Machine Intelligence 1 **Exercise Sheet 9**

WS 2019/20, Obermayer/Kashef due: 16.01.2020 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?

The Lagrangian of the primal 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:

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

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

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

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

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

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

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

(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?

See tutorial notes.

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

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

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.

First, **construct your data**:

- Create a *training set* of 80 points as described in exercise H7.1,
  In the case of SVMs, binary labels are typically defined as $y_T \in \{-1, +1\}$ instead of $y_T \in \{0, 1\}$. Therefore, it might be beneficial to use the former and not the latter set.
- Create a *hold-out set* of 80 points from the same distribution.

Second, **the software**:

For constructing the C-SVM, you can use existing software that implements optimization routines for SVMs. Possible options include:

- `libsvm`[1] for Matlab & Python
- `scikit.learn` class[2] that wraps around the above `libsvm` package for Python.
- The package `e1071` implements SVM-optimization for `R`.

1. Download, install, and familiarize yourself with software package of your choice.

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

2. Read the *Practical Guide to Support Vector Classification*[3] especially section 3.2 on *Cross-Validation*.

Third, **train your first classifier and evaluate it**:

1. Use your chosen SVM implementation to train a C-SVM with RBF kernel and the software's standard parameters.
2. Classify the data from the hold-out set and report the classification error quantified by the 0-1 loss function (percentage of wrong predictions).

   Deliverables:

3. 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.
4. Highlight the support vectors in the aforementioned visualization.

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

(a) (2 points) Use cross-validation (e.g. 10-fold) and grid-search to determine good values[4] 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.

$$C \in \{2^{-6}, 2^{-4}, \ldots, 2^{10}\} \text{ and}$$

$$\gamma \in \{2^{-5}, 2^{-3}, \ldots, 2^{9}\}.$$

.
Make sure you only use the training data in this step. The hold-out set is completely excluded from the hyperparameter selection process and should only be used for estimating generlization performance.

Deliverables:

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*).

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

[4] 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.

(b) (1 point) Find the combination of C and γ that yield the best cross-validation performance in the previous step. Use this combination to train another RBF C-SVM on the *entire* training data.

Deliverables:
Plot the results in the same way as in exercise H9.2 using the data from the hold-out set.

(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 your best C **or** γ by 4?

Total 10 points.