**Machine Intelligence 1** **Exercise Sheet 8**
WS 2016/17, Obermayer/Böhmer due: 05.01.2017

# Support Vector Machines

**The winter holidays start next week.**
Homework of this exercise sheet is due next year, on 05.01.2017!

## Exercise T8.1: Structural Risk Minimization (tutorial)

(a) Discuss the concept of the *margin* for the linear connectionist neuron: What is the effect of a small vs. a big margin on generalization?

(b) Write down and explain the *primal optimization problem* of model selection through structural risk minimization (SRM).

(c) Write down the Lagrangian of the primal problem and explain the intuition behind the theorem of Kuhn and Tucker. Why can we expect sparse dual variables?

(d) Discuss SVM classification of non-separable classes. How can this be regularized? Write down the primal problem of the C-SVM.

(e) What is the kernel-trick and how can we exploit it?

**Solution**

(c) The Lagrangian of the SRM optimization problem for the Support Vector Machine is

$$L(\underline{\mathbf{w}}, b, \lambda_1, \ldots, \lambda_p) := \frac{1}{2}\|\underline{\mathbf{w}}\|^2 - \sum_{\alpha=1}^{p} \lambda_\alpha \left\{ y_T^{(\alpha)} \left( \underline{\mathbf{w}}^\top \underline{\mathbf{x}}^{(\alpha)} + b \right) - 1 \right\} .$$

The derivative w.r.t. $\underline{\mathbf{w}}$ and $\underline{\mathbf{b}}$ are:

$$\frac{\partial L}{\partial \underline{\mathbf{w}}} = \underline{\mathbf{w}} - \sum_{\alpha=1}^{p} \lambda_\alpha \, y_T^{(\alpha)} \, \underline{\mathbf{x}}^{(\alpha)} \stackrel{!}{=} 0 \quad \text{and} \quad \frac{\partial L}{\partial b} = -\sum_{\alpha=1}^{p} \lambda_\alpha \, y_T^{(\alpha)} \stackrel{!}{=} 0 \, .$$

Substituting $\underline{\mathbf{w}} = \sum_{\alpha=1}^{p} \lambda_\alpha \, y_T^{(\alpha)} \, \underline{\mathbf{x}}^{(\alpha)}$ into the original Lagrangian yields the dual:

$$\max_{\boldsymbol{\lambda}} L(\lambda_1, \ldots, \lambda_p) := -\frac{1}{2} \sum_{\alpha,\beta=1}^{p} \lambda_\alpha \, \lambda_\beta \, y_T^{(\alpha)} \, y_T^{(\beta)} \boldsymbol{x}^{(\alpha)\top} \boldsymbol{x}^{(\beta)} + \sum_{\alpha=1}^{p} \lambda_\alpha$$

$$\text{s.t.} \qquad \lambda_\alpha \geq 0 \, , \quad \forall \alpha \qquad \text{and} \qquad \sum_{\alpha=1}^{p} \lambda_\alpha \, y_T^{(\alpha)} = 0 \, .$$

(d) C-SVM defines slack-variables $\varphi_\alpha \geq 0$ for all samples, and the primal problem is

$$\min_{\underline{\mathbf{w}}, b} \tfrac{1}{2}\|\underline{\mathbf{w}}\|^2 + \tfrac{C}{p} \sum_{\alpha=1}^{p} \varphi_\alpha \quad \text{s.t.} \quad y_T^{(\alpha)} \left( \underline{\mathbf{w}}^\top \underline{\mathbf{x}}^{(\alpha)} + b \right) \geq 1 - \varphi_\alpha \, , \quad \text{and} \quad \varphi_\alpha \geq 0 \, , \forall \alpha \, .$$

## Exercise H8.1: Deriving the C-SVM optimization problem (homework, 3 points)

(a) (1 point) Linear connectionist neurons have a degree of freedom that is not used in classification. By setting the constraint

$$\min_{\alpha=1,\dots,p} \left| \underline{\mathbf{w}}^\top \underline{\mathbf{x}}^{(\alpha)} + b \right| \stackrel{!}{=} 1$$

this degree is eliminated. Show that under this constraint the Euclidean distance $d(\underline{\mathbf{x}}^{(\alpha)}, \underline{\mathbf{w}}, b)$ of sample $\underline{\mathbf{x}}^{(\alpha)}$ to the closest point of the decision boundary $\{x|y(x) = 0\}$ is bounded by

$$d(\underline{\mathbf{x}}^{(\alpha)}, \underline{\mathbf{w}}, b) \geq \frac{1}{\|\underline{\mathbf{w}}\|}, \quad \forall \alpha \in \{1, \dots, p\}.$$

(b) (2 points) Write down the Lagrangian of the primal optimization problem of the C-SVM and derive the dual optimization problem of the C-SVM:

$$\max_{\boldsymbol{\lambda}} \left\{ -\frac{1}{2} \sum_{\alpha=1}^{p} \sum_{\beta=1}^{p} \lambda_\alpha \lambda_\beta \, y_T^{(\alpha)} y_T^{(\beta)} \left(\boldsymbol{x}^{(\alpha)}\right)^\top \boldsymbol{x}^{(\beta)} + \sum_{\alpha=1}^{p} \lambda_\alpha \right\}$$

$$\text{with} \qquad 0 \leq \lambda_\alpha \leq \tfrac{C}{p}\,, \forall \alpha\,, \qquad \text{and} \qquad \sum_{\alpha=1}^{p} \lambda_\alpha \, y_T^{(\alpha)} = 0\,.$$

**Solution**

(a) Let's call the closest point on the boundary $\underline{\mathbf{x}}'$. Note that for the distance $d = \|\underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{x}}'\|$ holds $\underline{\mathbf{x}}^{(\alpha)} = \underline{\mathbf{x}}' + d\,\frac{\underline{\mathbf{w}}}{\|\underline{\mathbf{w}}\|}$. Multiplying $\underline{\mathbf{w}}^\top$ from the left, adding $b$ and taking the absolutum yields $|\underline{\mathbf{w}}^\top \underline{\mathbf{x}}^{(\alpha)} + b| = \left|\underline{\mathbf{w}}^\top \underline{\mathbf{x}}' + b + d\,\|\underline{\mathbf{w}}\|\right|$. The point $\underline{\mathbf{x}}'$ lies on the decision surface, which implies that $\underline{\mathbf{w}}^\top \underline{\mathbf{x}}' + b = 0$ and thus $d = \frac{|w^\top \underline{\mathbf{x}}^{(\alpha)} + b|}{\|\underline{\mathbf{w}}\|}$. Using the constraint yields $d \geq \frac{1}{\|\underline{\mathbf{w}}\|}$.

(b) The Lagrangian is

$$L(\underline{\mathbf{w}}, b, \underline{\lambda}, \underline{\xi}) := \tfrac{1}{2}\|\underline{\mathbf{w}}\|^2 + \tfrac{C}{p}\sum_{\alpha=1}^{p} \varphi_\alpha - \sum_{\alpha=1}^{p} \lambda_\alpha \left\{ y_T^{(\alpha)} \left( \underline{\mathbf{w}}^\top \underline{\mathbf{x}}^{(\alpha)} + b \right) - 1 + \varphi_\alpha \right\} - \sum_{\alpha=1}^{p} \xi_\alpha \varphi_\alpha\,.$$

The derivatives w.r.t. $\underline{\mathbf{w}}$ and $b$ remain th same, but for $\varphi_\alpha$ one gets:

$$\tfrac{\partial L}{\partial \varphi_\alpha} = \tfrac{C}{p} - \lambda_\alpha - \xi_\alpha \stackrel{!}{=} 0$$

Substituting $\xi_\alpha = \frac{C}{p} - \lambda_\alpha$ yields the Lagrangian for the separable case. However, since $\xi_\alpha \geq 0$ has to be fulfilled, one arrives at the additional contraint $\lambda_\alpha \leq \frac{C}{p}$.

## Exercise H8.2: C-SVM with standard parameters (homework, 3 points)

In this exercise, we use C-SVMs to solve the "XOR"-classification problem from exercise sheet 6. To this end (1) first create a *training set* of 80 data as described in exercise H6.1 and (2) create a *test set* of 80 data from the same distribution.

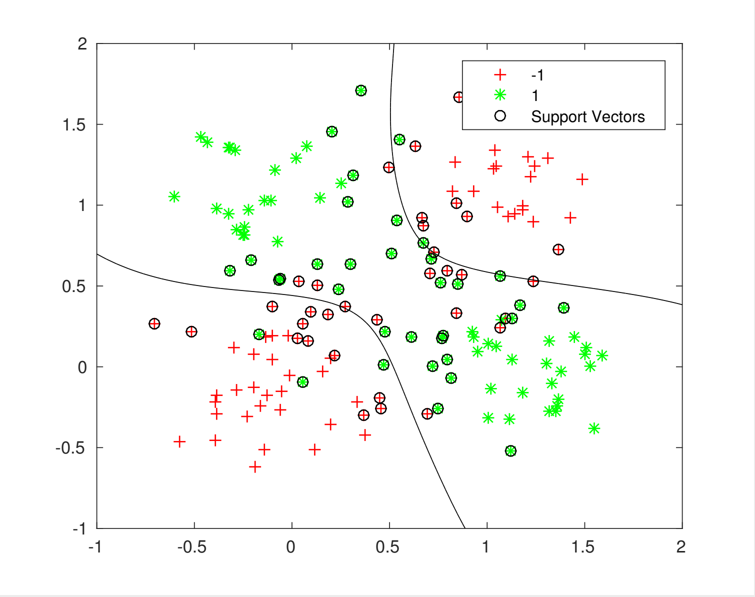
You can use existing software: `libsvm`[1] implements optimization routines (Matlab & Python) for SVMs. Alternatively, you can use the corresponding `scikit.learn` class[2]. For `R`, the package `e1071` implements SVM-optimization.

- Download, install, and familiarize yourself with `LIBSVM` or one of the other packages.
- Read the *Practical Guide to Support Vector Classification*[3] especially section 3.2 on *Cross-Validation*.

Next, use your chosen SVM implementation to train a C-SVM with RBF kernel and the software's standard parameters. Classify the test data and report the classification error quantified by the 0/1 loss function (percentage of wrong predictions). Visualize the results as in exercise H6.2: plot the training patterns and the decision boundary (e.g. with a contour plot) in input space.

**Solution**

SVM solution with 68 SV:




## Exercise H8.3: C-SVM parameter optimization (homework, 4 points)

(a) (2 points) Use cross-validation and grid-search to determine good values for $C$ and the kernel parameter $\gamma$. Follow the procedure described in the *guide*: Define the grid using exponentially growing sequences of $C$ and $\gamma$, e.g. $C \in \{2^{-6}, 2^{-4}, \ldots, 2^{10}\}$, $\gamma \in \{2^{-5}, 2^{-3}, \ldots, 2^{9}\}$. Make sure youonly use the training data in this step. Plot the mean training-set classification rate and cross-validation performance as a function of $C$ and $\gamma$ (e.g. using contour plots as in figure 2 of the *guide*).

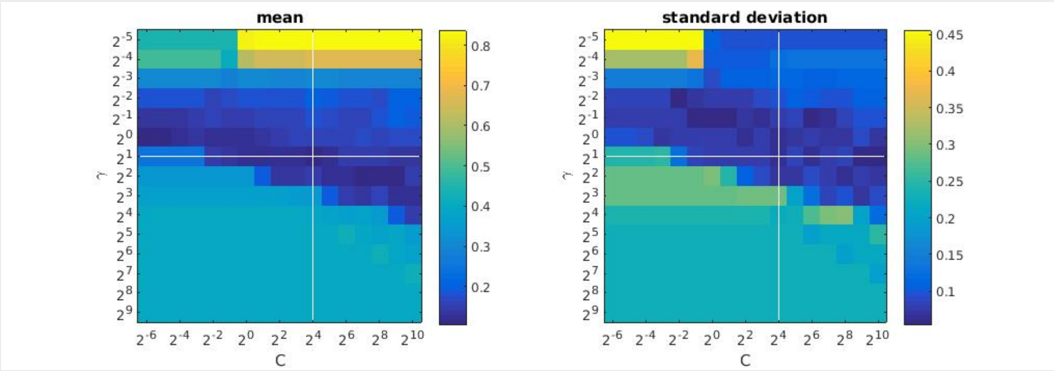
[1] `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`
[2] `http://tinyurl.com/lrpxw9k`
[3] `http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf`

(b) (1 point) Find the best combination of $C$ and $\gamma$ and train the RBF C-SVM on the *entire* training data, this time using these "optimal" parameters. Plot the results in the same way as in exercise H8.2.

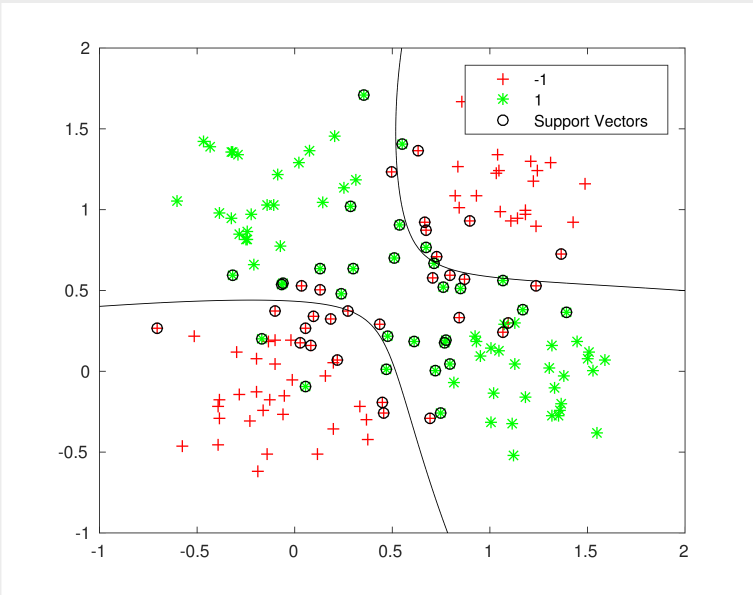
(c) (1 point) Compare the results with those obtained in H8.2, both in terms of statistics (e.g. classification performance, number of support vectors) and visually (e.g. signs of over- and under-fitting). What happens when you divide $C$ or $\gamma$ by 4?

**Solution**

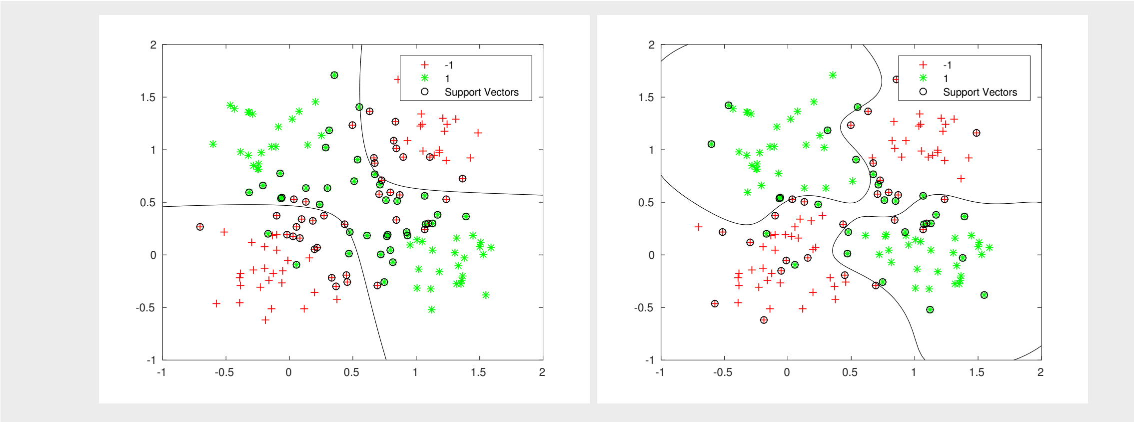
(a) Mean and stadard deviation over a nested 10-fold cross validation:




(b) SVM solution with $C = 16$ and $\gamma = 2$ with 57 SV:




(c) SVM solution with $C = 4$ and $\gamma = 2$ with 73 SV on the left and with $C = 16$ and $\gamma = 0.5$ with 53 SV on the right:




**Total 10 points.**