, “mySVM: another one of those support vector machines”http://www-ai.cs.uni-dortmund. de/SOFTWARE/MYSVM, 2003.

[11] V.D.A. Sanchez, “Advanced support vector machines and kernel methods”, *Neurocomputting*, 55,pp. 5-20, 2003.

[12] B. Schőlkopf, C.J. Burges, A.J. Smola, *Advances in Kernel Methods – Support Vector Learning*, Cambridge, MA, MIT

Press,1999.
[13] L. State, C. Cocianu, D. Fusaru, "A Survey on Potential of the Support Vector Machines in Solving Classification and Regression Problems", in: *Informatica Economică*, nr. 3/2010, pp. 128-139.
[14] J.A.K. Suykens, J. De Brabanter, L. Lukas, "Weighted least squares support vector machines: robustness and sparse approximation", in *Neurocomputing special issue*, 2002.
[15] V. Vapnik, *Estimation of Dependences Based on Empirical Data*, Springer Verlag, Berlin,1982.
[16] V. Vapnik, *Statistical Learning Theory*, John Wiley, N.Y.,1998.
[17] Y. Yue, T. Finley, F. Radlinski, T. Joachims, "A Support Vector Method for Optimizing Average Precision", in *Proceedings of the Conference on Research and Development in Information Retrieval* (SIGIR),2007.
[18] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, N.Y., 1995.

Cătălina-Lucia COCIANU, Professor, PhD, currently working with Academy of Economic Studies, Faculty of Cybernetics, Statistics and Informatics, Department of Informatics in Economy. Competence areas: statistical pattern recognition, digital image processing. Research in the fields of pattern recognition, data mining, signal processing. Author of 12 books and more than 80 papers published in national and international journals.

Luminiţa STATE, Professor, PhD, currently working with University of Pitesti, Department of Mathematics and Computer Science. Competence areas: artificial intelligence, machine learning, statistical pattern recognition, digital image processing. Research in the fields of machine learning, pattern recognition, neural computation. Author of 15 books and more than 120 papers published in national and international journals.

Cristian Răzvan USCATU, Associated Professor, PhD, currently working with the Academy of Economic Studies, Faculty of Cybernetics, Statistics and Informatics, Department of Informatics in Economy. Competence areas: computer programming, data structures. Author and co-author of 10 books and more than 30 papers published in national and international journals.

Raluca-Mariana ŞTEFAN, Assistant Professor, PhD Candidate, currently working with Spiru Haret University, Faculty of Financial Accounting Management, Department of Management and Accounting Management Information Systems. Competence areas: mathematics and various programming languages. Research in the fields of testing statistical symmetry and sorting data structures.