Machine Intelligence 1 — **Exercise Sheet 9**

WS 2017/18, Obermayer/Augustin — due: 17.01.2017 at 23:55

Support Vector Machines

Exercise T9.1: 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 (and VC dimension)?

(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?

(f) How can multi-class problems be solved with the C-SVM method?

(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:

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

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

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

Exercise H9.1: 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:

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

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

Exercise H9.2: C-SVM with standard parameters (homework, 3 points)

In this exercise, we use C-SVMs to solve the "XOR"-classification problem from exercise sheet 7. To this end (1) first create a *training set* of 80 data as described in exercise H7.1 and (2) create a *test set* of 80 data from the same distribution. [1]

You can use existing software: `libsvm`[2] implements optimization routines (Matlab & Python) for SVMs. Alternatively, you can use the corresponding `scikit.learn` class[3]. 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*[4] 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 H7.2: plot the training patterns and the decision boundary (e.g. with a contour plot) in input space. Additionally, highlight the support vectors.

Exercise H9.3: C-SVM parameter optimization (homework, 4 points)

[1] Since for SVMs typically labels $y_T \in \{-1, +1\}$ are used instead of $y_T \in \{0, 1\}$ it might be beneficial to use the former and not the latter.

[2] `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

[3] `http://tinyurl.com/lrpxw9k`

[4] `http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf`

(a) (2 points) Use cross-validation and grid-search to determine good values[5] for the C and the kernel parameter γ. Follow the procedure described in the *guide*: Define the grid using exponentially growing sequences of C and γ, 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 γ (e.g. using contour plots as in figure 2 of the *guide*).

(b) (1 point) Find the best combination of C and γ 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 H9.2.

(c) (1 point) Compare the results with those obtained in H9.2, 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 γ by 4?

Total 10 points.

[5]Note that $\gamma = 1/(2\sigma^2)$ in the lecture notation and that the C from the guide corresponds to C/p from the lecture; you do not have to use the lecture's notation but can stick to the (more standard) variant from the guide.