Knowledge About the Margin of an Example

Marcin Orchel

Abstract

In this report, we propose a method called φ support vector classification (φ-SVC) for incorporating knowledge about margin of an example for classification and regression problems. We propose two applications for φ-SVC: decreasing the generalization error of reduced models while preserving the similar number of support vectors, and incorporating the nonlinear constraint of a special type to the problem. The method was tested for support vector machines (SVM) classifier and ε-insensitive support vector regression (ε-SVR). Experiments on real world data sets show decreased generalization error of reduced models for linear and polynomial kernels.

Contents

Chapter 1

Introduction

1.1 Support Vector Classification Basics

For a classification problem, we consider a set of n training vectors $\vec{x_i}$ for $i \in \{1, \ldots, n\}$, where $\vec{x_i} = \left(x_i^1, \ldots, x_i^m\right)$. The i-th training vector is mapped to $y_{\mathrm{c}}^i \in \{-1, 1\}$. The m is a dimension of the problem. The support vector classification (SVC) optimization problem for hard margin case with $\|\cdot\|_1$ norm is

OP 1.

$$\min_{\vec{w_{\mathrm{c}}}, b_{\mathrm{c}}} \quad f\left(\vec{w_{\mathrm{c}}}, b_{\mathrm{c}}\right) = \left\|\vec{w_{\mathrm{c}}}\right\|^2 \tag{1.1}$$

subject to

$$y_{\mathrm{c}}^i h\left(\vec{x_i}\right) \geq 1 \tag{1.2}$$

for $i \in \{1, \ldots, n\}$, where

$$h\left(\vec{x_i}\right) = \vec{w_{\mathrm{c}}} \cdot \vec{x_i} + b_{\mathrm{c}} \ . \tag{1.3}$$

All points must be correctly classified Fig. 1.1a. The SVC soft margin case optimization problem with $\|\cdot\|_1$ norm is

OP 2.

$$\min_{\vec{w_{\mathrm{c}}}, b_{\mathrm{c}}, \vec{\xi_{\mathrm{c}}}} \quad f\left(\vec{w_{\mathrm{c}}}, b_{\mathrm{c}}, \vec{\xi_{\mathrm{c}}}\right) = \frac{1}{2}\left\|\vec{w_{\mathrm{c}}}\right\|^2 + C_{\mathrm{c}} \sum_{i=1}^{n} \xi_{\mathrm{c}}^i \tag{1.4}$$

subject to

$$y_{\mathrm{c}}^i h\left(\vec{x_i}\right) \geq 1 - \xi_{\mathrm{c}}^i \tag{1.5}$$

$$\vec{\xi_{\mathrm{c}}} \geq 0 \tag{1.6}$$

for $i \in \{1, \ldots, n\}$, where

$$h\left(\vec{x_i}\right) = \vec{w_{\mathrm{c}}} \cdot \vec{x_i} + b_{\mathrm{c}} \ , \tag{1.7}$$

$$C_c > 0 \ . \tag{1.8}$$

The $h^*\left(\vec{x}\right) = \vec{w_{\mathrm{c}}^*} \cdot \vec{x} + b_{\mathrm{c}}^* = 0$ is a decision curve of the classification problem. Some of training points can be incorrectly classified Fig. 1.1b.

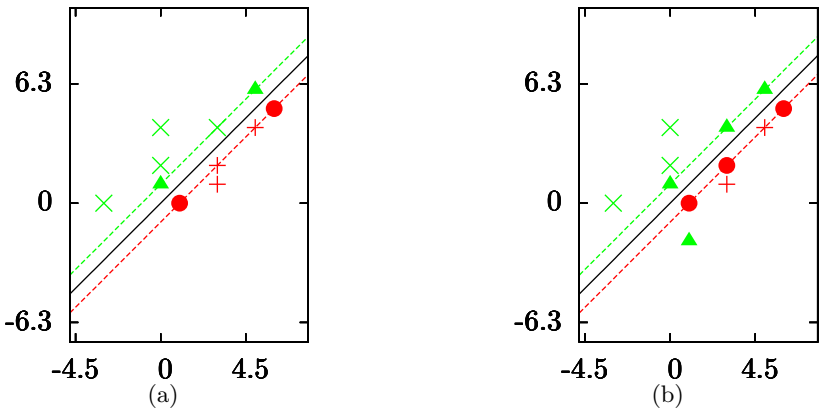


Figure 1.1: Two types of margin classifiers: hard and soft. Example points, support vectors (triangles and circles), solutions (solid lines), margin lines (dashed lines). (a) Hard. (b) Soft. A misclassified point is in (1, -2)

SVC Dual Optimization Problem

The OP 2 optimization problem after transformation into an equivalent dual optimization problem becomes

OP 3.

$$\max_{\vec{\alpha}} \quad f\left(\vec{\alpha}\right) = 1 \cdot \vec{\alpha} - \frac{1}{2}\vec{\alpha^T}\mathbf{Q}\vec{\alpha} \tag{1.9}$$

subject to

$$\vec{\alpha} \cdot \vec{y} = 0 \tag{1.10}$$

$$0 \leq \alpha_i \leq C_c \tag{1.11}$$

where

$$Q_{ij} = y_i y_j K\left(\vec{x_i}, \vec{x_j}\right) \tag{1.12}$$

for all $i, j \in \{1, \ldots, n\}$.

The $\vec{w_c^*} \cdot \vec{x}$ can be computed as

$$\vec{w_c^*} \cdot \vec{x} = \sum_{i=1}^{n} y_c^i \alpha_i^* K\left(\vec{x_i}, \vec{x}\right) \ . \tag{1.13}$$

Therefore, the decision curve is

$$h^*\left(\vec{x}\right) = \sum_{i=1}^{n} y_c^i \alpha_i^* K\left(\vec{x_i}, \vec{x}\right) + b_c^* = 0 \ , \tag{1.14}$$

where α_i are Lagrange multipliers of the dual problem, $K\left(\cdot, \cdot\right)$ is a kernel function, which appears only in the dual problem. *Margin boundaries* are defined as the two hyperplanes

$h(\vec{x}) = -1$ and $h(\vec{x}) = 1$. *Optimal margin boundaries* are defined as the two hyperplanes $h^*(\vec{x}) = -1$ and $h^*(\vec{x}) = 1$. *Geometric margin of the hyperplane* h is defined as $1/\|\vec{w_c}\|$. The i-th training example is *a support vector*, when $\alpha_i^* \neq 0$. A set of support vectors contains all training examples lying below optimal margin boundaries ($y_c^i h^*(\vec{x_i}) < 1$), and part of the examples lying exactly on the optimal margin boundaries ($y_c^i h^*(\vec{x_i}) = 1$), Fig. 1.1b.

SVC Without the Offset

Another variant of SVC is the SVC without the offset b_c, analyzed recently in [13]. The optimization problem is the same except missing b_c term, for the soft case it is

OP 4.

$$\min_{\vec{w_c}, \vec{\xi_c}} \quad f\left(\vec{w_c}, \vec{\xi_c}\right) = \frac{1}{2}\|\vec{w_c}\|^2 + C_c \cdot \vec{\xi_c} \tag{1.15}$$

subject to

$$y_c^i h(\vec{x_i}) \geq 1 - \xi_c^i \tag{1.16}$$

$$\vec{\xi} \geq 0 \tag{1.17}$$

for $i \in \{1, \ldots, n\}$, where

$$h(\vec{x_i}) = \vec{w_c} \cdot \vec{x_i} \ , \tag{1.18}$$

$$C_c > 0 \ . \tag{1.19}$$

The dual problem is

OP 5.

$$\max_{\vec{\alpha}} \quad d(\vec{\alpha}) = \vec{\alpha} - \frac{1}{2}\vec{\alpha^T} Q \vec{\alpha} \tag{1.20}$$

subject to

$$0 \leq \vec{\alpha} \leq C_c \tag{1.21}$$

where $Q_{ij} = y_i y_j K(\vec{x_i}, \vec{x_j})$, for all $i, j \in \{1, \ldots, n\}$.

We can notice missing linear constraint. The decision curve is

$$h^*(\vec{x}) = \sum_{i=1}^{n} y_c^i \alpha_i^* K(\vec{x_i}, \vec{x}) = 0 \ . \tag{1.22}$$

1.2 Support Vector Regression Basics

In a regression problem, we consider a set of training vectors $\vec{x_i}$ for $i \in \{1, \ldots, n\}$, where $\vec{x_i} = \left(x_i^1, \ldots, x_i^m\right)$. The i-th training vector is mapped to $y_{\mathrm{r}}^i \in \mathbb{R}$. The m is a dimension of the problem. The ε-SVR soft case optimization problem is

OP 6.

$$\min_{\vec{w_{\mathrm{r}}}, b_{\mathrm{r}}, \vec{\xi_{\mathrm{r}}}, \vec{\xi_{\mathrm{r}}^*}} \quad f\left(\vec{w_{\mathrm{r}}}, b_{\mathrm{r}}, \vec{\xi_{\mathrm{r}}}, \vec{\xi_{\mathrm{r}}^*}\right) = \frac{1}{2}\left\|\vec{w_{\mathrm{r}}}\right\|^2 + C_{\mathrm{r}} \sum_{i=1}^{n} \left(\xi_{\mathrm{r}}^i + \xi_{\mathrm{r}}^{*i}\right) \tag{1.23}$$

subject to

$$y_{\mathrm{r}}^i - g\left(\vec{x_i}\right) \leq \varepsilon + \xi_{\mathrm{r}}^i \tag{1.24}$$

$$g\left(\vec{x_i}\right) - y_{\mathrm{r}}^i \leq \varepsilon + \xi_{\mathrm{r}}^{i*} \tag{1.25}$$

$$\vec{\xi_{\mathrm{r}}} \geq 0 \tag{1.26}$$

$$\vec{\xi_{\mathrm{r}}^*} \geq 0 \tag{1.27}$$

for $i \in \{1, \ldots, n\}$, where

$$g\left(\vec{x_i}\right) = \vec{w_{\mathrm{r}}} \cdot \vec{x_i} + b_{\mathrm{r}} \ , \tag{1.28}$$

$$\varepsilon \in \mathbb{R} \ . \tag{1.29}$$

The $g^*\left(\vec{x}\right) = \vec{w_{\mathrm{r}}^*} \cdot \vec{x} + b_{\mathrm{r}}^*$ is a regression function. Optimization problem 6 is transformed into an equivalent dual problem. The regression function becomes

$$g^*\left(\vec{x}\right) = \sum_{i=1}^{n} \left(\alpha_i^* - \beta_i^*\right) K\left(\vec{x_i}, \vec{x}\right) + b_{\mathrm{r}}^* \ , \tag{1.30}$$

where α_i, β_i are Lagrange multipliers, $K\left(\cdot, \cdot\right)$ is a kernel function. The ε boundaries are defined as $g\left(\vec{x}\right) - \varepsilon$ and $g\left(\vec{x}\right) + \varepsilon$. The i-th training example is *a support vector*, when $\alpha_i^* - \beta_i^* \neq 0$. For $\varepsilon \geq 0$, a set of support vectors contains all training examples lying outside ε boundaries, and part of the examples lying exactly on ε boundaries, Fig. 1.2. The number of support vectors can be controlled by ε parameter.

1.3 SVM with C_i Weights

A 1-norm soft margin SVC optimization problem for training examples $\vec{x_i}$ with weights C_i is

OP 7.

$$\min_{\vec{w}, b, \vec{\xi}} \quad f\left(\vec{w}, b, \vec{\xi}\right) = \frac{1}{2}\left\|\vec{w}\right\|^2 + \vec{C_c} \cdot \vec{\xi} \tag{1.31}$$

subject to

$$y_i h\left(\vec{x_i}\right) \geq 1 - \xi_i \tag{1.32}$$

$$\vec{\xi} \geq 0 \tag{1.33}$$

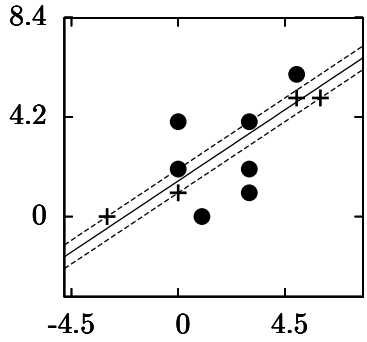


Figure 1.2: The idea of ε-SVR. Points - examples, circles - support vectors, a solid line - a solution, dashed lines - ε boundaries

for $i \in \{1, \ldots, n\}$, where

$$\vec{C_c} \gg 0 \tag{1.34}$$

$$h\left(\vec{x_i}\right) = \vec{w} \cdot \vec{x_i} + b \ . \tag{1.35}$$

The C_i weights were also used with ε-SVR for predicting time series data, [14]. The other types of weights that were used with ε-SVR are ε_i weights per example replacing the parameter ε. They were used for density estimation, [15]. Sometimes we use different ε weights for inequalities (1.24), (1.25), ε_u and ε_d respectively. And finally we can combine above changes and use ε_u^i and ε_d^i weights

OP 8.

$$\min_{\vec{w_{\mathrm{r}}}, b_{\mathrm{r}}, \vec{\xi_{\mathrm{r}}}, \vec{\xi_{\mathrm{r}}^*}} \ f\left(\vec{w_{\mathrm{r}}}, b_{\mathrm{r}}, \vec{\xi_{\mathrm{r}}}, \vec{\xi_{\mathrm{r}}^*}\right) = \frac{1}{2}\left\|\vec{w_{\mathrm{r}}}\right\|^2 + \vec{C_{\mathrm{r}}} \sum_{i=1}^{n} \left(\xi_{\mathrm{r}}^i + \xi_{\mathrm{r}}^{*i}\right) \tag{1.36}$$

subject to

$$y_{\mathrm{r}}^i - g\left(\vec{x_i}\right) \leq \varepsilon_u^i + \xi_{\mathrm{r}}^i \tag{1.37}$$

$$g\left(\vec{x_i}\right) - y_{\mathrm{r}}^i \leq \varepsilon_d^i + \xi_{\mathrm{r}}^{i*} \tag{1.38}$$

$$\vec{\xi_{\mathrm{r}}} \geq 0 \tag{1.39}$$

$$\vec{\xi_{\mathrm{r}}^*} \geq 0 \tag{1.40}$$

for $i \in \{1, \ldots, n\}$, where

$$g\left(\vec{x_i}\right) = \vec{w_{\mathrm{r}}} \cdot \vec{x_i} + b_{\mathrm{r}} \ . \tag{1.41}$$

We can notice that changing y_r^i value by Δy_r^i is equivalent to changing ε_u^i by $-\Delta y_r^i$ and changing ε_d^i by Δy_r^i.

1.4 Introduction to δ-SVR

We consider a set of training vectors $\vec{x_i}$ for $i \in \{1, \ldots, n\}$, where $\vec{x_i} = \left(x_i^1, \ldots, x_i^m\right)$. The i-th training vector is mapped to $y_{\mathrm{r}}^i \in \mathbb{R}$. The δ support vector regression (δ-SVR) method finds a regression function by the following procedure.

1. Every training example $\vec{x_i}$ is duplicated, an output value y_{r}^i is increased by a value of a parameter $\delta \geq 0$ for original training examples, and decreased by δ for duplicated training examples.

2. Every training example $\vec{x_i}$ is converted to a classification example by incorporating the output to the input vector as an additional feature and setting class 1 for original training examples, class -1 for duplicated training examples.

3. The SVC method is launched for a classification problem.

4. The solution of SVC method is converted back to function form.

The result of the first step is a set of training mappings for $i \in \{1, \ldots, 2n\}$

$$\begin{cases} \vec{x_i} \to y_{\mathrm{r}}^i + \delta & \text{for } \; i \in \{1, \ldots, n\} \\ \vec{x_i} \to y_{\mathrm{r}}^i - \delta & \text{for } \; i \in \{n+1, \ldots, 2n\} \end{cases} \tag{1.42}$$

where $\vec{x_{n+i}} = \vec{x_i}$, $y_r^{n+i} = y_r^i$ for $i \in \{1, \ldots, n\}$, $\delta \in \mathbb{R}$. The δ is called *the translation parameter*. The result of the second step is a set of training mappings for $i \in \{1, \ldots, 2n\}$

$$\vec{c_i} = \left(x_i^1, \ldots, x_i^m, y_{\mathrm{r}}^i + y_c^i \delta\right) \to y_c^i \tag{1.43}$$

where $y_c^i = 1$ for $i \in \{1, \ldots, n\}$, and $y_c^i = -1$ for $i \in \{n+1, \ldots, 2n\}$. The dimension of the $\vec{c_i}$ vectors is equal to $m+1$. The set of $\vec{x_i}$ mappings before duplication is called *a regression data setting*, the set of $\vec{c_i}$ ones is called *a classification data setting*. In the third step, OP 2 is solved with $\vec{c_i}$ examples, so we can write OP 2 as

OP 9.

$$\min_{\vec{w_{\mathrm{c}}}, b_{\mathrm{c}}, \vec{\xi_{\mathrm{c}}}} \quad f\left(\vec{w_{\mathrm{c}}}, b_{\mathrm{c}}, \vec{\xi_{\mathrm{c}}}\right) = \frac{1}{2}\left\|\vec{w_{\mathrm{c}}}\right\|^2 + C_{\mathrm{c}} \sum_{i=1}^{2n} \xi_{\mathrm{c}}^i \tag{1.44}$$

subject to

$$y_{\mathrm{c}}^i \left(w_{\mathrm{c,red}} \cdot \vec{x_i} + w_c^{m+1}\left(y_r^i + y_c^i \delta\right) + b_c\right) \geq 1 - \xi_{\mathrm{c}}^i \tag{1.45}$$

$$\vec{\xi_{\mathrm{c}}} \geq 0 \tag{1.46}$$

for $i \in \{1, \ldots, 2n\}$.

The $w_{\mathrm{c,red}}$ is defined as $w_{\mathrm{c,red}} = (w_1, \ldots, w_m)$. The $h^*\left(\vec{x}\right) = \vec{w_{\mathrm{c}}^*} \cdot \vec{x} + b_{\mathrm{c}}^* = 0$ is a decision curve of the classification problem. Note that $h^*\left(\vec{x}\right)$ is in the implicit form of the last coordinate of $\vec{x}$. In the fourth step, an explicit form of the last coordinate needs

to be find. The explicit form is needed for example for testing new examples. The $\vec{w}_{\mathrm{c}}$ variable of the primal problem for a simple linear kernel is found in the following way

$$\vec{w}_{\mathrm{c}} = \sum_{i=1}^{2n} y_{\mathrm{c}}^{i} \alpha_i \vec{c}_i \ . \tag{1.47}$$

For a simple linear kernel the explicit form of (1.14) is

$$x_{m+1} = \frac{-\sum_{j=1}^{m} w_{\mathrm{c}}^{j} x_j - b_{\mathrm{c}}}{w_{\mathrm{c}}^{m+1}} \ . \tag{1.48}$$

The regression solution is $g^*(\vec{x}) = \vec{w}_{\mathrm{r}} \cdot \vec{x} + b_{\mathrm{r}}$, where $w_{\mathrm{r}}^{i} = -w_{\mathrm{c}}^{i}/w_{\mathrm{c}}^{m+1}$, $b_{\mathrm{r}} = -b_{\mathrm{c}}/w_{\mathrm{c}}^{m+1}$ for $i = 1, \ldots, m$. For nonlinear kernels, a conversion to the explicit form has some limitations. First, a decision curve could have more than one value of the last coordinate for specific values of remaining coordinates of $\vec{x}$ and therefore it cannot be converted unambiguously to the function (e.g. a polynomial kernel with a dimension equals to 2). Second, even when the conversion to the function is possible, there is no explicit analytical formula (e.g. a polynomial kernel with a dimension greater than 4), hence a special method for finding the explicit formula of the coordinate should be used, e.g. a bisection method. The disadvantage of this solution is a longer time of testing new examples. To overcome these problems, we proposed to incorporate prior knowledge to the classification problem, that the solution will be always in the form of the function in the chosen direction. Thus, we proposed in [10] a new kernel type in which the last coordinate is placed only inside a linear term. The new kernel is constructed from an original kernel by removing the last coordinate, and adding the linear term with the last coordinate

$$K(\vec{x}, \vec{y}) = K_o(\vec{x_{\mathrm{red}}}, \vec{y_{\mathrm{red}}}) + x_{m+1} y_{m+1} \ , \tag{1.49}$$

where $\vec{x}$ and $\vec{y}$ are $m+1$ dimensional vectors, $\vec{x_{\mathrm{red}}} = (x_1, \ldots, x_m)$, $\vec{y_{\mathrm{red}}} = (y_1, \ldots, y_m)$, $K_{\mathrm{o}}(\cdot, \cdot)$ is the original kernel from which the new one was constructed. For the most popular kernels polynomial, radial basis function (RBF) and sigmoid, the conversions are respectively

$$(\vec{x} \cdot \vec{y})^d \quad \to \quad \left(\sum_{i=1}^{m} x_i y_i \right)^d + x_{m+1} y_{m+1} \ , \tag{1.50}$$

$$\exp - \frac{\|\vec{x} - \vec{y}\|^2}{2\sigma^2} \quad \to \quad \exp - \frac{\sum_{i=1}^{m} (x_i - y_i)^2}{2\sigma^2} + x_{m+1} y_{m+1} \ , \tag{1.51}$$

$$\tanh \vec{x}\vec{y} \quad \to \quad \tanh \sum_{i=1}^{m} x_i y_i + x_{m+1} y_{m+1} \ , \tag{1.52}$$

where $\vec{x}$ and $\vec{y}$ are $m+1$ dimensional vectors. For the new kernel type, the explicit form of (1.14) for δ-SVR is

$$x_{m+1} = \frac{-\sum_{i=1}^{2n} y_{\mathrm{c}}^{i} \alpha_i K_{\mathrm{o}}(\vec{x_i}, \vec{x_{\mathrm{red}}}) - b_{\mathrm{c}}}{\sum_{i=1}^{2n} y_{\mathrm{c}}^{i} \alpha_i c_i^{m+1}} \ . \tag{1.53}$$

1.4.1 Weighting the Translation Parameter

We can consider incorporating prior knowledge by setting different values of the translation parameter for each example, so we can have δ_i parameters, for $i \in \{1, \ldots, n\}$ and the same parameters for $i \in \{n+1, \ldots, 2n\}$. We can also consider setting different values of δ for up and down translations, so we can have two parameters: δ_{u} and δ_{d}. And finally we can also consider the parameters δ_{u}^i and δ_{d}^i (additionally with the $\vec{C_c}$ weight)

OP 10.

$$\min_{\vec{w_{\mathrm{c}}}, b_{\mathrm{c}}, \vec{\xi_{\mathrm{c}}}} \quad f\left(\vec{w_{\mathrm{c}}}, b_{\mathrm{c}}, \vec{\xi_{\mathrm{c}}}\right) = \frac{1}{2}\left\|\vec{w_{\mathrm{c}}}\right\|^2 + \vec{C_{\mathrm{c}}} \sum_{i=1}^{2n} \xi_{\mathrm{c}}^i \tag{1.54}$$

subject to

$$y_{\mathrm{c}}^i \left(\vec{w_{\mathrm{c,red}}} \cdot \vec{x_i} + w_c^{m+1}\left(y_r^i + y_c^i \delta_i\right) + b_c\right) \geq 1 - \xi_{\mathrm{c}}^i \tag{1.55}$$

$$\vec{\xi_{\mathrm{c}}} \geq 0 \tag{1.56}$$

for $i \in \{1, \ldots, 2n\}$, where $\delta_i = \delta_u^i$ for $i \in \{1, \ldots, n\}$ and $\delta_i = \delta_d^i$ for $i \in \{n+1, \ldots, 2n\}$.

Chapter 2

Knowledge About a Margin

We define the margin of an example (sometimes called just a margin) in the following way:

Definition 2.0.1. Given some curve $h(\vec{x}) = 0$, *the margin of the* $\vec{x_p}$ *example* is defined as a value $|h(\vec{x_p})|$.

For hard margin SVC, OP 1, the margin of the closest examples is equal to 1. *The knowledge about the margin of an example* (sometimes called the knowledge about a margin) is defined as prior information about the margins of particular examples.

2.1 Introduction to φ-SVC

The φ-SVC method is a recently proposed method of incorporating knowledge about the margin of an example to SVC, [8, 9, 11]. The φ-SVC optimization problem is defined with an additional parameter per example added in the right side of the inequality (1.5). Another modification of inequality constraints was proposed in [17]. The authors modify the inequalities by multiplying the left side of the inequalities by some monotonically decreasing function of additional example weights. The φ-SVC is a more general concept of weights per example with any values possible and with different interpretation.

Now, we will closely look at φ-SVC optimization problem. We define φ-SVC optimization problem based on SVC with cost weights per example OP 7 as

OP 11.

$$\min_{\vec{w_c}, b_c, \vec{\xi_c}} \quad f\left(\vec{w_c}, b_c, \vec{\xi_c}\right) = \frac{1}{2}\left\|\vec{w_c}\right\|^2 + \vec{C_c} \cdot \vec{\xi_c} \tag{2.1}$$

subject to

$$y_c^i h(\vec{x_i}) \geq 1 + \varphi_i - \xi_c^i \tag{2.2}$$

$$\vec{\xi_c} \geq 0 \tag{2.3}$$

for $i \in \{1, \ldots, n\}$, where

$$\vec{C_c} \gg 0 \tag{2.4}$$

$$\varphi_i \in \mathbb{R} \tag{2.5}$$

$$h\left(\vec{x_i}\right) = \vec{w_c} \cdot \vec{x_i} + b_c \ . \tag{2.6}$$

The new weights φ_i are present only in (2.2). When $\vec{\varphi} = 0$, the OP 11 is equivalent to OP 7. When all φ_i are equal to some constant $\varphi > -1$, we will get the same decision boundary as for $\varphi = 0$ when we change $\vec{C_c}$ to $\vec{C_c}/\left(1+\varphi\right)$.

Proof. We will prove that we get the same decision boundary when we replace 1 with $1/d$, for some $d > 0$, $\varphi_i = 0$. Let's replace ξ_i with ξ_i/d and we get the inequalities $y_c^i h\left(\vec{x_i}\right) \geq 1/d - \xi_c^i/d$. After multiplying by d we get $y_c^i dh\left(\vec{x_i}\right) \geq 1 - \xi_c^i$. The objective function can be multiplied by d^2 and we get $\frac{1}{2}\left\|d\vec{w_c}\right\|^2 + d\vec{C_c} \cdot \vec{\xi_c}$, the inequalities $\xi_i/d \geq 0$ can be replaced by $\xi_i \geq 0$. So we get the same optimization problem as the original one with the new $\vec{C_c} = d\vec{C_c}$ and with the new decision curve $dh\left(\vec{x}\right) = 0$. □

We can notice that when neglecting ξ_c^i, we get different bounds for the margin: when $y_c^i = 1$ and $g\left(\vec{x_i}\right) \geq 0$, then we get a lower bound $1 + \varphi_i$, when $y_c^i = -1$ and $g\left(\vec{x_i}\right) \geq 0$, then we get an upper bound $-\left(1+\varphi_i\right)$, when $y_c^i = 1$ and $g\left(\vec{x_i}\right) < 0$, then we get an upper bound $-\left(1+\varphi_i\right)$, when $y_c^i = -1$ and $g\left(\vec{x_i}\right) < 0$, then we get a lower bound $1+\varphi_i$. We can distinguish three cases: $1+\varphi_i > 0$, $1+\varphi_i < 0$ and $1+\varphi_i = 0$. For the first one, we get a lower bound on the margin equal to $1+\varphi_i$. For the second, we get an upper bound on the margin equal to $-\left(1+\varphi_i\right)$, for the third, we get an upper bound on the margin equal to 0. Therefore, by using φ_i weights, we can incorporate knowledge about the margin of an example.

The next property of φ_i weights is that they have impact on the distance between the curve $h\left(\vec{x}\right) = 0$ and the i-th example. We can conduct the same analysis as above for distances by dividing both sides of (2.2) by $\left\|\vec{w_c}\right\|$, so we get $y_c^i h\left(\vec{x_i}\right) / \left\|\vec{w_c}\right\| \geq 1/\left\|\vec{w_c}\right\| + \varphi_i/\left\|\vec{w_c}\right\|$. The conclusion is similar: for the case when $1+\varphi_i > 0$, we get a lower bound on the distance equal to $\left(1+\varphi_i\right)/\left\|\vec{w_c}\right\|$. For the case when $1+\varphi_i < 0$, we get an upper bound on the distance equal to $-\left(1+\varphi_i\right)/\left\|\vec{w_c}\right\|$. For the case when $1+\varphi_i = 0$, we get an upper bound on the distance equal to 0.

Note that when we take into account ξ_c^i, we can see that knowledge about the margin of an example is incorporated as imperfect prior knowledge. The violation of knowledge about the margin of an example is controlled by the C_c^i parameters. Comparing loosely φ_i weights with slack variables: φ_i weights are constant, they are absent in the objective function, whereas a sum of slack variables is minimized as part of the objective function.

We can also derive the equivalent optimization problem to OP 11, where φ_i weights are present in the constraints with slack variables

OP 12.

$$\min_{\vec{w_c}, b_c, \vec{\xi_c}} \quad f\left(\vec{w_c}, b_c, \vec{\xi_c}\right) = \frac{1}{2}\left\|\vec{w_c}\right\|^2 + \vec{C_c} \cdot \vec{\xi_c} \tag{2.7}$$

subject to

$$y_c^i h\left(\vec{x_i}\right) \geq 1 - \xi_c^i \tag{2.8}$$

$$\vec{\xi} \geq \varphi_i \tag{2.9}$$

for $i \in \{1, \ldots, n\}$.

In order to construct an efficient algorithm for the OP 11 its dual form was derived (derivation in A.1). The final form of the dual problem is

OP 13.

$$\max_{\vec{\alpha}} \quad d\left(\vec{\alpha}\right) = \vec{\alpha} \cdot \left(1 + \vec{\varphi}\right) - \frac{1}{2}\vec{\alpha^T} Q \vec{\alpha} \tag{2.10}$$

subject to

$$\vec{\alpha} \cdot \vec{y_{\mathrm{c}}} = 0 \tag{2.11}$$

$$0 \leq \vec{\alpha} \leq \vec{C} \ , \tag{2.12}$$

where

$$Q_{ij} = y_{\mathrm{c}}^i y_{\mathrm{c}}^j \left(\vec{x_i} \cdot \vec{x_j}\right) \tag{2.13}$$

for all $i, j \in \{1, \ldots, n\}$.

It differs from the original SVC dual form OP 3 by only $\vec{\alpha} \cdot \vec{\varphi}$ term (also here we use C_{c}^i weights instead of C_{c}). In the above formulation, similarly as for the original SVC, it is possible to introduce nonlinear decision functions by using a kernel function instead of a scalar product. The final decision boundary has a form

$$h^*\left(\vec{x}\right) = \sum_{i=1}^{n} y_{\mathrm{c}}^i \alpha_i^* K\left(\vec{x_i}, \vec{x}\right) + b^* = 0 \ , \tag{2.14}$$

where $K\left(\cdot, \cdot\right)$ is a kernel function. The Karush-Kuhn-Tucker (KKT) complementary condition is

$$\alpha_i \left(y_{\mathrm{c}}^i h\left(\vec{x_i}\right) - 1 - \varphi_i + \xi_{\mathrm{c}}^i\right) = 0 \tag{2.15}$$

$$\left(C_i - \alpha_i\right) \xi_{\mathrm{c}}^i = 0 \ . \tag{2.16}$$

The conclusions from (2.15) and (2.16) are: when $\alpha_i = 0$, then $\xi_{\mathrm{c}}^i = 0$, when $0 < \alpha_i < C_{\mathrm{c}}$, then $\xi_{\mathrm{c}}^i = 0$ and $y_{\mathrm{c}}^i h\left(\vec{x_i}\right) = 1 + \varphi_i$, when $\alpha_i = C_{\mathrm{c}}$, then $y_{\mathrm{c}}^i h\left(\vec{x_i}\right) = 1 + \varphi_i - \xi_{\mathrm{c}}^i$. Moreover, when $\xi_{\mathrm{c}}^i > 0$, then $\alpha_i = C_{\mathrm{c}}$ and $y_{\mathrm{c}}^i h\left(\vec{x_i}\right) = 1 + \varphi_i - \xi_{\mathrm{c}}^i$, when $y_{\mathrm{c}}^i h\left(\vec{x_i}\right) > 1 + \varphi_i - \xi_{\mathrm{c}}^i$, then $\alpha_i = 0$, $\xi_{\mathrm{c}}^i = 0$ and $y_{\mathrm{c}}^i h\left(\vec{x_i}\right) > 1 + \varphi_i$. We can find ξ_{c}^i parameters from the solution of the dual form as following. When

$$y_{\mathrm{c}}^i h^*\left(\vec{x_i}\right) \geq 1 + \varphi_i \ , \tag{2.17}$$

then $\xi_{\mathrm{c}}^i = 0$, else

$$\xi_{\mathrm{c}}^i = 1 + \varphi_i - y_{\mathrm{c}}^i h\left(\vec{x_i}\right) \ . \tag{2.18}$$

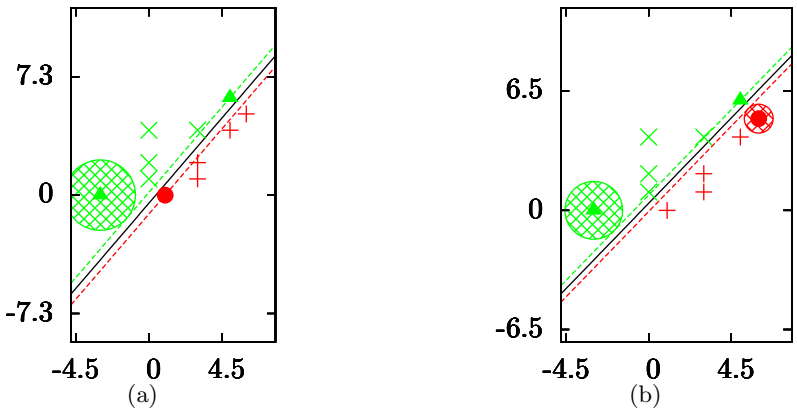


Figure 2.1: Interpretation of knowledge about a margin as dynamic hyperspheres. Points - examples, solid lines - solutions, triangles and circles - support vectors, circles filled with grid pattern - dynamic hyperspheres. The example in (-3, 0) has $\varphi_i = 5.0$ and it is a marginal support vector. The hyperspheres must not cross the appropriate margin boundaries. (a) $C = 100$. (b) $C = 10$ and $\varphi_i = 2.5$ in (6,5). A radius of a dynamic hypersphere in (-3, 0) differs in both cases, in (a) it is 2.2, and in (b) it is 1.6

2.2 Knowledge About the Margin as Dynamic Hyperspheres

Knowledge about the margin of the p-th example can be visualized as a hypersphere with the center in the p point and a radius equal to $\varphi_p / \|\vec{w}_c\|$. We call it *a dynamic hypersphere*, because a radius depends on $\|\vec{w}_c\|$. For the two solution candidates $h_1(\vec{x}) = 0$ and $h_2(\vec{x}) = 0$, where $h_2(\vec{x}) = a h_1(\vec{x})$ and $a \neq 0$ (both hyperplanes have the same geometric location), the hyperspheres are respectively $S_1(\vec{x_p}, r)$, and $S_2(\vec{x_p}, r/a)$, Fig. 2.1. Knowledge about the margin of an example in the form of a lower bound on the margin can be interpreted as a constraint that the hypersphere must not cross the appropriate margin boundary, Fig. 2.1. Knowledge about the margin of an example in the form of an upper bound on the margin can be interpreted as a constraint that the hypersphere must cross the appropriate margin boundary, Fig. 2.2.

2.3 Solving φ-SVC Optimization Problem

For solving OP 13, a decomposition method similar to sequential minimal optimization (SMO), [12] was derived. In every step, two parameter subproblems are solved. Heuristic and stopping criterion are based on KKT conditions.

2.4 New Types of Support Vectors

The i-th example is a support vector, when $\alpha_i^* \neq 0$. From KKT (2.15),(2.16), we can conclude which examples could be support vectors. In the original SVC, only the example that lies on the optimal margin boundaries ($y_i h^*(\vec{x_i}) = 1$) or below optimal

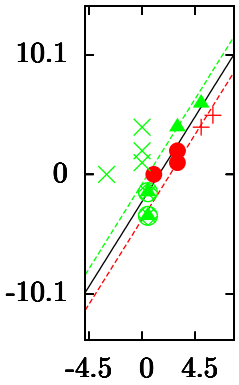


Figure 2.2: Interpretation of knowledge about a margin as dynamic hyperspheres. Points - examples, a solid line - a solution, triangles and circles - support vectors, circles filled with grid pattern - dynamic hyperspheres. The example in (0.5, -1.5) has $\varphi_i = -1.0$. Its hypersphere must cross the appropriate margin boundary

margin boundaries ($y_i h^* (\vec{x_i}) < 1$) could be a support vector. In φ-SVC, also the example satisfying $\varphi_i > 0$ and lying above margin boundaries ($y_i h^* (\vec{x_i}) > 1$) could be a support vector. Such example is called *marginal support vector*, Fig. 2.1. Because an output model is formulated based on support vectors, introducing the new type of support vectors leads to richer models with additional examples lying above optimal margin boundaries.

Examples with upper bounded margins could be the examples that lie below the margin boundary and are not support vectors. Moreover, such examples could lead to the new type of support vectors – that have slack variables equal to zero and lie below the margin boundaries.

The better flexibility of φ-SVC in choosing support vectors suggests that we can achieve solutions with fewer support vectors for the similar generalization performance.

2.5 Reformulation of ε-SVR as φ-SVC

We have found that the ε-SVR is a special case of φ-SVC, [11]. The similar reformulation was previously implemented in LibSVM, [1] for solving ordinal classification problems – without prior knowledge, and ε-SVR. The consequence of this reformulation is that we can apply all the applications for φ-SVC also for ε-SVR.

Here we present a reformulation of the OP 6 (derivation in A.2). Every regression training example is duplicated, Fig. 2.3. Every original training example gets 1 class, and the duplicated training example gets -1 class and therefore we get

OP 14.

$$\min_{\vec{w}_c, b_c, \vec{\xi}_c} \quad f\left(\vec{w}_c, b_c, \vec{\xi}_c\right) = \frac{1}{2}\left\|\vec{w}_c\right\|^2 + C_c \sum_{i=1}^{2n} \xi_c^i \tag{2.19}$$

subject to

$$y_{\mathrm{c}}^{i} h\left(\vec{x_i}\right) \geq 1 - \xi_{\mathrm{c}}^{i} + \varphi_i \tag{2.20}$$

$$\vec{\xi_{\mathrm{c}}} \geq 0 \tag{2.21}$$

for $i \in \{1, \ldots, 2n\}$, where

$$h\left(\vec{x_i}\right) = \vec{w_{\mathrm{c}}} \cdot \vec{x_i} + b_{\mathrm{c}} \tag{2.22}$$

and

$$\varphi_i = y_{\mathrm{c}}^{i} y_r^{i} - \varepsilon - 1 \ . \tag{2.23}$$

The OP 14 is a special case of OP 11. Instead of using a decision curve of OP 14 we use a regression function

$$\sum_{i=1}^{2n} y_{\mathrm{c}}^{i} \alpha_{c,i}^{*} K\left(\vec{x_i}, \vec{x}\right) + b_{\mathrm{c}}^{*} = 0 \ \rightarrow \ g^{*}\left(\vec{x}\right) = \sum_{i=1}^{n} y_{\mathrm{c}}^{i} \alpha_{r,i}^{*} K\left(\vec{x_i}, \vec{x}\right) + \sum_{i=n+1}^{2n} y_{\mathrm{c}}^{i} \beta_{r,i-n}^{*} K\left(\vec{x_i}, \vec{x}\right) + b_r^{*} \ , \tag{2.24}$$

where $\alpha_{r,i}^{*} = \alpha_{c,i}^{*}$, $\beta_{r,i-n}^{*} = \alpha_{c,i}^{*}$, $b_r^{*} = b_{\mathrm{c}}^{*}$. Because data are duplicated we can merge the sums in the final regression and we get

$$g^{*}\left(\vec{x}\right) = \sum_{i=1}^{n} \left(\alpha_{r,i}^{*} - \beta_{r,i}^{*}\right) K\left(\vec{x_i}, \vec{x}\right) + b_r^{*} \ . \tag{2.25}$$

In a typical scenario $\varphi_i < 0$, because ε is close to 0 and y_r^{i} is less than 1. We can notice the following property of the OP 14. Because every training example is duplicated, for every possible solution, n training examples will be always incorrectly classified except those lying on a classification decision boundary, Fig. 2.3. The KKT complementary condition for ε-SVR is the same as for φ-SVC after reformulation. From (2.15) and (2.16), we get

$$\alpha_i \left(h\left(\vec{x_i}\right) - y_r^{i} + \varepsilon + \xi_i\right) = 0 \tag{2.26}$$

$$\beta_i \left(-h\left(\vec{x_i}\right) + y_r^{i} + \varepsilon + \xi_i^{*}\right) = 0 \tag{2.27}$$

$$\left(C_{\mathrm{c}} - \alpha_i\right) \xi_i = 0 \ . \tag{2.28}$$

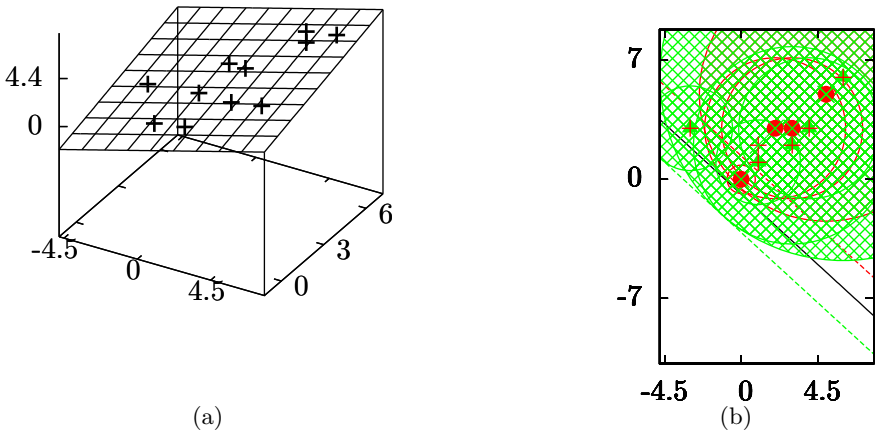
$$\left(C_{\mathrm{c}} - \beta_i\right) \xi_i^{*} = 0 \ . \tag{2.29}$$

We can also analyze some properties of the variable $\gamma_r^{i} = \alpha_{r,i} - \beta_{r,i}$. When $\gamma_r^{i} = -C_{\mathrm{c}}$, then $\alpha_i = 0$ and $\beta_i = C_{\mathrm{c}}$, so $g\left(\vec{x_i}\right) - y_r^{i} \geq \varepsilon$. When $\gamma_r^{i} = C_{\mathrm{c}}$, then $\alpha_i = C_{\mathrm{c}}$ and $\beta_i = 0$, so $g\left(\vec{x_i}\right) - y_r^{i} \leq \varepsilon$. When $\gamma_r^{i} = 0$, then $\alpha_i = \beta_i$. When $\alpha_i = \beta_i < C_{\mathrm{c}}$, then $\xi_i = 0$ and $\xi_i^{*} = 0$. Because $-\varepsilon - \xi_r^{*} \leq g\left(\vec{x_i}\right) - y_r^{i} \leq \varepsilon + \xi_r^{*}$, so $-\varepsilon \leq g\left(\vec{x_i}\right) - y_r^{i} \leq \varepsilon$. The case when $\alpha_i = \beta_i = C_{\mathrm{c}}$ is possible only when $\xi_i + \xi_i^{*} = -2\varepsilon$, then $g\left(\vec{x_i}\right) - y_r^{i} = -\varepsilon - \xi_i$. When $-C_{\mathrm{c}} < \gamma_r^{i} < 0$, then $\alpha_i < \beta_i$, so $\alpha_i < C_{\mathrm{c}}$ and $\beta_i > 0$. From KKT we have

$$-g\left(\vec{x_i}\right) + y_r^{i} + \varepsilon + \xi_i^{*} = 0$$

and

$$\alpha_i \left(g\left(\vec{x_i}\right) - y_r^{i} + \varepsilon\right) = 0$$



(a) (b)

Figure 2.3: The idea of reformulating ε-SVR as φ-SVC. (a) Points - regression examples, a plane - a solution. (b) The transformed problem. Points - classification examples, triangles and circles - support vectors, circles filled with grid pattern - dynamic hyperspheres, a solid line - a solution, dashed lines - margin boundaries

Merging both we get

$$\alpha_i \left(2\varepsilon + \xi_i^*\right) = 0$$

So $\xi_i^* = -2\varepsilon$ or $\alpha_i = 0$. For the second case, $\beta < C_\mathrm{c}$, because $\gamma_r^i > -C_\mathrm{c}$, so

$$g\left(\vec{x_i}\right) - y_r^i = \varepsilon \ .$$

Finally for the case when $0 < \gamma_r^i < C_\mathrm{c}$, then $\alpha_i > \beta_i$, so $\alpha_i > 0$ and $\beta_i < C_\mathrm{c}$. From KKT we have

$$g\left(\vec{x_i}\right) - y_r^i + \varepsilon + \xi_i = 0$$

and

$$\beta_i \left(-g\left(\vec{x_i}\right) + y_r^i + \varepsilon\right) = 0$$

Merging both we get

$$\beta_i \left(2\varepsilon + \xi_i\right) = 0$$

So $\xi_i = -2\varepsilon$ or $\beta_i = 0$. For the second case, $\alpha < C_\mathrm{c}$, because $\gamma_r^i < C_\mathrm{c}$, so

$$g\left(\vec{x_i}\right) - y_r^i = -\varepsilon \ .$$

2.6 Using Knowledge About a Margin with ε-SVR

The ε-SVR can be reformulated as φ-SVC. We can additionally modify margin weights for incorporating knowledge about a margin. There are two options of incorporation

available, either we can modify φ_p and φ_{p+n} for some vector p after transforming the problem into φ-SVC or we can modify ε_u^p or ε_d^p in OP 8 before the transformation. In the first option, the φ_p and φ_{p+n} weights are already set according to (2.23). So adding knowledge about a margin means the manipulation of these weights. In the second option, ε_u^p or ε_d^p weights are set to some ε value for standard ε-SVR. Both approached are equivalent, in the sense that modification of ε_u^p and ε_d^p is equivalent to the modification

$$\Delta\varphi_p = -\Delta\varepsilon_d^p \ , \tag{2.30}$$

$$\Delta\varphi_{p+n} = -\Delta\varepsilon_u^p \ . \tag{2.31}$$

2.7 Using Knowledge About a Margin with δ-SVR

The δ-SVR is transformed into the classification problem, so we can use φ-SVC instead of standard SVC for incorporating knowledge about a margin. From some point p, we set φ_p for the original example, and φ_{p+n} for the duplicated one.

2.8 Changing the Output Curve

After modification of φ_i weights, either the output curve stays the same, or it is changed. For example, let's assume that we modify only a one example $\vec{p}$ and $\varphi_{\vec{p}}$ is equal to zero before the modification. When $y_{\vec{p}}h^*\left(\vec{p}\right) > 1$, then after setting $0 < \varphi_{\vec{p}} \leq y_{\vec{p}}h^*\left(\vec{p}\right) - 1$ the solution remains the same. When we set $\varphi_{\vec{p}} > y_{\vec{p}}h^*\left(\vec{p}\right) - 1$, the solution will be different, but not necessarily a decision boundary. Particularly, setting $\varphi_{\vec{p}} > 0$ can increase a slack variable, when a value of $C_{\vec{p}}$ is small, or it can cause decrease of the geometric margin (increase of $\|\vec{w_c}\|$), and in both cases the solution can stay the same. We can see the example of changed output curve for φ-SVC, Fig. 2.4, for δ-SVR, Fig. 2.5, for ε-SVR, Fig. 2.6.

Let's now analyze in details changing the output curve. First consider changing the $\vec{C_c}$ parameter. We can see that for OP 7, if we increase value of C_c^p for some point p for which $\xi_c^p = 0$, then the solution remains the same. Consider now OP 11 and the solution with all $\varphi_i = 0$. When we set $\varphi_p > 0$ for some example p, such as

$$\varphi_p \leq y_c^p h\left(x_c^p\right) - 1 + \xi_c^p \ , \tag{2.32}$$

the solution remains the same, in the other case variables $\vec{w}, \vec{\xi}, b$ will be adjusted, but the decision curve can stay the same. It will stay the same in the following cases:

1. The ξ_c^p is only adjusted, to

$$\xi_{cnew}^p = \varphi_p - y_c^p h\left(\vec{x_p}\right) + 1 \ . \tag{2.33}$$

The objective function will raise by

$$\Delta f = C_p \Delta\xi_c^p \ . \tag{2.34}$$

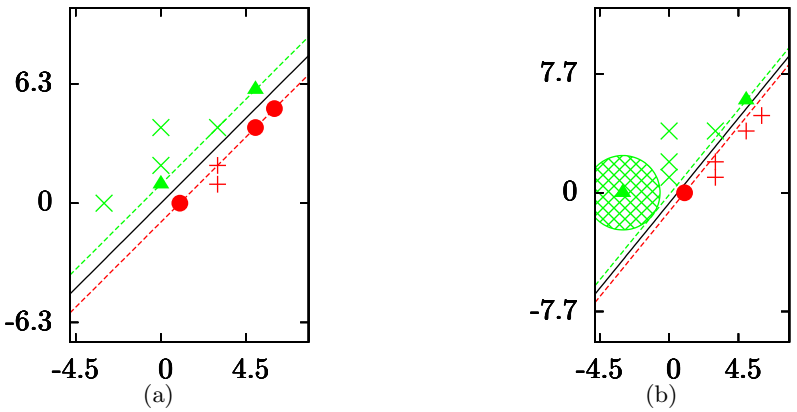


Figure 2.4: Direct curve manipulation for SVC. Points - examples, triangles and circles - support vectors, circles filled with grid pattern - dynamic hyperspheres, solid lines - solutions, dashed lines - margin boundaries. (b) The example with φ_i weight caused lowering the solution from (a)

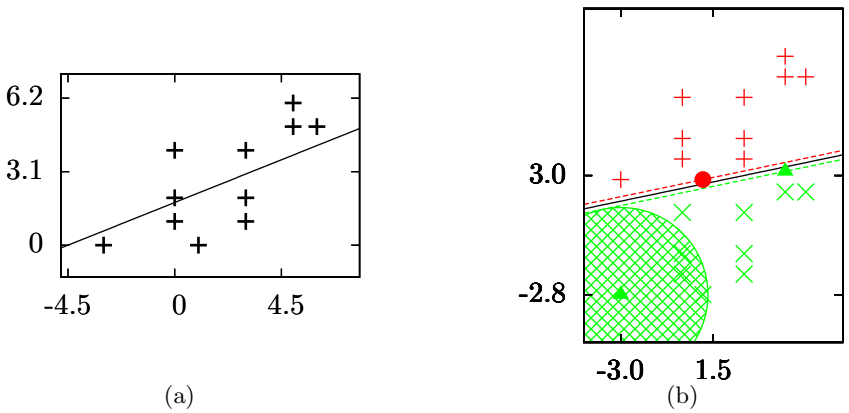


Figure 2.5: Direct curve manipulation for δ-SVR. (a) Points - regression examples, a solid line - a solution. (b) The transformed problem. Points - classification examples, triangles and circles - support vectors, circles filled with grid pattern - dynamic hyperspheres, a solid line - a solution, dashed lines - margin boundaries. The example with φ_i weight caused changing the solution from (a)

2. The $h\left(\vec{x_c}\right)$ is modified

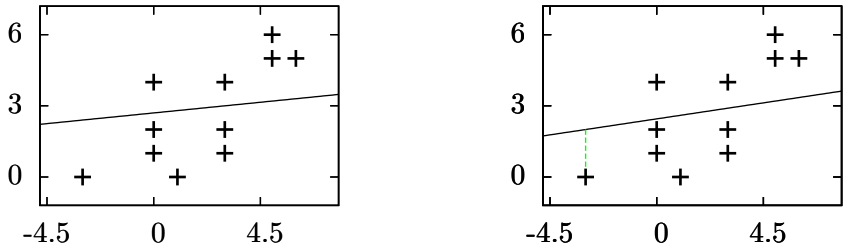
$$h_{cnew}\left(\vec{x_c}\right) = ch\left(\vec{x_c}\right) \quad , \tag{2.35}$$

where $c > 1$. The objective function will raise by

$$\Delta f = \frac{1}{2}\left\|\vec{w}\right\|^2\left(c^2 - 1\right) \quad . \tag{2.36}$$

The modification parameter c is set in a way that the following equation is satisfied

$$cy_c^p h\left(\vec{x_p}\right) = 1 - \xi_c^p + \varphi_p \quad . \tag{2.37}$$



(a) (b)

Figure 2.6: Direct curve manipulation for ε-SVR. Points - regression examples, solid lines - solutions. (b) A dotted line must cross the solution. The example with φ_i weight caused changing the solution from (a)

Note that in this case other non-zero ξ_c^i values have to be modified

$$\xi_c^{iold} = 1 - y_c^i h\left(\vec{x_i}\right) \tag{2.38}$$

$$\xi_c^{inew} = 1 - c y_c^i h\left(\vec{x_i}\right) \tag{2.39}$$

$$\Delta\xi_c^i = y_c^i h\left(\vec{x_i}\right)\left(1-c\right) \quad . \tag{2.40}$$

For misclassified examples the change of an objective function will be positive, otherwise negative

$$\Delta f = \frac{1}{2}\left\|\vec{w}\right\|^2\left(c^2-1\right) + \sum_{i=1, i\neq p}^{n} C_c^i \Delta\xi_c^i \quad . \tag{2.41}$$

3. Both above changes are present simultaneously. The objective function will be changed by

$$\Delta f = \frac{1}{2}\left\|\vec{w}\right\|^2\left(c^2-1\right) + C_c^p\Delta\xi_c^p + \sum_{i=1, i\neq p}^{n} C_c^i \Delta\xi_c^i \quad . \tag{2.42}$$

Changing the decision curve is possible, especially when Δf is high, because other solutions, which are able to change the decision curve, are more likely to have a better value of f. We can see that we can increase a chance for changing the decision boundary by increasing the parameter C_c^p.

2.9 Incorporating Linear Dependency of Function Values

Another example of application for knowledge about the margin of an example is the incorporation of the additional constraint in the form of a constant value of linear de-

pendency on function values, that the solution must satisfy, for regression it is

$$\sum_{i=1}^{s} s_i g\left(\vec{d}_i\right) = e \ , \tag{2.43}$$

where s_i are some parameters, for which $\sum_{i=1}^{s} s_i \neq 0$, d_i are some points, s is the number of d_i points, e is a parameter, g is defined in (1.28), and for classification it is

$$\sum_{i=1}^{s} s_i h\left(\vec{d}_i\right) = e \ . \tag{2.44}$$

where s_i are some parameters, for which $\sum_{i=1}^{s} s_i \neq 0$, d_i are some points, s is the number of d_i points, e is a parameter.

We will show how to incorporate this constraint to φ-SVC, δ-SVR, and ε-SVR. Knowledge about the margin of an example is used in incorporation of this constraint to φ-SVC and ε-SVR.

2.9.1 Incorporating Linear Dependency on Function Values to φ-SVC

Incorporating (2.44) to OP 11 leads to the φ-SVC optimization problem without the offset, which is the SVC optimization problem without the offset OP 4 with additional margin weights (and $\vec{C}_{\mathrm{c}}$ weights for completeness)

OP 15.

$$\min_{\vec{w}_{\mathrm{c}}, \vec{\xi}_{\mathrm{c}}} \quad f\left(\vec{w}_{\mathrm{c}}, \vec{\xi}_{\mathrm{c}}\right) = \frac{1}{2}\left\|\vec{w}_{\mathrm{c}}\right\|^2 + \vec{C}_{\mathrm{c}} \cdot \vec{\xi}_{\mathrm{c}} \tag{2.45}$$

subject to

$$y_{\mathrm{c}}^i h\left(\vec{x}_i\right) \geq 1 - \xi_{\mathrm{c}}^i + \varphi_i \tag{2.46}$$

$$\vec{\xi} \geq 0 \tag{2.47}$$

for $i \in \{1, \ldots, n\}$, where

$$\vec{C}_{\mathrm{c}} \gg 0 \tag{2.48}$$

$$\varphi_{\mathrm{c}}^i \in \mathbb{R} \tag{2.49}$$

$$h\left(\vec{x}_i\right) = \vec{w}_{\mathrm{c}} \cdot \vec{x}_i \ . \tag{2.50}$$

The dual problem compared to OP 5 contains additionally margin weights in an objective function (derivation in Appendix A.3)

OP 16.

$$\max_{\vec{\alpha}} \quad d\left(\vec{\alpha}\right) = \vec{\alpha} \cdot \left(1 + \vec{\varphi}\right) - \frac{1}{2}\vec{\alpha^T} Q \vec{\alpha} \tag{2.51}$$

subject to

$$0 \leq \vec{\alpha} \leq \vec{C}_{\mathrm{c}} \tag{2.52}$$

where $Q_{ij} = y_i y_j K\left(\vec{x}_i, \vec{x}_j\right)$, for all $i, j \in \{1, \ldots, n\}$.

Incorporating (2.44) to OP 11 also leads to the new kernel in the form of transformation of the input vectors (derivation in Appendix A.4)

$$\vec{x} \to \vec{x} - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i \vec{d}_i \ , \tag{2.53}$$

$$K\left(\vec{x}, \vec{y}\right) = \left(\vec{x} - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i \vec{d}_i\right) \left(\vec{y} - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i \vec{d}_i\right) \ . \tag{2.54}$$

This is a symmetrical kernel. We set the margin weights as

$$\varphi_i = \varphi_{old} - y_i \frac{e}{\sum_{i=1}^{s} s_i} \ . \tag{2.55}$$

We propose also a nonlinear kernel by further kernelization of (2.54), we get

$$K\left(\vec{x}, \vec{y}\right) = K_o\left(\vec{x}, \vec{y}\right) - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i K_o\left(\vec{x}, \vec{d}_i\right) - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i K_o\left(\vec{y}, \vec{d}_i\right) \tag{2.56}$$

$$+ \frac{1}{\left(\sum_{i=1}^{s} s_i\right)^2} \sum_{i=1}^{s} \sum_{j=1}^{s} s_i s_j K_o\left(\vec{d}_i, \vec{d}_j\right) \ . \tag{2.57}$$

This is also a symmetrical kernel. In the final solution, we use the transformation of the input vectors (2.53) only for the training vectors present in the term $\vec{w}_{\mathrm{c}}$, therefore for the solution, we get the following kernel

$$K\left(\vec{x_j}, \vec{x}\right) = K_o\left(\vec{x_j}, \vec{x}\right) - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i K_o\left(\vec{d}_i, \vec{x}\right) \ , \tag{2.58}$$

where $j \in \{1, \ldots, n\}$. Note that this is a kernel between a training vector and $\vec{x}$ used only for defining the solution. For the final solution, we compute the offset as

$$b = \frac{1}{\sum_{i=1}^{s} s_i} \left(e - \sum_{i=1}^{s} s_i \sum_{j=1}^{n} \alpha_j y_{\mathrm{c}}^j K_o\left(\vec{x_j}, \vec{d}_i\right)\right) \tag{2.59}$$

$$+ \frac{1}{\left(\sum_{i=1}^{s} s_i\right)^2} \sum_{i=1}^{s} s_i \sum_{j=1}^{n} \sum_{k=1}^{s} \alpha_j y_{\mathrm{c}}^j s_k K_o\left(\vec{d}_k, \vec{d}_i\right) \ . \tag{2.60}$$

2.9.2 Incorporating the Linear Dependency on Function Values to ε-SVR

We incorporated (2.43) to OP 6. Because ε-SVR is a special case of φ-SVC, and we have derived already the incorporation for φ-SVC, for ε-SVR we set the following weights

$$\varphi_i = y_{\mathrm{c}}^i y_r^i - \varepsilon - 1 - y_{\mathrm{c}}^i \frac{e}{\sum_{i=1}^{s} s_i} \tag{2.61}$$

for $i \in \{1, \ldots, 2n\}$. We also use input space transformation. We use φ-SVC without the offset.

2.9.3 Incorporating Linear Dependency on Function Values to δ-SVR

For completeness we show here the incorporation of linear dependency on function values to δ-SVR. We do not use margin weights in this case. We use kernels developed for δ-SVR, (1.49). Incorporating (2.43) to δ-SVR leads to the SVC optimization problem without the offset OP 4 with the new kernel (derivation in Appendix A.5)

$$K\left(\vec{x},\vec{y}\right)=\left(\vec{x_{\mathrm{red}}}-\frac{1}{\sum_{i=1}^{s}s_i}\sum_{i=1}^{s}s_i\vec{d_{\mathrm{i,red}}}\right)\left(\vec{y_{\mathrm{red}}}-\frac{1}{\sum_{i=1}^{s}s_i}\sum_{i=1}^{s}s_i\vec{d_{\mathrm{i,red}}}\right) \tag{2.62}$$

$$+\left(x_{m+1}-\frac{e}{\sum_{i=1}^{s}s_i}\right)\left(y_{m+1}-\frac{e}{\sum_{i=1}^{s}s_i}\right)\ , \tag{2.63}$$

where $\vec{d_{i,\mathrm{red}}}=\left(d_i^1,\ldots,d_i^m\right)$. This is a symmetrical kernel. We propose also a nonlinear kernel by further kernelization of (2.62), we get

$$K\left(\vec{x},\vec{y}\right)=K_o\left(\vec{x_{\mathrm{red}}},\vec{y_{\mathrm{red}}}\right)-\frac{1}{\sum_{i=1}^{s}s_i}\sum_{i=1}^{s}s_iK_o\left(\vec{d_{\mathrm{i,red}}},\vec{x_{\mathrm{red}}}\right) \tag{2.64}$$

$$-\frac{1}{\sum_{i=1}^{s}s_i}\sum_{i=1}^{s}s_iK_o\left(\vec{d_{\mathrm{i,red}}},\vec{y_{\mathrm{red}}}\right)+\frac{1}{\left(\sum_{i=1}^{s}s_i\right)^2}\sum_{i=1}^{s}\sum_{j=1}^{s}s_is_jK_o\left(\vec{d_{\mathrm{i,red}}},\vec{d_{\mathrm{j,red}}}\right) \tag{2.65}$$

$$+x_{m+1}y_{m+1}-x_{m+1}\frac{e}{\sum_{i=1}^{s}s_i}-y_{m+1}\frac{e}{\sum_{i=1}^{s}s_i}+\frac{e^2}{\left(\sum_{i=1}^{s}s_i\right)^2}\ . \tag{2.66}$$

This is also a symmetrical kernel. In the final solution, we use the kernelization process only for the training vectors present in the term $\vec{w_{\mathrm{c}}}$, therefore for the solution, we get the following kernel

$$K\left(\vec{x_j},\vec{x}\right)=K_o\left(\vec{x_{\mathrm{j,red}}},\vec{x_{\mathrm{red}}}\right)-\frac{1}{\sum_{i=1}^{s}s_i}\sum_{i=1}^{s}s_iK_o\left(\vec{d_{\mathrm{i,red}}},\vec{x_{\mathrm{red}}}\right)+\left(x_j^{m+1}-\frac{e}{\sum_{i=1}^{s}s_i}\right)x_{m+1}\ , \tag{2.67}$$

where $j\in\{1,\ldots,n\}$. Note that this is a kernel between a training vector and $\vec{x}$ used only for defining the solution. For the final solution, we compute the offset as

$$b_{\mathrm{c}}=-\frac{1}{\sum_{i=1}^{s}s_i}e\sum_{j=1}^{2n}\alpha_jy_{\mathrm{c}}^j\left(x_j^{m+1}-\frac{e}{\sum_{i=1}^{s}s_i}\right)-\frac{1}{\sum_{i=1}^{s}s_i}\sum_{i=1}^{s}s_i\sum_{j=1}^{2n}\alpha_jy_{\mathrm{c}}^jK_o\left(\vec{x_{\mathrm{j,red}}},\vec{d_{\mathrm{i,red}}}\right) \tag{2.68}$$

$$+\frac{1}{\sum_{i=1}^{s}s_i}\sum_{i=1}^{s}s_i\sum_{j=1}^{2n}\alpha_jy_{\mathrm{c}}^j\frac{1}{\sum_{i=1}^{s}s_i}\sum_{k=1}^{s}s_kK_o\left(\vec{d_{\mathrm{k,red}}},\vec{d_{\mathrm{i,red}}}\right)\ . \tag{2.69}$$

2.10 Incorporating Inequalities with Function Values

Another example of application for knowledge about the margin of an example is the incorporation of the additional constraints in the form of inequalities with function values for training points for regression case:

$$g\left(\vec{x_i}\right) \geq a_i \tag{2.70}$$

for $i = 1..n$, where a_i are some parameters, g is defined in (1.28). We propose soft incorporation by changing φ_i values. For ε-SVR, it leads to the modification of the ε_u^i value in OP 8:

$$\varepsilon_{\mathrm{u,new}}^i = \min\left(\varepsilon_{\mathrm{u,old}}^i, y_r^i - a_i\right) \tag{2.71}$$

Proof. From (1.37) we have

$$g\left(\vec{x_i}\right) \geq y_r^i - \varepsilon_u^i - \xi_r^i \tag{2.72}$$

It is a soft incorporation so

$$g\left(\vec{x_i}\right) \geq y_r^i - \varepsilon_u^i \tag{2.73}$$

So we should set ε_u^i such as

$$y_r^i - \varepsilon_u^i \geq a_i \tag{2.74}$$

$$\varepsilon_u^i \leq y_r^i - a_i \tag{2.75}$$

so

$$\varepsilon_{\mathrm{u,new}}^i = \min\left(\varepsilon_{\mathrm{u,old}}^i, y_r^i - a_i\right) \tag{2.76}$$

□

For δ-SVR, we set a value of the δ_d^i parameter in OP 10:

$$\delta_{\mathrm{d,new}}^i = \min\left(\delta_{\mathrm{d,old}}^i, y_r^i - a_i\right) \quad . \tag{2.77}$$

Proof. After shifting data the barrier for the function is the shifted point. It will be more restrict barrier, because it is a barrier for the margin boundary. It implies the barrier for the function. We have

$$y_r^i - \delta_d^i \geq a_i \tag{2.78}$$

$$\delta_d^i \leq y_r^i - a_i \quad . \tag{2.79}$$

So we modify the shifting parameter δ_d^i as

$$\delta_{\mathrm{d,new}}^i = \min\left(\delta_{\mathrm{d,old}}^i, y_r^i - a_i\right) \quad . \tag{2.80}$$

□

Although we do not modify directly φ_i weights, they are modified indirectly for ε-SVR, because changing ε_i leads to changing φ_i. Changing δ_i can be interpreted as changing ε_i.

Similar incorporation scheme exists for

$$g\left(\vec{x_i}\right) \leq a_i \quad . \tag{2.81}$$

For ε-SVR, it leads to the modification of the ε_d^i value in OP 8:

$$\varepsilon_{\mathrm{d,new}}^i = \min\left(\varepsilon_{\mathrm{d,old}}^i, a_i - y_r^i\right) \tag{2.82}$$

Proof. From (1.38) we have

$$g\left(\vec{x_i}\right) \leq \varepsilon_d^i + y_r^i + \xi_r^i \tag{2.83}$$

It is a soft incorporation so

$$g\left(\vec{x_i}\right) \leq \varepsilon_d^i + y_r^i \tag{2.84}$$

So we should set ε_d^i such as

$$\varepsilon_d^i + y_r^i \leq a_i \tag{2.85}$$

$$\varepsilon_d^i \leq a_i - y_r^i \tag{2.86}$$

so

$$\varepsilon_{\mathrm{d,new}}^i = \min\left(\varepsilon_{\mathrm{d,old}}^i, a_i - y_r^i\right) \tag{2.87}$$

□

For δ-SVR, we set a value of the δ_u^i parameter in OP 10:

$$\delta_{\mathrm{u,new}}^i = \min\left(\delta_{\mathrm{u,old}}^i, a_i - y_r^i\right) \tag{2.88}$$

Proof. We have

$$y_r^i + \delta_u^i \leq a_i \tag{2.89}$$

$$\delta_u^i \leq a_i - y_r^i \tag{2.90}$$

So we modify the shifting parameter δ_u^i as

$$\delta_{\mathrm{u,new}}^i = \min\left(\delta_{\mathrm{u,old}}^i, a_i - y_r^i\right) \quad . \tag{2.91}$$

□

We can also increase values of proper C_i parameters to improve fulfillment of the constraints.

Another incorporation type is of the same form as (2.70) but defined for a new point $\vec{x_{n+1}}$ without defined y_r^{n+1}

$$g\left(\vec{x_{n+1}}\right) \geq a_{n+1} \quad , \tag{2.92}$$

where a_{n+1} is a parameter, g is defined in (1.28). We propose soft incorporation. We will use a special version of ε-SVR, where we allow presence of only one constraint either (1.37) or (1.38). So for (2.92) we will have only one constraint that is (1.37). We set

$$y_r^{n+1} = a_{n+1} \tag{2.93}$$

After substituting above to (2.71) we get

$$\varepsilon_{\text{u,new}}^{n+1} = \min\left(\varepsilon_{\text{u,old}}^{n+1}, 0\right) \tag{2.94}$$

One constraint in the formulation means that while transforming to the φ-SVC only one classification point will be created, in this case with $y_c^{n+1} = 1$.

For δ-SVR, we propose also soft incorporation. We will use a special version of δ-SVR, where we allow presence of points which are not duplicated but shifted either up or down. We define such point with $y_c^{n+1} = -1$ and we set

$$y_r^{n+1} = a_{n+1} \tag{2.95}$$

and we shift it down by

$$\delta_{\text{d,new}}^{n+1} = \min\left(\delta_{\text{d,old}}^{n+1}, 0\right) \quad . \tag{2.96}$$

For

$$g\left(\vec{x_{n+1}}\right) \leq a_{n+1} \quad , \tag{2.97}$$

For (2.97) we will have only one constraint that is (1.38). We set

$$y_r^{n+1} = a_{n+1} \tag{2.98}$$

After substituting above to (2.82), we get

$$\varepsilon_{\text{d,new}}^{n+1} = \min\left(\varepsilon_{\text{d,old}}^{n+1}, 0\right) \tag{2.99}$$

One constraint in the formulation means that while transforming to the φ-SVC only one classification point will be created, in this case with $y_c^{n+1} = -1$.

For δ-SVR, we define a point with $y_c^{n+1} = 1$ and we set

$$y_r^{n+1} = a_{n+1} \tag{2.100}$$

and we shift it up by

$$\delta_{\text{u,new}}^{n+1} = \min\left(\delta_{\text{u,old}}^{n+1}, 0\right) \quad . \tag{2.101}$$

2.11 Reduce a Model with φ-SVC

Various methods for reducing the complexity of the output model were widely investigated, [4]. In particular, the reduction by removing support vectors was also analyzed in [3] for regression problems.

Sparse models have an advantage of faster post-processing such as testing new examples. For highly nonseparable data with a linear decision boundary, SVC has multiple support vectors. The general idea of constructing even more sparse solutions than SVC is to find a solution spanned on the given number of support vectors as close to the original SVC solution as possible. The most representative method for this idea is presented in [16]. Another example for this group are reduced support vector machines (RSVM), which randomly selects support vectors to a reduced set, [2, 7]. The alternative approach is to replace SVM with another training method that is designed for returning sparse solutions. The most representative method for this group is a greedy approach that adds basis functions to the solution until no progress in optimizing a cost function is made, [4].

Our proposed approach, [9, 11], is conceptually closer to the first idea. After SVM is trained, we remove randomly selected data vectors, and run again SVM on a reduced training set with incorporated prior knowledge about the original solution, Fig. 2.7, Fig. 2.8, Fig. 2.9. A concept of randomly removing support vectors was presented in [3]. For highly small number of support vectors removing some of them would definitely lead to decreasing generalization performance. The proposed method generates reduced models from the original full model with incorporated knowledge about the margin of an example. The procedure of generating knowledge about the margin of an example is as following. First, φ_i weights are automatically generated from an existing solution as

$$\varphi_i = y_i h^* \left(\vec{x_i}\right) - 1 \tag{2.102}$$

for all training examples. For ε-SVR, instead of modifying φ_i weights, we can alternatively modify ε_u^i and ε_d^i weights in OP 8:

$$\varepsilon_u^i = y_r^i - g^* \left(\vec{x_i}\right) \quad ,$$

$$\varepsilon_d^i = g^* \left(\vec{x_i}\right) - y_r^i \quad .$$

After that, a reduced model is generated by removing a bunch of data vectors – randomly selected data vectors, with maximal removal ratio of $p\%$ off all training vectors, where p is a configurable parameter. Not that in this set, there are also support vectors that will be removed. Finally, we run φ-SVC with a reduced data set.

2.12 Generation of Prior Knowledge

When prior knowledge comes from the external source, we can directly use it for comparison of accuracy of the models. Otherwise, we have to generate prior knowledge from the available data. Mangasarian et al. [6] divide a data set to two sets (about 20% and

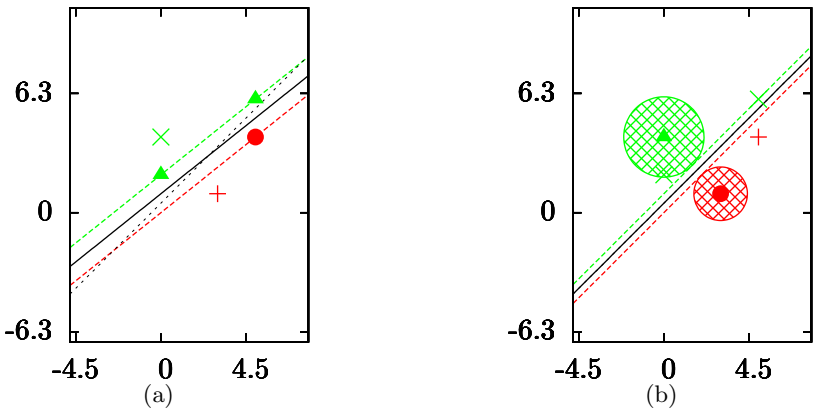


Figure 2.7: The idea of reduced SVC. Points - examples after reducing, triangles and circles - support vectors, solid lines - solutions, dotted lines - original solutions before reduction, dashed lines - margin boundaries. (b) Circles filled with grid pattern - dynamic hyperspheres

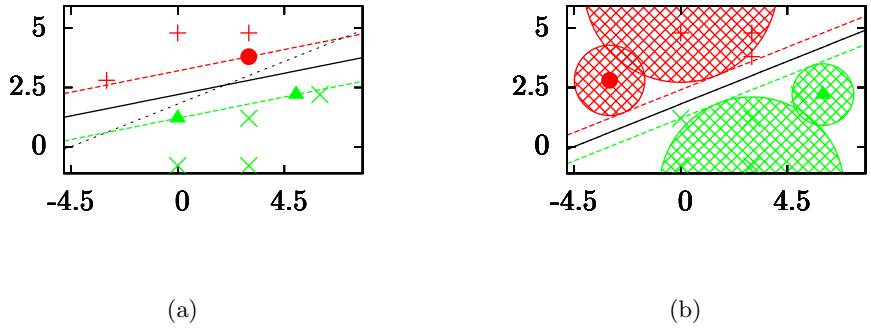
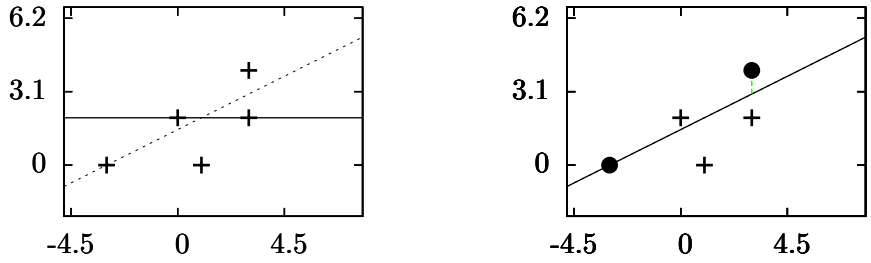


Figure 2.8: The idea of reduced δ-SVR. Points - examples after reducing, triangles and circles - support vectors, solid lines - solutions, dotted lines - original solutions before reduction, dashed lines - margin boundaries. (b) Circles filled with grid pattern - dynamic hyperspheres

80%), and they generate manually prior knowledge from the first set, and use it in the other. We proposed a slightly modified procedure. We generate knowledge from the whole training data set, and then we remove some of data vectors. The procedure is as follows:

1. Find a solution of SVM problem without prior knowledge.
2. Extract prior knowledge from the solution.
3. Remove randomly p% of input data.
4. Find the solution with and without prior knowledge on a reduced data set.



(a) (b)

Figure 2.9: The idea of reduced ε-SVR. Points - examples after reducing, filled circles - support vectors, solid lines - solutions, dotted lines - original solutions before reduction. (b) The dashed lines must cross the solution

2.13 Experiments

In the first experiment, we compare the performance of the SVM without and with knowledge about the margin of an example depends on the removal ratio p. For comparison purposes, a reduced model is the same for both methods. We use our implementation of all methods. In the second experiment, we compare the performance of both methods related to various number of support vectors.

For all data sets, every feature is scaled linearly to $[0, 1]$ including an output. For variable parameters like the C, σ for the RBF kernel, δ for δ-SVR, and ε for ε-SVR, we use a grid search method for finding best values. The number of values searched by the grid method is a trade-off between an accuracy and a speed of simulations. Note that for a particular data set it is possible to use more accurate grid searches than for massive tests with multiple number of simulations.

2.13.1 First Experiment

For synthetic data, we compare both methods on data generated from particular functions with added Gaussian noise for output values. For $p = 70$, the SVC with knowledge about a margin has smaller generalization error for every test except one, Table 2.1. The testing performance gain varies from 0% to 3%. The number of support vectors is slightly smaller for the method with knowledge about a margin. For the regression case, for $p = 70$, the ε-SVR with knowledge about a margin has smaller generalization error for every test, Table 2.2. The testing performance gain varies from 0% to 50%. The number of support vectors is bigger for the method with knowledge about a margin.

The real world data sets were taken from the LibSVM site, [1, 5], except stock price data. The stock price data consist of monthly prices of the Dow Jones Industrial Average (DJIA) index from 1898 up to 2010. We generated the training set as follows: for every

Table 2.1: Performance of φ-SVC for reduced models for synthetic data. Column descriptions: *id* – id of the test, *a function* – a function used for generating data $y_1 = \sum_{i=1}^{\dim-1} x_i$, $y_4, y_5 = \left(\sum_{i=1}^{\dim-1} x_i\right)^{\text{kerP}}$, $y_6 = 0.5 \sum_{i=1}^{\dim-1} \sin 10x_i + 0.5$, *ker* – a kernel with a kernel parameter (for a polynomial kernel it is a dimension, for the RBF kernel it is σ), *idRef* – a reference to the test, *te12M* – a percent average difference in correctly classified examples for testing data, if greater than 0 than a method with knowledge about a margin is better, *s1* – the average number of support vectors for a method without knowledge about a margin, *s2* – the average number of support vectors for a method with knowledge about a margin

(a)

id	function	ker
0	y_1	denseLinear 0.0
1	$y_2 = 3y_1$	denseLinear 0.0
2	$y_3 = 1/3y_1$	denseLinear 0.0
3	y_4	densePolynomial 5.0
4	y_5	denseRBF 0.25
5	y_6	denseRBF 0.25

(b)

idRef	te12M	s1	s2
0	2.4	23	19
1	3.22	23	19
2	−0.63	24	21
3	1.19	23	16
4	0.92	25	20
5	0.43	26	25

Table 2.2: Performance of φ-SVC for reduced models for synthetic data, for regression. Column descriptions: *id* – id of the test, *a function* – a function used for generating data $y_1 = \sum_{i=1}^{\dim-1} x_i$, $y_2 = \left(\sum_{i=1}^{\dim-1} x_i\right)^{\text{kerP}}$, $y_3 = 0.5 \sum_{i=1}^{\dim-1} \sin 10x_i + 0.5$, *ker* – a kernel with a kernel parameter (for a polynomial kernel it is a dimension, for the RBF kernel it is σ), *idRef* – a reference to the test, *te12M* – a percent average difference in MSE for testing data, if greater than 0 than a method with knowledge about a margin is better, *s1* – the average number of support vectors for a method without knowledge about a margin, *s2* – the average number of support vectors for a method with knowledge about a margin

(a)

id	function	ker
0	y_1	denseLinear 0.0
1	y_2	densePolynomial 5.0
2	y_3	denseRBF 0.25

(b)

idRef	te12M	s1	s2
0	50.05	11	20
1	0.98	42	52
2	4.6	41	54

month the output value is a growth/fall comparing to the next month. Every feature i is a percent price change between the month and the i-th previous month. In every simulation, training data are randomly chosen, the remaining examples become test data. We can see that for variable p, the SVM with knowledge about a margin has generally smaller generalization error than without, Fig. 2.10, Fig. 2.11, Fig. 2.12, Fig. 2.13. For

Table 2.3: Performance of φ-SVC for reduced models for real world data. Column descriptions: *id* – id of the test, *dn* – a data set, *ker* – a kernel with a kernel parameter (for a polynomial kernel it is a dimension, for the RBF kernel it is σ), *idRef* – a reference to the test, *te12M* – a percent average difference in correctly classified examples for testing data, if greater than 0 than a method with knowledge about a margin is better, *s1* – the average number of support vectors for a method without knowledge about a margin, *s2* – the average number of support vectors for a method with knowledge about a margin

(a)

id	dn	ker
0	a1aAll	denseLinear 0.0
1	a1aAll	densePolynomial 5.0
2	a1aAll	denseRBF 0.00813
3	breast-cancer	denseLinear 0.0
4	breast-cancer	densePolynomial 3.0
5	breast-cancer	denseRBF 0.1
6	diabetes	denseLinear 0.0
7	diabetes	densePolynomial 3.0
8	diabetes	denseRBF 0.125
9	djia	denseLinear 0.0
10	djia	densePolynomial 5.0
11	djia	denseRBF 0.08333

(b)

idRef	te12M	s1	s2
0	15.47	16	16
1	3.79	11	25
2	0.0	27	27
3	14.26	7	8
4	24.71	5	6
5	3.61	24	25
6	9.41	20	17
7	7.16	16	15
8	0.42	26	26
9	0.67	23	16
10	1.26	22	16
11	1.52	29	28

$p = 70$, the SVC with knowledge about a margin has smaller generalization error for all data sets, with smaller or equal number of support vectors in 8 out of 12 tests, Table 2.3. The testing performance gain varies from 0% to 25%. For a regression case, for $p = 70$, the ε-SVR with knowledge about a margin has smaller generalization error for every test, Table 2.4. The testing performance gain varies from 0% to 34%. The number of support vectors is bigger for the method with knowledge about a margin, for all regression tests.

We achieve smaller generalization error for the method with knowledge about a margin. We confirmed that knowledge about a margin in deed is able to preserve information about the original solution.

2.13.2 Second experiment

In the second experiment, we compare performance of the method without and with knowledge about a margin, but related to the number of support vectors. We performed tests for classification and regression. We can notice that the method with knowledge about a margin tends to achieve smaller generalization error for the similar number of support vectors, although for the RBF kernel the results are similar (Fig. 2.14, Fig. 2.15, Fig. 2.16, Fig. 2.17).

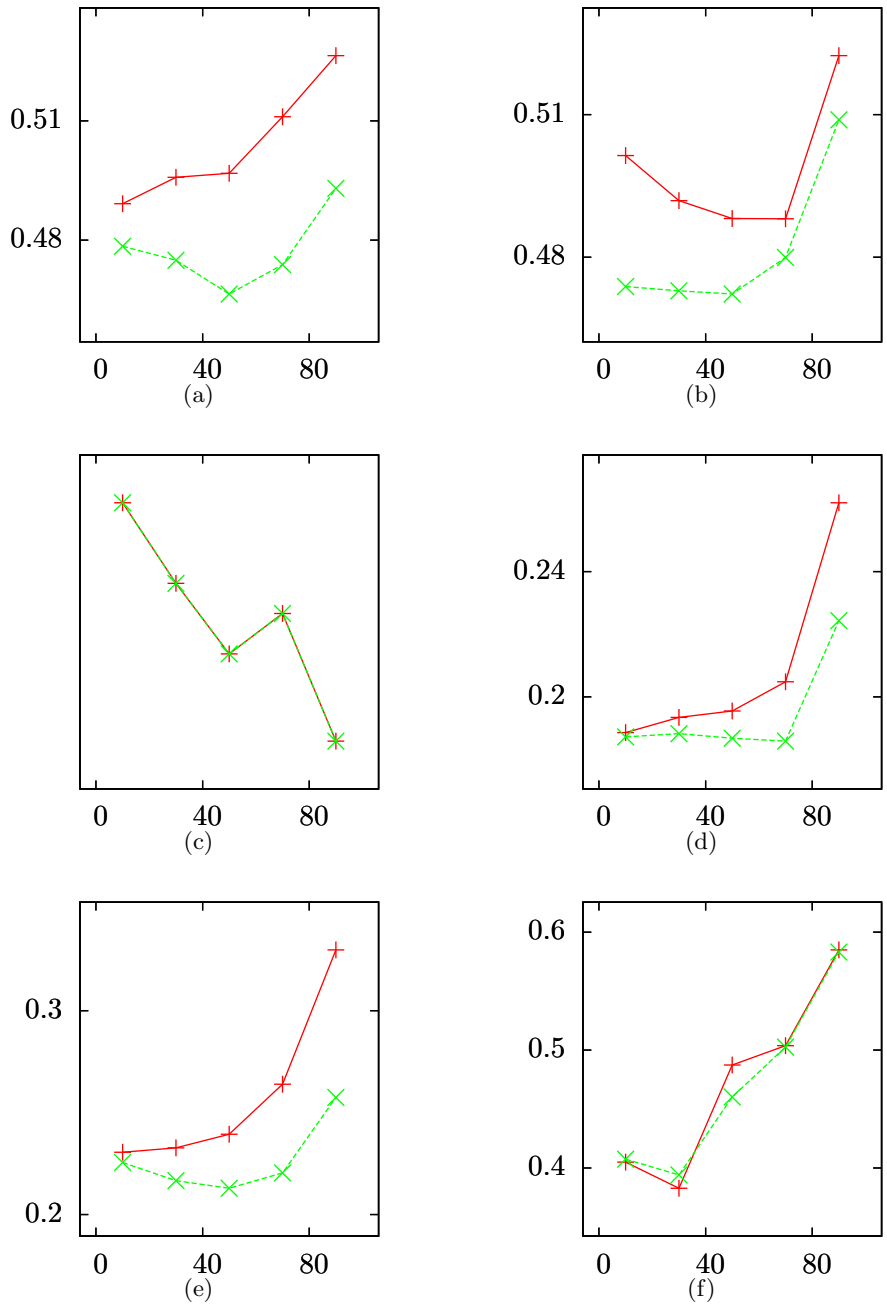


Figure 2.10: Comparison of two methods of removing support vectors for the test cases with ids 0-5 from Table 2.3. The x axis - the percent of removed data, y axis - a percent difference in misclassified testing examples, the line with '+' points - a random removing method, the line with 'x' points - proposed removing method with knowledge about a margin

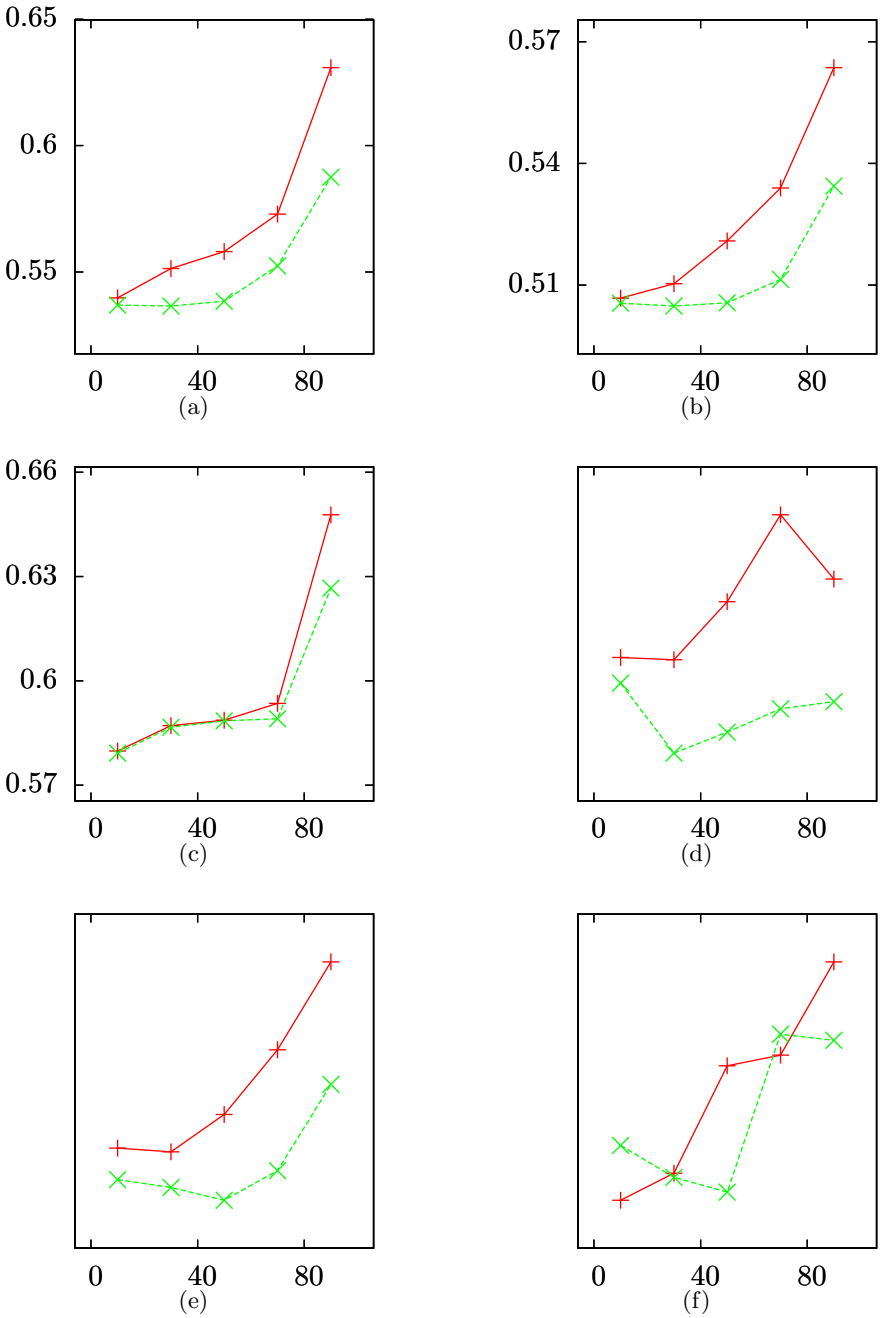


Figure 2.11: Comparison of two methods of removing support vectors for the test cases with ids 6-11 from Table 2.3, cont. The x axis - the percent of removed data, y axis - a percent difference in misclassified testing examples, the line with '+' points - a random removing method, the line with 'x' points - proposed removing method with knowledge about a margin

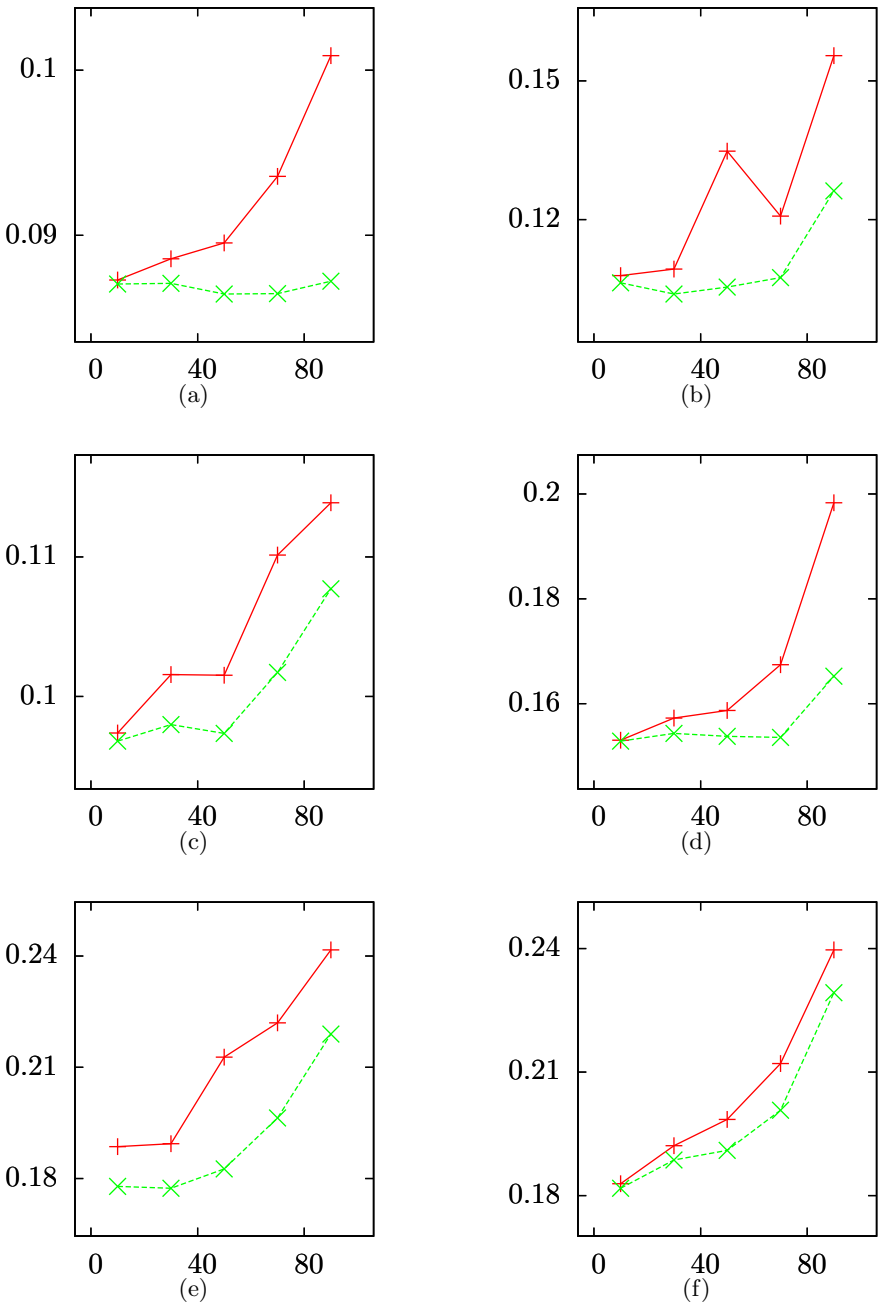


Figure 2.12: Comparison of two methods of removing support vectors for the test cases with ids 0-5 from Table 2.4. The x axis - the percent of removed data, y axis - a percent difference in MSE for testing data, the line with '+' points - a random removing method, the line with 'x' points - proposed removing method with knowledge about a margin

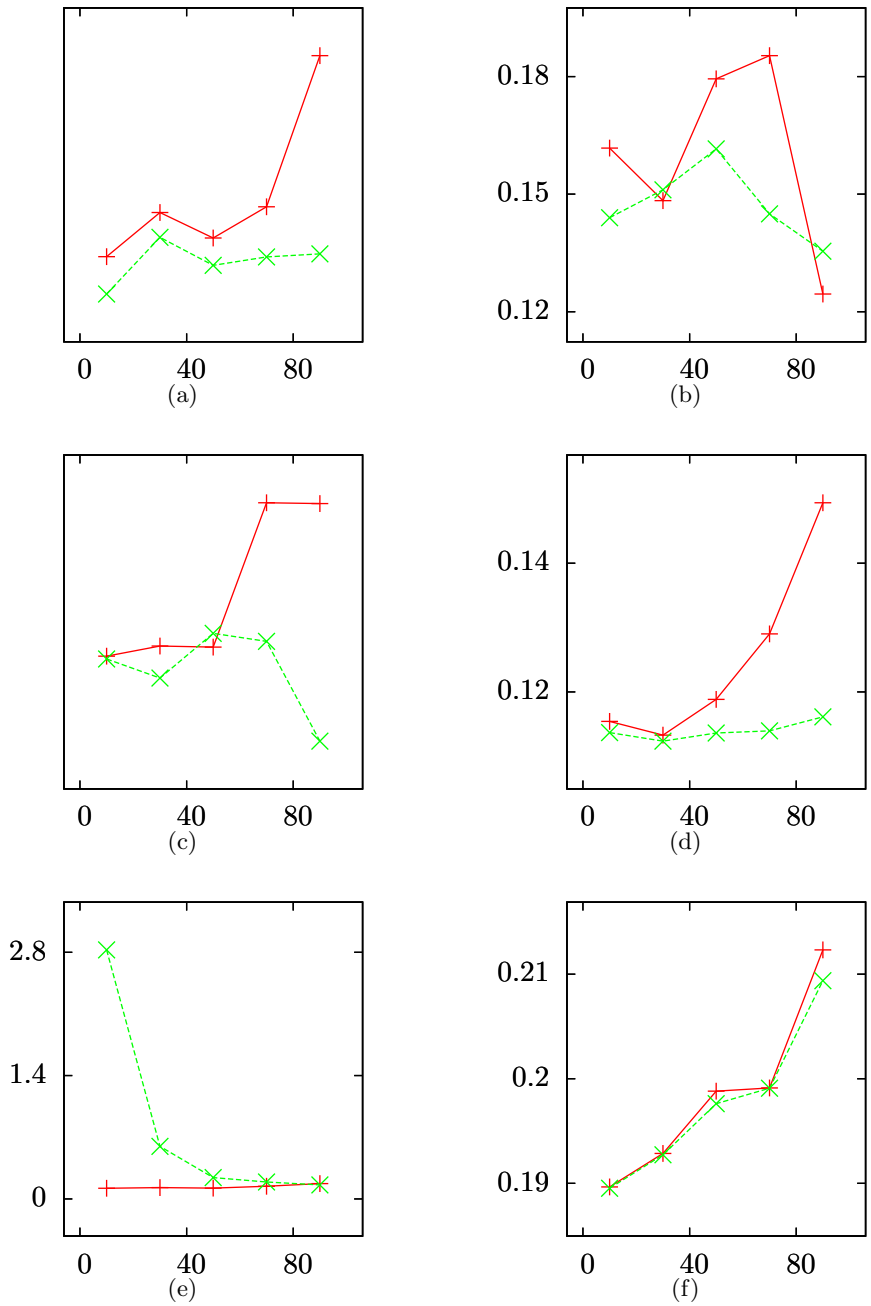


Figure 2.13: Comparison of two methods of removing support vectors for the test cases with ids 6-11 from Table 2.4, cont. The x axis - the percent of removed data, y axis - a percent difference in MSE for testing data, the line with '+' points - a random removing method, the line with 'x' points - proposed removing method with knowledge about a margin

Table 2.4: Performance of φ-SVC for reduced models for real world data, for regression. Column descriptions: *id* – id of the test, *dn* – a data set, *ker* – a kernel with a kernel parameter (for a polynomial kernel it is a dimension, for the RBF kernel it is σ), *idRef* – a reference to the test, *te12M* – a percent average difference in MSE for testing data, if greater than 0 than a method with knowledge about a margin is better, *s1* – the average number of support vectors for a method without knowledge about a margin, *s2* – the average number of support vectors for a method with knowledge about a margin

(a)

id	dn	ker
0	abalone	denseLinear 0.0
1	abalone	densePolynomial 5.0
2	abalone	denseRBF 0.125
3	cadata	denseLinear 0.0
4	cadata	densePolynomial 5.0
5	cadata	denseRBF 0.125
6	djia	denseLinear 0.0
7	djia	densePolynomial 5.0
8	djia	denseRBF 0.1
9	housing	denseLinear 0.0
10	housing	densePolynomial 5.0
11	housing	denseRBF 0.077

(b)

idRef	te12M	s1	s2
0	11.39	12	30
1	34.86	14	48
2	11.11	18	53
3	14.08	25	47
4	16.38	26	52
5	11.41	34	53
6	5.81	9	26
7	34.16	13	48
8	0.2	8	50
9	19.47	17	29
10	18.39	19	53
11	1.33	30	53

2.14 Summary

In this report, we analyzed the possibility to preserve knowledge about the original solution by using margin weights while creating reduced models. We also used knowledge about the margin of an example for incorporating the additional nonlinear constraint to the problem. We also showed that the method with knowledge about the margin of an example can achieve smaller generalization error for similar number of support vectors, for reduced models. A potential list of applications for margin weights is much broader. In future research, we plan to investigate possibility to create knowledge about a margin by experts and to apply it for time series data. Moreover, we plan to use it for combining SVM classifiers with each other in order to decrease generalization error.

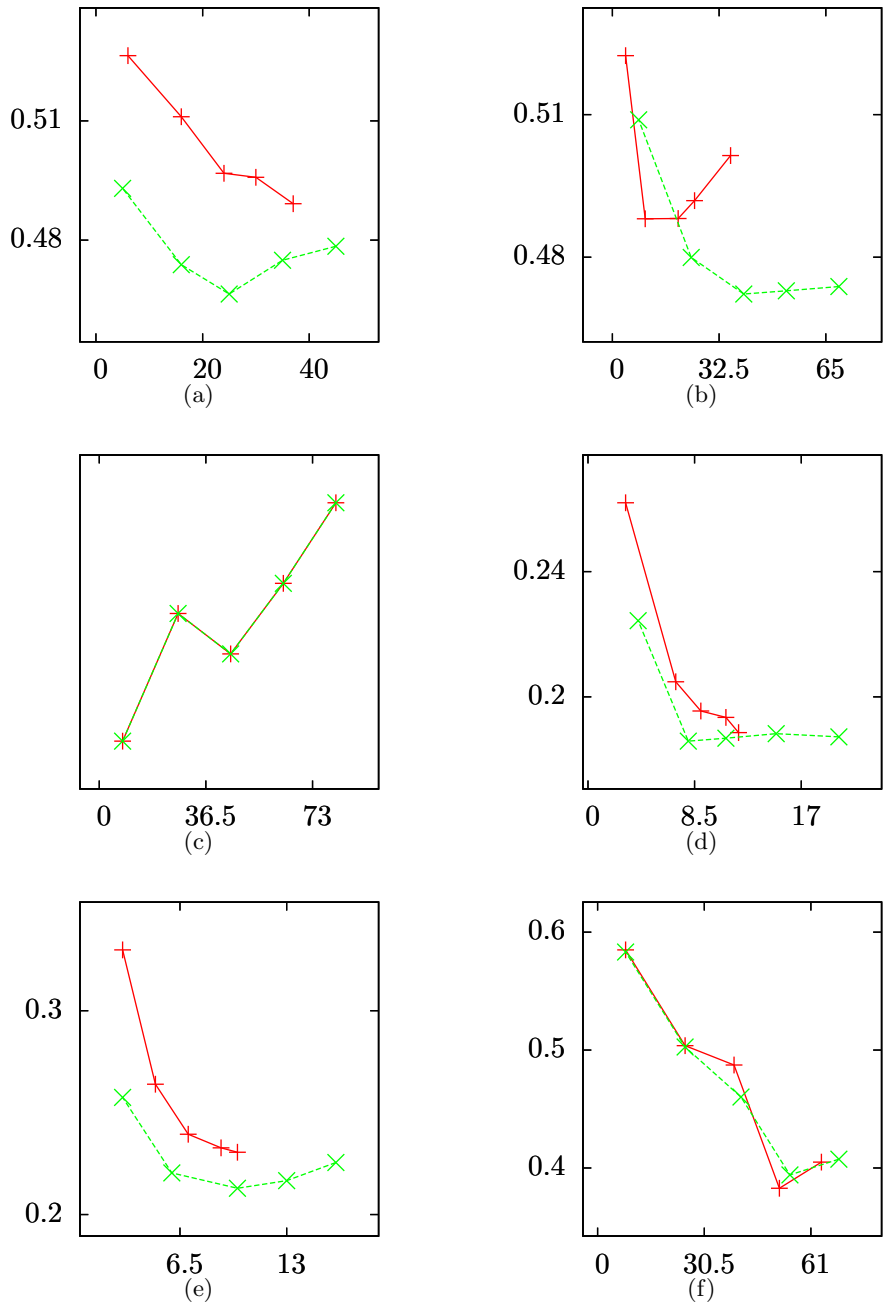


Figure 2.14: Comparison of two methods of removing support vectors for the test cases with ids 0-5 from Table 2.3. The x axis - the number of support vectors, y axis - a percent difference in misclassified testing examples, the line with '+' points - a random removing method, the line with 'x' points - proposed removing method with knowledge about a margin

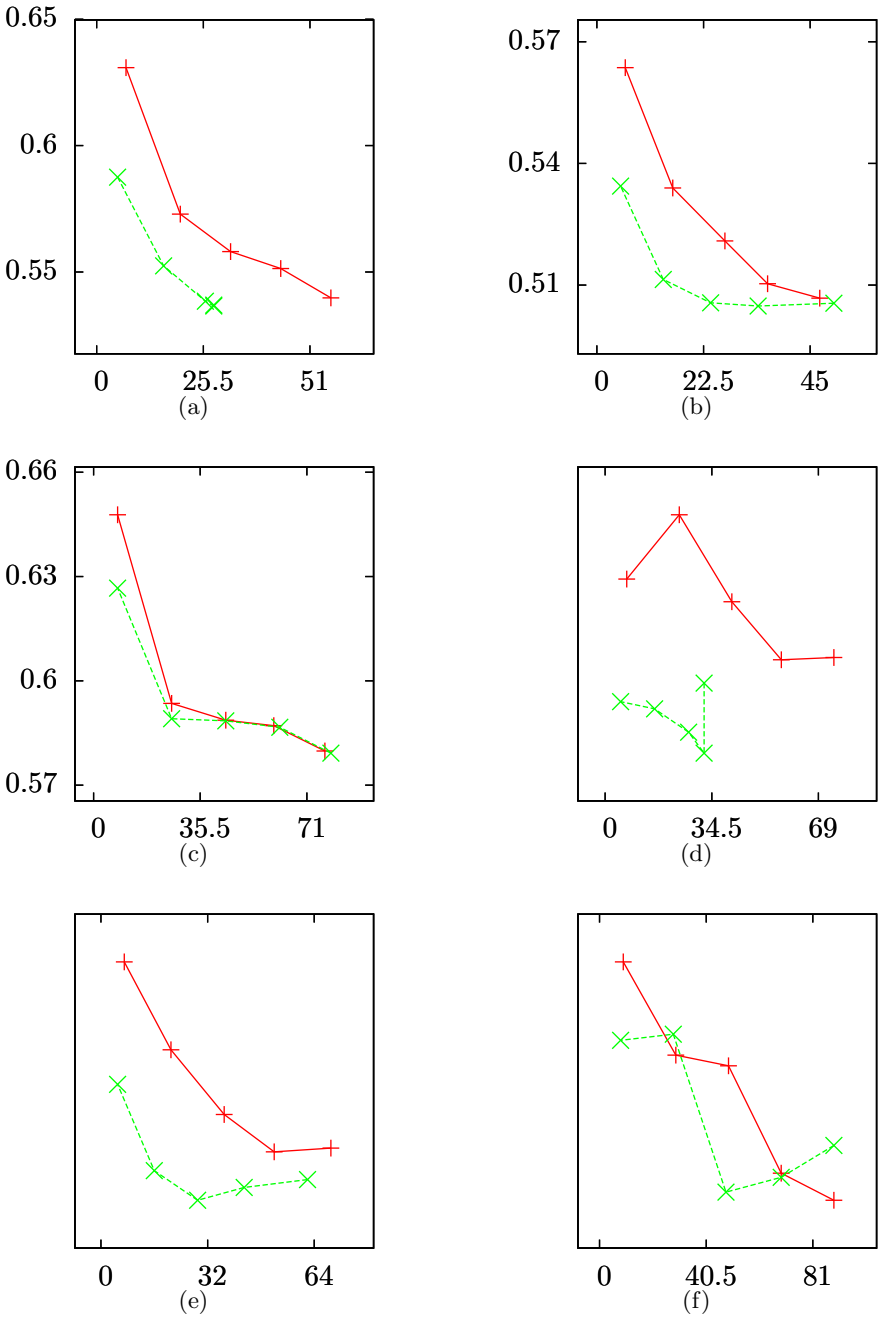


Figure 2.15: Comparison of two methods of removing support vectors for the test cases with ids 6-11 from Table 2.3, cont. The x axis - the number of support vectors, y axis - a percent difference in misclassified testing examples, the line with '+' points - a random removing method, the line with 'x' points - proposed removing method with knowledge about a margin

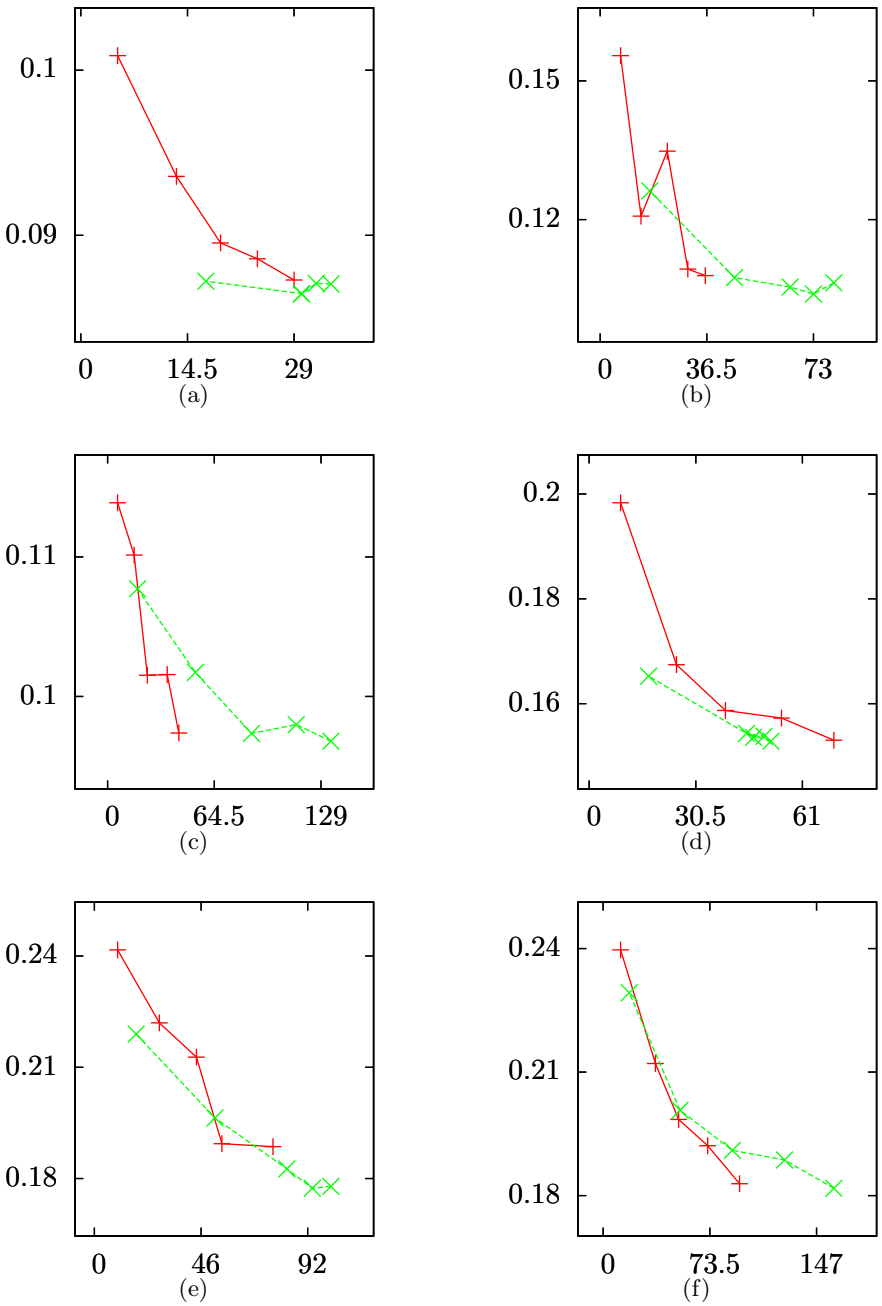


Figure 2.16: Comparison of two methods of removing support vectors for the test cases with ids 0-5 from Table 2.4. The x axis - the number of support vectors, y axis - a percent difference in MSE for testing data, the line with '+' points - a random removing method, the line with 'x' points - proposed removing method with knowledge about a margin

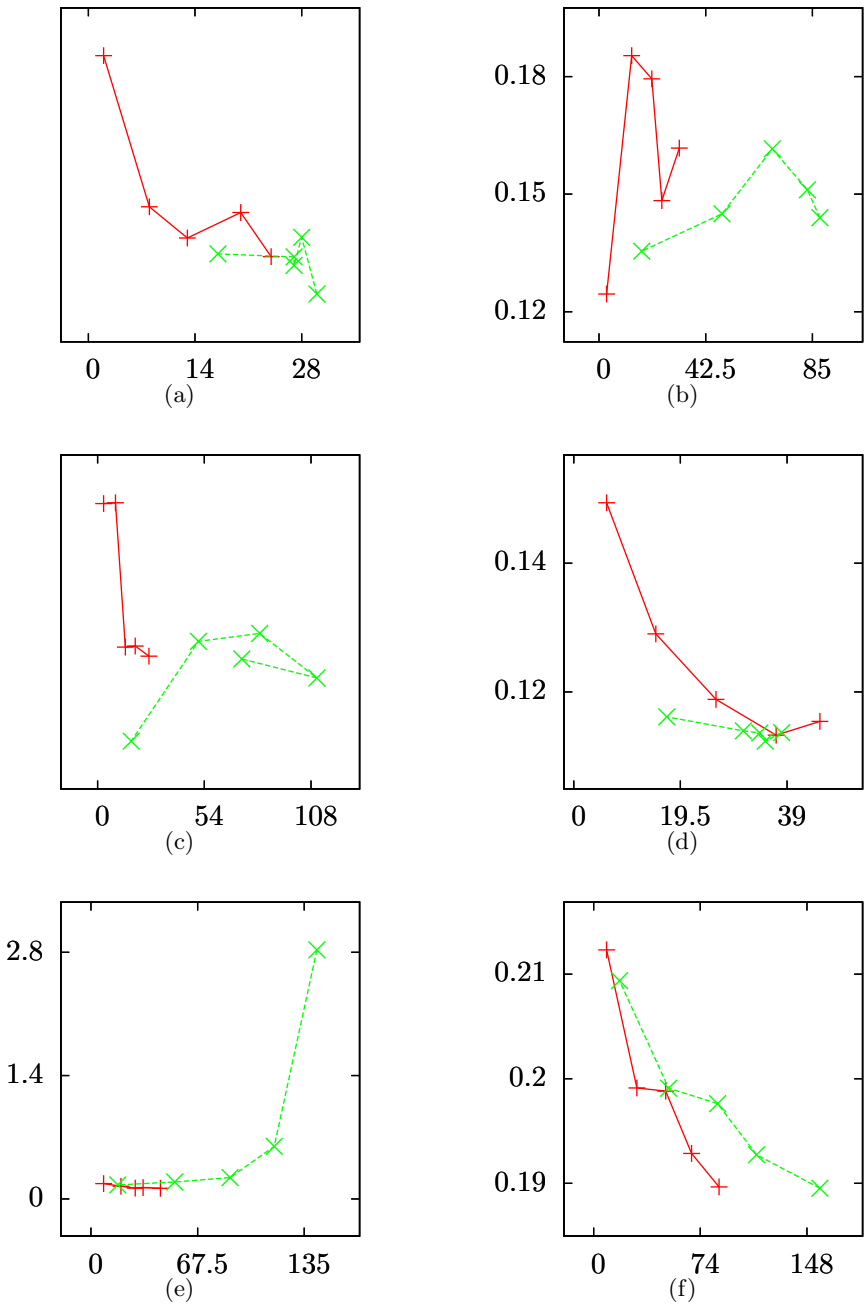


Figure 2.17: Comparison of two methods of removing support vectors for the test cases with ids 6-11 from Table 2.4, cont. The x axis - the number of support vectors, y axis - a percent difference in MSE for testing data, the line with '+' points - a random removing method, the line with 'x' points - proposed removing method with knowledge about a margin

Appendix A

A.1 Derivation of the Dual Form of OP 11

OP 17.

$$\max_{\vec{\alpha},\vec{r}} \quad d\left(\vec{\alpha},\vec{r}\right) \tag{A.1}$$

where

$$\begin{aligned} d\left(\vec{\alpha},\vec{r}\right) &= \min_{\vec{w},b,\vec{\xi}} t\left(\vec{w},b,\vec{\alpha},\vec{\xi},\vec{r}\right) \\ t\left(\vec{w},b,\vec{\alpha},\vec{\xi},\vec{r}\right) &= \frac{1}{2}\left\|\vec{w}\right\|^2 + \sum_{i=1}^{n} C_i\xi_i - \\ &\quad - \sum_{i=1}^{n} \alpha_i\left(y_{\mathrm{c}}^i h\left(\vec{x_i}\right) - 1 + \xi_i - \varphi_i\right) - \sum_{i=1}^{n} r_i\xi_i \end{aligned}$$

subject to

$$\begin{aligned} \alpha_i &\geq 0 \\ r_i &\geq 0 \end{aligned}$$

for $i \in \{1,\ldots,n\}$.

A partial derivative with respect to w_i is

$$\frac{\partial t\left(\vec{w},b,\vec{\alpha},\vec{\xi},\vec{r}\right)}{\partial w_i} = w_i - \sum_{j=1}^{n} \alpha_j y_{\mathrm{c}}^j x_{ji} = 0 \tag{A.2}$$

for $i \in \{1,\ldots,m\}$. A partial derivative with respect to b is

$$\frac{\partial t\left(\vec{w},b,\vec{\alpha},\vec{\xi},\vec{r}\right)}{\partial b} = \sum_{i=1}^{n} \alpha_i y_{\mathrm{c}}^i = 0 \ . \tag{A.3}$$

A partial derivative with respect to ξ_i is

$$\frac{\partial t\left(\vec{w},b,\vec{\alpha},\vec{\xi},\vec{r}\right)}{\partial \xi_i} = C_i - r_i - \alpha_i = 0 \ . \tag{A.4}$$

After substitution of above equations to $d(\vec{\alpha}, \vec{r})$ we get

$$\begin{aligned} d(\vec{\alpha}, \vec{r}) &= \tfrac{1}{2} \sum_{i=1}^{m} \left(\sum_{j=1}^{n} \alpha_j y_{\mathrm{c}}^j x_{ji} \right) \left(\sum_{k=1}^{n} \alpha_k y_{\mathrm{c}}^k x_{ki} \right) \\ &- \sum_{i=1}^{n} \alpha_i y_{\mathrm{c}}^i \left(\sum_{j=1}^{m} w_j x_{ij} + b \right) + \sum_{i=1}^{n} \alpha_i (1 + \varphi_i) + C_i \sum_{i=1}^{n} \xi_i \\ &- \sum_{i=1}^{n} \alpha_i \xi_i - \sum_{i=1}^{n} r_i \xi_i \end{aligned} \tag{A.5}$$

$$\begin{aligned} d(\vec{\alpha}, \vec{r}) &= \tfrac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{n} \alpha_k \alpha_j y_{\mathrm{c}}^k y_{\mathrm{c}}^j x_{ki} x_{ji} - \sum_{i=1}^{n} \alpha_i y_{\mathrm{c}}^i \sum_{j=1}^{m} w_j x_{ij} \\ &- b \sum_{i=1}^{n} \alpha_i y_{\mathrm{c}}^i + \sum_{i=1}^{n} \alpha_i (1 + \varphi_i) \end{aligned} \tag{A.6}$$

$$\begin{aligned} d(\vec{\alpha}, \vec{r}) &= \tfrac{1}{2} \sum_{j=1}^{n} \sum_{k=1}^{n} \alpha_k \alpha_j y_{\mathrm{c}}^k y_{\mathrm{c}}^j \sum_{i=1}^{m} x_{ji} x_{ki} \\ &- \sum_{i=1}^{n} \alpha_i y_{\mathrm{c}}^i \sum_{j=1}^{m} x_{ij} \sum_{k=1}^{n} \alpha_k y_{\mathrm{c}}^k x_{kj} + \sum_{i=1}^{n} \alpha_i (1 + \varphi_i) \end{aligned} \tag{A.7}$$

$$\begin{aligned} d(\vec{\alpha}, \vec{r}) &= \tfrac{1}{2} \sum_{j=1}^{n} \sum_{k=1}^{n} \alpha_k \alpha_j y_{\mathrm{c}}^k y_{\mathrm{c}}^j \sum_{i=1}^{m} x_{ji} x_{ki} \\ &- \sum_{i=1}^{n} \sum_{k=1}^{n} \alpha_k \alpha_i y_{\mathrm{c}}^k y_{\mathrm{c}}^i \sum_{j=1}^{m} x_{ij} x_{kj} + \sum_{i=1}^{n} \alpha_i (1 + \varphi_i) \end{aligned} \tag{A.8}$$

$$d(\vec{\alpha}, \vec{r}) = -\frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{n} \alpha_k \alpha_i y_{\mathrm{c}}^k y_{\mathrm{c}}^i \sum_{j=1}^{m} x_{ij} x_{kj} + \sum_{i=1}^{n} \alpha_i (1 + \varphi_i) \ . \tag{A.9}$$

The dual form is

OP 18.

$$\max_{\vec{\alpha}, \vec{r}} \quad d(\vec{\alpha}, \vec{r}) = \sum_{i=1}^{n} \alpha_i (1 + \varphi_i) - \frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{n} \alpha_k \alpha_i y_{\mathrm{c}}^k y_{\mathrm{c}}^i \sum_{j=1}^{m} x_{ij} x_{kj} \tag{A.10}$$

subject to

$$\sum_{i=1}^{n} \alpha_i y_{\mathrm{c}}^i = 0 \tag{A.11}$$

$$C_i = r_i + \alpha_i \tag{A.12}$$

$$\alpha_i \geq 0 \tag{A.13}$$

$$r_i \geq 0 \tag{A.14}$$

for $i \in \{1, \ldots, n\}$.

A.2 Derivation of ε-SVR Reformulation as φ-SVC

We present derivation of ε-SVR in the form OP 8 as φ-SVC

OP 19.

$$\min_{\vec{w_r}, b_r, \vec{\xi_r}, \vec{\xi_r^*}} \quad f\left(\vec{w_r}, b_r, \vec{\xi_r}, \vec{\xi_r^*}\right) = \frac{1}{2}\left\|\vec{w_r}\right\|^2 + C_r \sum_{i=1}^{n} \left(\xi_r^i + \xi_r^{*i}\right) \tag{A.15}$$

subject to

$$y_r^i - g\left(\vec{x_i}\right) \leq \varepsilon_u^i + \xi_r^i \tag{A.16}$$

$$g\left(\vec{x_i}\right) - y_r^i \leq \varepsilon_d^i + \xi_r^{i*} \tag{A.17}$$

$$\vec{\xi_r} \geq 0 \tag{A.18}$$

$$\vec{\xi_r^*} \geq 0 \tag{A.19}$$

for $i \in \{1, \ldots, n\}$, where

$$g\left(\vec{x_i}\right) = \vec{w_r} \cdot \vec{x_i} + b_r \ . \tag{A.20}$$

OP 20.

$$\min_{\vec{w_r}, b_r, \vec{\xi_r}, \vec{\xi_r^*}} \quad f\left(\vec{w_r}, b_r, \vec{\xi_r}, \vec{\xi_r^*}\right) = \frac{1}{2}\left\|\vec{w_r}\right\|^2 + C_r \sum_{i=1}^{2n} \xi_r^i \tag{A.21}$$

subject to

$$g\left(\vec{x_i}\right) \geq y_r^i - \varepsilon_u^i - \xi_r^i \tag{A.22}$$

for $i \in \{1, \ldots, n\}$,

$$g\left(\vec{x_i}\right) \leq \varepsilon_d^i + y_r^i + \xi_r^i \tag{A.23}$$

for $i \in \{n+1, \ldots, 2n\}$,

$$\vec{\xi_r} \geq 0 \ , \tag{A.24}$$

where

$$g\left(\vec{x_i}\right) = \vec{w_r} \cdot \vec{x_i} + b_r \ . \tag{A.25}$$

OP 21.

$$\min_{\vec{w_r}, b_r, \vec{\xi_r}, \vec{\xi_r^*}} \quad f\left(\vec{w_r}, b_r, \vec{\xi_r}, \vec{\xi_r^*}\right) = \frac{1}{2}\left\|\vec{w_r}\right\|^2 + C_r \sum_{i=1}^{2n} \xi_r^i \tag{A.26}$$

subject to

$$y_c^i g\left(\vec{x_i}\right) \geq y_r^i - \varepsilon_u^i - \xi_r^i \ \left(y_c^i = 1\right) \tag{A.27}$$

for $i \in \{1, \ldots, n\}$,

$$y_c^i g\left(\vec{x_i}\right) \geq -\varepsilon_d^i - y_r^i - \xi_r^i \ \left(y_c^i = -1\right) \tag{A.28}$$

for $i \in \{n+1, \ldots, 2n\}$,

$$\vec{\xi_r} \geq 0 \tag{A.29}$$

where

$$g\left(\vec{x_i}\right) = \vec{w_r} \cdot \vec{x_i} + b_r \ . \tag{A.30}$$

OP 22.

$$\min_{\vec{w_r},b_r,\vec{\xi_r},\vec{\xi_r^*}} \quad f\left(\vec{w_r},b_r,\vec{\xi_r},\vec{\xi_r^*}\right)=\frac{1}{2}\left\|\vec{w_r}\right\|^2+C_r\sum_{i=1}^{2n}\xi_r^i \tag{A.31}$$

subject to

$$y_c^i g\left(\vec{x_i}\right)\geq y_c^i y_r^i-\varepsilon_i-\xi_r^i \tag{A.32}$$

$$\vec{\xi_r}\geq 0 \tag{A.33}$$

for $i\in\{1,\ldots,2n\}$, where

$$\varepsilon_i=\varepsilon_u^i \text{ for } i\in\{1,\ldots,n\} \quad , \tag{A.34}$$

$$\varepsilon_i=\varepsilon_d^i \text{ for } i\in\{n+1,\ldots,2n\} \quad , \tag{A.35}$$

$$g\left(\vec{x_i}\right)=\vec{w_r}\cdot\vec{x_i}+b_r \quad . \tag{A.36}$$

OP 23.

$$\min_{\vec{w_r},b_r,\vec{\xi_r},\vec{\xi_r^*}} \quad f\left(\vec{w_r},b_r,\vec{\xi_r},\vec{\xi_r^*}\right)=\frac{1}{2}\left\|\vec{w_r}\right\|^2+C_r\sum_{i=1}^{2n}\xi_r^i \tag{A.37}$$

subject to

$$y_c^i g\left(\vec{x_i}\right)\geq 1+y_c^i y_r^i-\varepsilon_i-\xi_r^i-1 \tag{A.38}$$

$$\vec{\xi_r}\geq 0 \tag{A.39}$$

for $i\in\{1,\ldots,2n\}$, where

$$\varepsilon_i=\varepsilon_u^i \text{ for } i\in\{1,\ldots,n\} \quad , \tag{A.40}$$

$$\varepsilon_i=\varepsilon_d^i \text{ for } i\in\{n+1,\ldots,2n\} \quad , \tag{A.41}$$

$$g\left(\vec{x_i}\right)=\vec{w_r}\cdot\vec{x_i}+b_r \quad . \tag{A.42}$$

And we have

$$w_i=\sum_{j=1}^{2n}y_c^j\alpha_j x_{ij}=\sum_{j=1}^{n}\alpha_j x_{ij}-\sum_{j=n+1}^{2n}\alpha_j^* x_{ij}=\sum_{j=1}^{n}\left(\alpha_j-\alpha_j^*\right)x_{ij} \quad . \tag{A.43}$$

A.3 Derivation of the Dual Form of OP 16

OP 24.

$$\max_{\vec{\alpha},\vec{r}} \quad d\left(\vec{\alpha},\vec{r}\right) \tag{A.44}$$

where

$$\begin{aligned} d\left(\vec{\alpha},\vec{r}\right) &= \min_{\vec{w}} t\left(\vec{w},\vec{\alpha},\vec{\xi},\vec{r}\right) \\ t\left(\vec{w},\vec{\alpha},\vec{\xi},\vec{r}\right) &= \frac{1}{2}\left\|\vec{w}\right\|^2+\sum_{i=1}^{n}C_i\xi_i- \\ &\quad -\sum_{i=1}^{n}\alpha_i\left(y_i\vec{w}\cdot\vec{x_i}-1+\xi_i-\varphi_i\right)-\sum_{i=1}^{n}r_i\xi_i \end{aligned}$$

subject to

$$\begin{aligned} \alpha_i &\geq 0 \\ r_i &\geq 0 \end{aligned}$$

for $i \in \{1, \ldots, n\}$.

A partial derivative with respect to w_i is

$$\frac{\partial h\left(\vec{w}, \vec{\alpha}, \vec{\xi}, \vec{r}\right)}{\partial w_i} = w_i - \sum_{j=1}^{n} \alpha_j y_j x_{ji} = 0 \tag{A.45}$$

for $i \in \{1, \ldots, m\}$. A partial derivative with respect to ξ_i is

$$\frac{\partial h\left(\vec{w}, \vec{\alpha}, \vec{\xi}, \vec{r}\right)}{\partial \xi_i} = C_i - r_i - \alpha_i = 0 \ . \tag{A.46}$$

After substitution of above equations to $d\left(\vec{\alpha}, \vec{r}\right)$ we get

$$\begin{aligned} d\left(\vec{\alpha}, \vec{r}\right) &= \tfrac{1}{2} \sum_{i=1}^{m} \left(\sum_{j=1}^{n} \alpha_j y_j x_{ji} \right) \left(\sum_{k=1}^{n} \alpha_k y_k x_{ki} \right) \\ &- \sum_{i=1}^{n} \alpha_i y_i \left(\sum_{j=1}^{m} w_j x_{ij} \right) + \sum_{i=1}^{n} \alpha_i \left(1 + \varphi_i\right) + C_i \sum_{i=1}^{n} \xi_i \\ &- \sum_{i=1}^{n} \alpha_i \xi_i - \sum_{i=1}^{n} r_i \xi_i \end{aligned} \tag{A.47}$$

$$\begin{aligned} d\left(\vec{\alpha}, \vec{r}\right) &= \tfrac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{n} \alpha_k \alpha_j y_k y_j x_{ki} x_{ji} - \sum_{i=1}^{n} \alpha_i y_i \sum_{j=1}^{m} w_j x_{ij} \\ &+ \sum_{i=1}^{n} \alpha_i \left(1 + \varphi_i\right) \end{aligned} \tag{A.48}$$

$$\begin{aligned} d\left(\vec{\alpha}, \vec{r}\right) &= \tfrac{1}{2} \sum_{j=1}^{n} \sum_{k=1}^{n} \alpha_k \alpha_j y_k y_j \sum_{i=1}^{m} x_{ji} x_{ki} \\ &- \sum_{i=1}^{n} \alpha_i y_i \sum_{j=1}^{m} x_{ij} \sum_{k=1}^{n} \alpha_k y_k x_{kj} + \sum_{i=1}^{n} \alpha_i \left(1 + \varphi_i\right) \end{aligned} \tag{A.49}$$

$$\begin{aligned} d\left(\vec{\alpha}, \vec{r}\right) &= \tfrac{1}{2} \sum_{j=1}^{n} \sum_{k=1}^{n} \alpha_k \alpha_j y_k y_j \sum_{i=1}^{m} x_{ji} x_{ki} \\ &- \sum_{i=1}^{n} \sum_{k=1}^{n} \alpha_k \alpha_i y_k y_i \sum_{j=1}^{m} x_{ij} x_{kj} + \sum_{i=1}^{n} \alpha_i \left(1 + \varphi_i\right) \end{aligned} \tag{A.50}$$

$$d\left(\vec{\alpha}, \vec{r}\right) = -\frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{n} \alpha_k \alpha_i y_k y_i \sum_{j=1}^{m} x_{ij} x_{kj} + \sum_{i=1}^{n} \alpha_i \left(1 + \varphi_i\right) \ . \tag{A.51}$$

The dual form is

OP 25.

$$\max_{\vec{\alpha},\vec{r}} \quad d\left(\vec{\alpha},\vec{r}\right) = \sum_{i=1}^{n} \alpha_i \left(1+\varphi_i\right) - \frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{n} \alpha_k \alpha_i y_k y_i \sum_{j=1}^{m} x_{ij} x_{kj} \tag{A.52}$$

subject to

$$C_i = r_i + \alpha_i \tag{A.53}$$

$$\alpha_i \geq 0 \tag{A.54}$$

$$r_i \geq 0 \tag{A.55}$$

for $i \in \{1,\ldots,n\}$.

A.4 Incorporation of the Linear Dependency to φ-SVC

We will incorporate (2.44) to OP 11. After reformulation

$$\sum_{i=1}^{s} s_i \vec{w}_c \cdot \vec{d}_i + b\sum_{i=1}^{s} s_i = e \tag{A.56}$$

$$b = \frac{1}{\sum_{i=1}^{s} s_i}\left(e - \sum_{i=1}^{s} s_i \vec{w}_c \cdot \vec{d}_i\right) \quad . \tag{A.57}$$

Now we can substitute b to $h\left(\vec{x}\right)$ and we get

$$h\left(\vec{x}\right) = \vec{w}_c \cdot \vec{x} + \frac{1}{\sum_{i=1}^{s} s_i}\left(e - \sum_{i=1}^{s} s_i \vec{w}_c \cdot \vec{d}_i\right) \tag{A.58}$$

after reformulation

$$h\left(\vec{x}\right) = \vec{w}_c \cdot \vec{x} - \frac{1}{\sum_{i=1}^{s} s_i}\vec{w}_c \cdot \sum_{i=1}^{s} s_i \vec{d}_i + \frac{e}{\sum_{i=1}^{s} s_i} \quad . \tag{A.59}$$

After substituting above to OP 11, we get the φ-SVC problem without the offset with the new kernel in the form of transformation of the input vectors

$$\vec{x} \to \vec{x} - \frac{1}{\sum_{i=1}^{s} s_i}\sum_{i=1}^{s} s_i \vec{d}_i \tag{A.60}$$

$$K\left(\vec{x},\vec{y}\right) = \left(\vec{x} - \frac{1}{\sum_{i=1}^{s} s_i}\sum_{i=1}^{s} s_i \vec{d}_i\right)\left(\vec{y} - \frac{1}{\sum_{i=1}^{s} s_i}\sum_{i=1}^{s} s_i \vec{d}_i\right) \tag{A.61}$$

and φ_i weights set as

$$\varphi_i = \varphi_{old} - y_i \frac{e}{\sum_{i=1}^{s} s_i} \quad . \tag{A.62}$$

We get nonlinear solutions by using the following way of further kernelization

$$K\left(\vec{x},\vec{y}\right)=K_o\left(\vec{x},\vec{y}\right)-\frac{1}{\sum_{i=1}^{s}s_i}\sum_{i=1}^{s}s_iK_o\left(\vec{x},\vec{d}_i\right)-\frac{1}{\sum_{i=1}^{s}s_i}\sum_{i=1}^{s}s_iK_o\left(\vec{y},\vec{d}_i\right) \tag{A.63}$$

$$+\frac{1}{\left(\sum_{i=1}^{s}s_i\right)^2}\sum_{i=1}^{s}\sum_{j=1}^{s}s_is_jK_o\left(\vec{d}_i,\vec{d}_j\right) \tag{A.64}$$

This kernel is used only for solving the optimization problem. Now we derive the solution, because

$$\vec{w_c}=\sum_{j=1}^{n}\alpha_jy_c^j\left(\vec{x_j}-\frac{1}{\sum_{i=1}^{s}s_i}\sum_{i=1}^{s}s_i\vec{d}_i\right) \tag{A.65}$$

we get

$$h\left(\vec{x}\right)=\vec{w_c}\cdot\vec{x}+b=\sum_{j=1}^{n}\alpha_jy_c^j\left(\vec{x_j}-\frac{1}{\sum_{i=1}^{s}s_i}\sum_{i=1}^{s}s_i\vec{d}_i\right)\cdot\vec{x}+b \tag{A.66}$$

$$=\sum_{j=1}^{n}\alpha_jy_c^j\vec{x_j}\cdot\vec{x}-\frac{1}{\sum_{i=1}^{s}s_i}\sum_{j=1}^{n}\sum_{i=1}^{s}\alpha_jy_c^js_i\vec{d}_i\cdot\vec{x}+b \tag{A.67}$$

and after adding internal kernels we get

$$h\left(\vec{x}\right)=\sum_{j=1}^{n}\alpha_jy_c^jK_o\left(\vec{x_j},\vec{x}\right)-\frac{1}{\sum_{i=1}^{s}s_i}\sum_{j=1}^{n}\sum_{i=1}^{s}\alpha_jy_c^js_iK_o\left(\vec{d}_i,\vec{x}\right)+b \tag{A.68}$$

when b is computed as

$$b=\frac{1}{\sum_{i=1}^{s}s_i}\left(e-\sum_{i=1}^{s}s_i\sum_{j=1}^{n}\alpha_jy_c^j\left(\vec{x_j}-\frac{1}{\sum_{i=1}^{s}s_i}\sum_{k=1}^{s}s_k\vec{d_k}\right)\cdot\vec{d}_i\right) \tag{A.69}$$

$$=\frac{1}{\sum_{i=1}^{s}s_i}\left(e-\sum_{i=1}^{s}s_i\sum_{j=1}^{n}\alpha_jy_c^jK_o\left(\vec{x_j},\vec{d}_i\right)+\frac{1}{\sum_{i=1}^{s}s_i}\sum_{i=1}^{s}s_i\sum_{j=1}^{n}\sum_{k=1}^{s}\alpha_jy_c^js_kK_o\left(\vec{d}_k,\vec{d}_i\right)\right) \tag{A.70}$$

A.5 Incorporation of the Linear Dependency to δ-SVR

We will incorporate (2.43) to δ-SVR. We assume that we use special kernels developed for δ-SVR, (1.49). For δ-SVR the condition (2.43) becomes

$$\sum_{i=1}^{s}s_i\frac{-\vec{w_{\mathrm{red}}}\cdot\vec{d_{\mathrm{i,red}}}-b_{\mathrm{c}}}{w_{\mathrm{c}}^{m+1}}=e\ , \tag{A.71}$$

After transformation

$$ew_{\mathrm{c}}^{m+1} + \sum_{i=1}^{s} s_i \vec{w_{\mathrm{red}}} \cdot \vec{d_{\mathrm{i,red}}} + \sum_{i=1}^{s} s_i b_c = 0 \tag{A.72}$$

$$b_c = \frac{1}{\sum_{i=1}^{s} s_i} \left(-ew_{\mathrm{c}}^{m+1} - \sum_{i=1}^{s} s_i \vec{w_{\mathrm{red}}} \cdot \vec{d_{\mathrm{i,red}}} \right) \tag{A.73}$$

Substituting it to $h\left(\vec{x}\right)$ we get

$$h\left(\vec{x}\right) = \vec{w_c} \cdot \vec{x} + \frac{1}{\sum_{i=1}^{s} s_i} \left(-ew_{\mathrm{c}}^{m+1} - \sum_{i=1}^{s} s_i \vec{w_{\mathrm{red}}} \cdot \vec{d_{\mathrm{i,red}}} \right) \tag{A.74}$$

$$h\left(\vec{x}\right) = \vec{w_{\mathrm{red}}} \cdot \vec{x_{\mathrm{red}}} + w_c^{m+1} x_{m+1} + \frac{1}{\sum_{i=1}^{s} s_i} \left(-ew_{\mathrm{c}}^{m+1} - \sum_{i=1}^{s} s_i \vec{w_{\mathrm{red}}} \cdot \vec{d_{\mathrm{i,red}}} \right) \tag{A.75}$$

$$h\left(\vec{x}\right) = \vec{w_{\mathrm{red}}} \cdot \left(\vec{x_{\mathrm{red}}} - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i \vec{d_{\mathrm{i,red}}} \right) + w_c^{m+1} \left(x_{m+1} - \frac{e}{\sum_{i=1}^{s} s_i} \right) \tag{A.76}$$

After substituting above to OP 11, we get the φ-SVC problem without the offset with the new kernel in the form

$$K\left(\vec{x}, \vec{y}\right) = \left(\vec{x_{\mathrm{red}}} - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i \vec{d_{\mathrm{i,red}}} \right) \left(\vec{y_{\mathrm{red}}} - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i \vec{d_{\mathrm{i,red}}} \right) \tag{A.77}$$

$$+ \left(x_{m+1} - \frac{e}{\sum_{i=1}^{s} s_i} \right) \left(y_{m+1} - \frac{e}{\sum_{i=1}^{s} s_i} \right) \tag{A.78}$$

Then

$$K\left(\vec{x}, \vec{y}\right) = \vec{x_{\mathrm{red}}} \cdot \vec{y_{\mathrm{red}}} - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i \vec{d_{\mathrm{i,red}}} \cdot \vec{x_{\mathrm{red}}} - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i \vec{d_{\mathrm{i,red}}} \cdot \vec{y_{\mathrm{red}}} \tag{A.79}$$

$$+ \frac{1}{\left(\sum_{i=1}^{s} s_i\right)^2} \sum_{i=1}^{s} \sum_{j=1}^{s} s_i s_j \vec{d_{\mathrm{i,red}}} \cdot \vec{d_{\mathrm{j,red}}} + x_{m+1} y_{m+1} - x_{m+1} \frac{e}{\sum_{i=1}^{s} s_i} - y_{m+1} \frac{e}{\sum_{i=1}^{s} s_i} \tag{A.80}$$

$$+ \frac{e^2}{\left(\sum_{i=1}^{s} s_i\right)^2} \tag{A.81}$$

We get nonlinear solutions by using the following way of further kernelization

$$K\left(\vec{x}, \vec{y}\right) = K_o\left(\vec{x_{\mathrm{red}}}, \vec{y_{\mathrm{red}}}\right) - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i K_o\left(\vec{d_{\mathrm{i,red}}}, \vec{x_{\mathrm{red}}}\right) \tag{A.82}$$

$$- \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i K_o\left(\vec{d_{\mathrm{i,red}}}, \vec{y_{\mathrm{red}}}\right) + \frac{1}{\left(\sum_{i=1}^{s} s_i\right)^2} \sum_{i=1}^{s} \sum_{j=1}^{s} s_i s_j K_o\left(\vec{d_{\mathrm{i,red}}}, \vec{d_{\mathrm{j,red}}}\right) \tag{A.83}$$

$$+ x_{m+1}y_{m+1} - x_{m+1}\frac{e}{\sum_{i=1}^{s} s_i} - y_{m+1}\frac{e}{\sum_{i=1}^{s} s_i} + \frac{e^2}{\left(\sum_{i=1}^{s} s_i\right)^2} \ . \tag{A.84}$$

This kernel is used only for solving the optimization problem. So solving δ-SVR with the additional constraint leads to SVC optimization problem without the offset and with a special kernel presented above. Now we derive the solution

$$h\left(\vec{x}\right) = \vec{w_c} \cdot \vec{x} + b_c = \sum_{j=1}^{2n} \alpha_j y_c^j \left(x_{\mathrm{j,red}}^{\rightarrow} - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i d_{\mathrm{i,red}}^{\rightarrow} \right) \cdot x_{\mathrm{red}}^{\rightarrow} \tag{A.85}$$

$$+ \sum_{j=1}^{2n} \alpha_j y_c^j \left(x_j^{m+1} - \frac{e}{\sum_{i=1}^{s} s_i} \right) x_{m+1} + b_c \tag{A.86}$$

with internal kernels

$$h\left(\vec{x}\right) = \sum_{j=1}^{2n} \alpha_j y_c^j \left(K_o\left(x_{\mathrm{j,red}}^{\rightarrow}, x_{\mathrm{red}}^{\rightarrow}\right) - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i K_o\left(d_{\mathrm{i,red}}^{\rightarrow}, x_{\mathrm{red}}^{\rightarrow}\right) \right) \tag{A.87}$$

$$+ \sum_{j=1}^{2n} \alpha_j y_c^j \left(x_j^{m+1} - \frac{e}{\sum_{i=1}^{s} s_i} \right) x_{m+1} + b_c \tag{A.88}$$

where b_c is computed as

$$b_c = \frac{1}{\sum_{i=1}^{s} s_i} \left(-e \sum_{j=1}^{2n} \alpha_j y_c^j \left(x_j^{m+1} - \frac{e}{\sum_{i=1}^{s} s_i} \right) \right) \tag{A.89}$$

$$- \frac{1}{\sum_{i=1}^{s} s_i} \left(\sum_{i=1}^{s} s_i \sum_{j=1}^{2n} \alpha_j y_c^j \left(x_{\mathrm{j,red}}^{\rightarrow} - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{k=1}^{s} s_k d_{\mathrm{k,red}}^{\rightarrow} \right) \cdot d_{\mathrm{i,red}}^{\rightarrow} \right) \tag{A.90}$$

$$= -\frac{1}{\sum_{i=1}^{s} s_i} e \sum_{j=1}^{2n} \alpha_j y_c^j \left(x_j^{m+1} - \frac{e}{\sum_{i=1}^{s} s_i} \right) - \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i \sum_{j=1}^{2n} \alpha_j y_c^j K_o\left(x_{\mathrm{j,red}}^{\rightarrow}, d_{\mathrm{i,red}}^{\rightarrow}\right) \tag{A.91}$$

$$+ \frac{1}{\sum_{i=1}^{s} s_i} \sum_{i=1}^{s} s_i \sum_{j=1}^{2n} \alpha_j y_c^j \frac{1}{\sum_{i=1}^{s} s_i} \sum_{k=1}^{s} s_k K_o\left(d_{\mathrm{k,red}}^{\rightarrow}, d_{\mathrm{i,red}}^{\rightarrow}\right) \tag{A.92}$$

References

[1] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm. 15, 29

[2] Yuh jye Lee and Olvi L. Mangasarian. Rsvm: Reduced support vector machines. In *Data Mining Institute, Computer Sciences Department, University of Wisconsin*, pages 00–07, 2001. 27

[3] Masayuki Karasuyama, Ichiro Takeuchi, and Ryohei Nakano. Reducing svr support vectors by using backward deletion. In *KES '08: Proceedings of the 12th international conference on Knowledge-Based Intelligent Information and Engineering Systems, Part III*, pages 76–83, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-85566-8. 27

[4] S. Sathiya Keerthi, Olivier Chapelle, and Dennis DeCoste. Building support vector machines with reduced classifier complexity. *J. Mach. Learn. Res.*, 7:1493–1515, December 2006. ISSN 1532-4435. 27

[5] LibSVM data sets. Libsvm data sets. www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/, 06 2011. 29

[6] Olvi L. Mangasarian, Jude W. Shavlik, and Edward W. Wild. Knowledge-based kernel approximation. *Journal of Machine Learning Research*, 5:1127–1141, 2004. 27

[7] Kuan ming Lin and Chih jen Lin. A study on reduced support vector machines. *IEEE TRANSACTIONS ON NEURAL NETWORKS*, 14:1449–1459, 2003. 27

[8] Marcin Orchel. Incorporating detractors into svm classification. In Krzysztof Cyran, Stanislaw Kozielski, James Peters, Urszula Stańczyk, and Alicja Wakulicz-Deja, editors, *Man-Machine Interactions*, volume 59 of *Advances in Intelligent and Soft Computing*, pages 361–369. Springer Berlin / Heidelberg, 2009. ISBN 978-3-642-00562-6. doi: 10.1007/978-3-642-00563-3_38. URL http://dx.doi.org/10.1007/978-3-642-00563-3_38. 11

[9] Marcin Orchel. Incorporating a priori knowledge from detractor points into support vector classification. In Andrej Dobnikar, Uroš Lotric, and Branko Šter, editors, *Adaptive and Natural Computing Algorithms*, volume 6594 of *Lecture Notes in Computer Science*, pages 332–341. Springer Berlin / Heidelberg, 2011. ISBN 978-3-642-20266-7. doi: 10.1007/978-3-642-20267-4_35. URL http://dx.doi.org/10.1007/978-3-642-20267-4_35. 11, 27

[10] Marcin Orchel. Regression based on support vector classification. In Andrej Dobnikar, Uroš Lotric, and Branko Šter, editors, *Adaptive and Natural Computing Algorithms*, volume 6594 of *Lecture Notes in Computer Science*, pages 353–362. Springer Berlin / Heidelberg, 2011. ISBN 978-3-642-20266-7. doi: 10.1007/978-3-642-20267-4_37. URL http://dx.doi.org/10.1007/978-3-642-20267-4_37. 9

[11] Marcin Orchel. Support vector regression as a classification problem with a priori knowledge in the form of detractors. In Tadeusz Czachorski, Stanislaw Kozielski, and Urszula Stańczyk, editors, *Man-Machine Interactions 2*, volume 103 of *Advances in Intelligent and Soft Computing*, pages 353–362. Springer Berlin / Heidelberg, 2011. ISBN 978-3-642-23168-1. doi: 10.1007/978-3-642-23169-8_38. URL http://dx.doi.org/10.1007/978-3-642-23169-8_38. 11, 15, 27

[12] John C. Platt. *Fast training of support vector machines using sequential minimal optimization*, pages 185–208. MIT Press, Cambridge, MA, USA, 1999. ISBN 0-262-19416-3. 14

[13] Ingo Steinwart, Don R. Hush, and Clint Scovel. Training svms without offset. *Journal of Machine Learning Research*, 12:141–202, 2011. 5

[14] Francis Eng Hock Tay and Lijuan Cao. Modified support vector machines in financial time series forecasting. *Neurocomputing*, 48(1-4):847–861, 2002. 7

[15] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, September 1998. ISBN 0471030031. 7

[16] Mingrui Wu, Bernhard Schlkopf, and Gkhan Bakir. A direct method for building sparse kernel learning algorithms. *JOURNAL OF MACHINE LEARNING RESEARCH*, 7:603–624, 2006. 27

[17] Xiaoyun Wu and Rohini Srihari. Incorporating prior knowledge with weighted margin support vector machines. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 326–333, New York, NY, USA, 2004. ACM. ISBN 1-58113-888-1. 11

Notation and Symbols

Miscellaneous

$|A|$ the cardinality of a finite set A, i.e., the number of elements in the set A.

$\cdot$ Dot product of two vectors, sometimes it is written with additional parentheses, for example for two vectors: $\vec{u}$ and $\vec{v}$, the dot product is $\vec{u} \cdot \vec{v}$ or $(\vec{u} \cdot \vec{v})$.

$\vec{v} \geq \vec{w}$ For two n dimensional vectors $\vec{v}$ and $\vec{w}$, it means that for all $i = 1...n$ $v_i \geq w_i$.

$\vec{v} \gg \vec{w}$ For two n dimensional vectors $\vec{v}$ and $\vec{w}$, it means that for all $i = 1...n$ $v_i > w_i$.

$\rho(A)$ the rank of a matrix A.

$\vec{w}^i_{\mathbf{r}}$ When a vector has an index in the subscript, the coefficient index is placed in the superscript, the example means the i-th coefficient of the $\vec{w}_{\mathrm{r}}$.

Optimization theory

* an asterisk as a superscript in optimization theory denotes a solution of the optimization problem.

Abbreviations

δ-SVR δ support vector regression.

ν-SVC ν support vector classification.

ε-SVR ε-insensitive support vector regression.

φ-SVC φ support vector classification.

C-SVC C support vector classification.

DJIA Dow Jones Industrial Average.

KKT Karush-Kuhn-Tucker.

RBF radial basis function.

RSVM reduced support vector machines.

SMO sequential minimal optimization.

SVC support vector classification.

SVM support vector machines.

VC Vapnik-Chervonenkis.