Solving SVM, SVM+, SVR and SVR+ using Quadratic Programming

Nikolaos Sarafianos, Christophoros Nikou, Ioannis A. Kakadiaris

March 8, 2017

This tutorial along with the code provided were used for the *"Predicting Privileged Information for Height Estimation"* publication [1]. To create this tutorial we used the work of Vapnik and Vashist [2] which introduces the LUPI paradigm and contains the dual formulation in all cases discussed below, the tutorial on Support Vector Regression of Smola and Schölkopf [3] and the study on SMO-type decomposition of Chen *et al.* [4]. If you found this tutorial useful please cite accordingly.

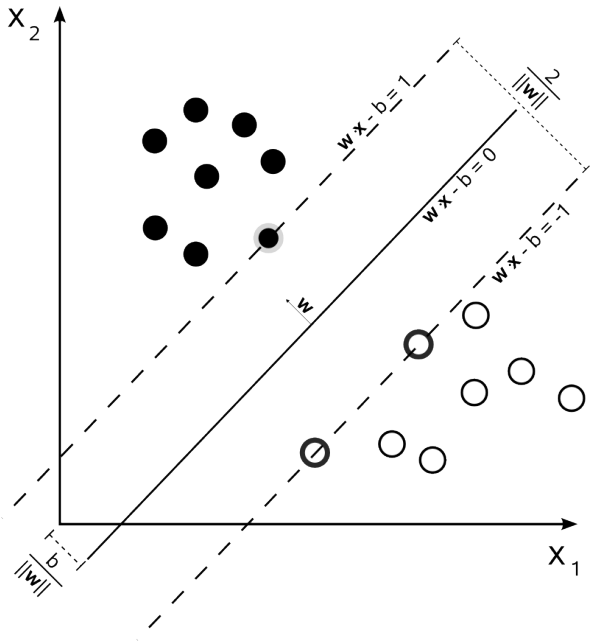
For the implementation part we used the CVXOPT[1] Python package for convex optimization. The quadrprog function in MATLAB can be used as well. For simplification, throughout the tutorial MATLAB notation is followed and thus, [a;b] refers to the concatenation of vectors, .* refers to element-wise multiplication. Functions such as eye, ones, zeros and repmat are used accordingly.
Note that a Quadratic Programming solution is reasonable only for datasets with not a lot of samples. For larger datasets SMO is strongly encouraged.

Quadratic Programming Formulation

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \frac{1}{2}x^T Q x \ + \ p^T x \\ \text{subject to} \quad & Ax = b, \ Gx \leq h \end{aligned}$$

Input features X of dimensions: $l \times m$ and label vector Y of dimensions $l \times 1$. In all cases we will use a Gaussian Kernel with γ_{rbf}. The equation of the Gaussian Kernel can be found here.

1 SVM



[1] http://cvxopt.org/

The dual formulation of the problem is as follows:

$$\begin{aligned} \underset{\alpha}{\text{minimize}} \quad & R(\alpha) = \frac{1}{2}\sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j) - \sum_{i=1}^{l} \alpha_i \\ \text{subject to} \quad & \sum_{i=1}^{l} y_i \alpha_i = 0,\ 0 \leq \alpha_i \leq C \end{aligned}$$

The vector we obtain from the minimization is $x = \alpha$ and has dimensions $l \times 1$. The formulation of the matrices in the quadratic programming formulation is as follows:

- $Q = (Y^T * Y)\ .*K$ with size of $l \times l$. The matrix multiplication of $Y^T * Y$ gives us an $l \times l$ matrix which contains the $y_i y_j$ terms and then we perform element-wise multiplication to get Q.
- $p = -ones(l, 1)$ of size $l \times 1$ so as when we obtain the transpose and multiply by x which is of size $l \times 1$ the result will be a number.
- $A = Y^T$ of size $1 \times l$ so as it can be multiplied by x.
- $b = 0$
- $G = [-eye(l)\ ;\ eye(l)]$ of size $2l \times l$. Through this we will formulate the range of α. The first l rows will correspond to the inequality $-\alpha \leq 0$ and the second l rows to the inequality $\alpha \leq C$.
- $h = [zeros(l, 1)\ ;\ C * ones(l, 1)]$ of size $2l \times l$.

Having obtained the matrices we solve for x and get the respective α. Given a threshold $thresh$ we select the α which are greater than this threshold. We obtain the respective support vectors and their corresponding labels and we then compute the bias by averaging over all the support vectors.
For each testing sample, the kernel matrix between each of the support vectors and the respective testing sample is computed. The rest is straightforward following the Eq. 19 of [2]:

$$f(x) = \sum_{i=1}^{l} y_i \alpha_i K(x_i, x) + b$$

and mapping the negative values to the -1 class and the positive to the +1 class.

Parameters: C, γ_{rbf}, $thresh$.

2 SVM+

Besides X in that case we also have X^* of size $l \times m_1$ which resides in another space, different than the original space X. We will refer to X^* as privileged features and to their space as privileged space. In this case the dual formulation has an extra term which corresponds to the privileged (correcting) space which is multiplied by a parameter γ_{cor} and is as follows:

$$\underset{\alpha,\beta}{\text{minimize}} \quad R(\alpha,\beta) = \frac{1}{2}\sum_{i,j=1}^{l}\alpha_i\alpha_j y_i y_j K(x_i,x_j) + \frac{1}{2\gamma_{cor}}\sum_{i,j=1}^{l}(\alpha_i+\beta_i-C)(\alpha_j+\beta_j-C)K^*(x_i^*,x_j^*)$$
$$-\sum_{i=1}^{l}\alpha_i$$
$$\text{subject to} \quad \sum_{i=1}^{l} y_i\alpha_i = 0,\ \sum_{i=1}^{l}(\alpha_i+\beta_i-C) = 0, 0 \le \alpha_i,\beta_i$$

The vector we obtain from the minimization is $x = [\alpha;\beta]$ and has dimensions $2l \times 1$. The matrices in the quadratic programming formulation are as follows:

- $$Q = \begin{bmatrix} (Y^T * Y) \,.*\, K + \frac{K^*}{\gamma_{cor}} & \frac{K^*}{\gamma_{cor}} \\ \frac{K^*}{\gamma_{cor}} & \frac{K^*}{\gamma_{cor}} \end{bmatrix}$$
  with size of $2l \times 2l$. The first element corresponds to the α^2 term which consists of the whole first sum and of the $\alpha_i\alpha_j K^*$ part of the second sum. The rest are obtained from the second sum.

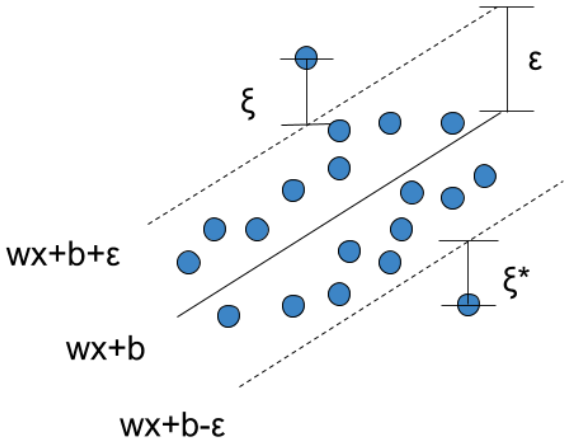
- $p^T = [-ones(1,l); zeros(1,l)] - \frac{C}{2\gamma_{cor}}[repmat\Big(sum(K^* + K^{*T},1),1,2\Big)]$ of size $2l \times 1$.
  The first part is obvious and comes from the $\sum_{i=1}^{l}\alpha_i$ term and it's followed by zeros that correspond to β. The second term is more tricky and thus, let's break it down with an example.
  From the second sum that corresponds to the correcting space we have used in matrix Q, four out of its nine terms. The term $C^2 * K^*$ is a constant term and thus we ignore it. For the four remaining terms this sum is written as: $\frac{C}{2\gamma_{cor}}\sum_{i,j=1}^{l}(\alpha_i+\beta_i+\alpha_j+\beta_j) * K^*(x_i^*,x_j^*)$. Suppose l is equal to three. We then have the following:
  $(\alpha_1+\beta_1+\alpha_1+\beta_1) * K_{11}^* + (\alpha_1+\beta_1+\alpha_2+\beta_2) * K_{12}^* + (\alpha_1+\beta_1+\alpha_3+\beta_3) * K_{13}^* +$
  $(\alpha_2+\beta_2+\alpha_1+\beta_1) * K_{21}^* + (\alpha_2+\beta_2+\alpha_2+\beta_2) * K_{22}^* + (\alpha_2+\beta_2+\alpha_3+\beta_2) * K_{23}^* +$
  $(\alpha_3+\beta_3+\alpha_1+\beta_1) * K_{31}^* + (\alpha_3+\beta_3+\alpha_2+\beta_2) * K_{32}^* + (\alpha_3+\beta_3+\alpha_3+\beta_3) * K_{33}^*$
  For the α (because for the β is the same and that's why we repmat) this sum can be represented as the following matrix multiplication:
  $[2K_{11}^* + K_{12}^* + K_{13}^* + K_{21}^* + K_{31}^*,\ 2K_{22}^* + K_{12}^* + K_{32}^* + K_{21}^* + K_{23}^*,\ 2K_{33}^* + K_{13}^* + K_{23}^* +$
  $K_{31}^* + K_{32}^*] * [\alpha_1,\alpha_2,\alpha_3]^T$. Each term is the sum in one dimension (for example rows) of the sum of K^* and K^{*T} in the respective index i. Note that you need to transpose it back in order to pass it, to the optimization algorithm.

- $A = [Y^T, zeros(1,l)\ ;\ ones(1,2l)]$ of size $2 \times 2l$
- $b = [0; lC]$
- $G = -eye(2l)$ of size $2l \times 2l$.
- $h = zeros(2l,1)$ of size $2l \times 1$.

Having obtained the matrices we solve for x and get the respective $\alpha = x(1 : l)$ and $\beta = x(l+1 : 2l)$. The process to obtain the support vectors (we are interested only of those in the original space since those are the ones used in the decision function) and to evaluate on the testing set is the same with SVM.

Parameters: C, $\gamma_{rbf}, \gamma_{rbf^*}$, $thresh$, γ_{cor}.

3 ϵ-SVR



According to Smola and Schölkopf [3] in support vector regression our goal is to find a function f(x) that has the most ϵ deviation from the actually obtained targets y_i for all the training data, and at the same time, is as flat as possible. In other words, we do not care about errors as long as they are less than ϵ but will not accept any deviation larger than this. The dual formulation of this problem is as follows:

$$\underset{\alpha,\alpha^*}{\text{minimize}} \quad R(\alpha,\alpha^*) = \frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(x_i,x_j) + \sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*) + \epsilon\sum_{i=1}^{l}(\alpha_i + \alpha_i^*)$$

$$\text{subject to} \quad \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0,\ 0 \le \alpha_i, \alpha_i^* \le C$$

Note that the α^* in that case has nothing to do with the privileged space. It denotes the space of width ϵ below the margin. In ϵ-SVR we're looking for $x = [\alpha\ ;\ \alpha^*]$ the dimensions of which are $2l \times 1$. Following exactly the same procedure with Chen *et al.* [4] we obtain the following:

- $$Q = \begin{bmatrix} K & -K \\ -K & K \end{bmatrix}$$

  with size of $2l \times 2l$.

- $p = [\epsilon * ones(l,1) + Y,\ \epsilon * ones(l,1) - Y]$ of size $2l \times 1$ which combines the last two sums in the minimization function.
- $A = [ones(1,l)\ ,\ -ones(1,l)]$ of size $1 \times 2l$.
- $b = 0$
- $G = [-eye(2l)\ ;\ eye(2l)]$ of size $4l \times 2l$.
- $h = [zeros(2l,1)\ ;\ C * ones(2l,1)]$ of size $4l \times l$.

We solve the optimization problem and obtain the α, α^* we map to zero those who are smaller than a threshold and then we compute the bias. For each testing sample the kernel between each training sample and the respective testing is computed as before and then the solution is obtained by applying:

$$y = \sum_{i=1}^{l} (\alpha_i^* - \alpha_i) K(x_i, x) + b$$

Parameters: $C,\ \gamma_{rbf},\ thresh,\ \epsilon$.

4 ϵ-SVR+

In ϵ-SVR+ the dual formulation of the problem is as follows:

$$\underset{\alpha,\alpha^*,\beta,\beta^*}{\text{minimize}}\ R(\alpha,\alpha^*,\beta,\beta^*) = \frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(x_i,x_j) +$$
$$\frac{1}{2\gamma_{cor}}\sum_{i,j=1}^{l}(\alpha_i^* + \beta_i^* - C)(\alpha_j^* + \beta_j^* - C)K^*(x_i^*,x_j^*) +$$
$$\frac{1}{2\gamma_{cor}}\sum_{i,j=1}^{l}(\alpha_i + \beta_i - C)(\alpha_j + \beta_j - C)K^*(x_i^*,x_j^*) + \sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*) + \epsilon\sum_{i=1}^{l}(\alpha_i + \alpha_i^*)$$
$$\text{subject to} \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0,\ \sum_{i=1}^{l}(\alpha_i^* + \beta_i^* - C) = 0,\ \sum_{i=1}^{l}(\alpha_i + \beta_i - C) = 0,\ 0 \le \alpha_i, \alpha_i^* \le C$$

In ϵ-SVR+ we're looking for $x = [\alpha\ ;\ \alpha^*\ ;\ \beta\ ;\ \beta^*]$ the dimensions of which are $4l \times 1$. α, α^* correspond to the solution in the original space whereas β, β^* to the privileged space. Using the same procedure with SVM+ to obtain the respective matrices we have:

- $$Q = \begin{bmatrix} K + \frac{K^*}{\gamma_{cor}} & -K & \frac{K^*}{\gamma_{cor}} & zeros(l,l) \\ -K & K + \frac{K^*}{\gamma_{cor}} & zeros(l,l) & \frac{K^*}{\gamma_{cor}} \\ \frac{K^*}{\gamma_{cor}} & zeros(l,l) & \frac{K^*}{\gamma_{cor}} & zeros(l,l) \\ zeros(l,l) & \frac{K^*}{\gamma_{cor}} & zeros(l,l) & \frac{K^*}{\gamma_{cor}} \end{bmatrix}$$

  with size of $4l \times 4l$.

- $p^T = (-\frac{C}{2\gamma_{cor}}) * repmat(sum(K^* + K^{*T}, 1), 1, 4) + [Y^T + \epsilon * ones(1, l),\ -Y + \epsilon * ones(1, l),\ zeros(1, 2l)$
- G = -eye(4l)
- h = zeros(4l,1)
- $$A = \begin{bmatrix} ones(1,l) & -ones(1,l) & zeros(1,l) & zeros(1,l) \\ zeros(1,l) & ones(1,l) & zeros(1,l) & ones(1,l) \\ ones(1,l) & zeros(1,l) & ones(1,l) & zeros(1,l) \end{bmatrix}$$
  with size of $3 \times 4l$.
- b = [0 ; l*C ; l*C]

We solve the optimization problem and obtain $\alpha, \alpha^*, \beta, \beta^*$ and the bias. The process for the testing set is the same with SVR.

Parameters: $C,\ \gamma_{rbf}, \gamma_{rbf^*},\ thresh,\ \gamma_{cor}, \epsilon$.

References

[1] N. Sarafianos, N. Christophoros, I. A. Kakadiaris, Predicting privileged information for height estimation, in: Proc. International Conference on Pattern Recognition, Cancn, Mexico, 2016. 1

[2] V. Vapnik, A. Vashist, A new learning paradigm: Learning using privileged information, Neural Networks 22 (5) (2009) 544–557. 1, 2

[3] A. J. Smola, B. Schölkopf, A tutorial on support vector regression, Statistics and computing 14 (3) (2004) 199–222. 1, 4

[4] P.-H. Chen, R.-E. Fan, C.-J. Lin, Algorithmic Learning Theory: 16th International Conference, ALT 2005, Singapore, October 8-11, 2005. Proceedings, 2005, Ch. Training Support Vector Machines via SMO-Type Decomposition Methods, pp. 45–62. 1, 4