Machine Learning Exercise Sheet 09

SVM and Kernels

In-Class

1 SVM

$y_i = f(x_i) + \xi_i$ $\quad y_i(w^T x_i + b) \geq 1 - \xi_i$ deal with noise

Problem 1: What is the connection between soft-margin SVM and logistic regression?

Problem 2: Consider a soft-margin SVM fitted to a linearly separable dataset $\mathcal{D}$ using the Hinge loss formulation of the optimization task.

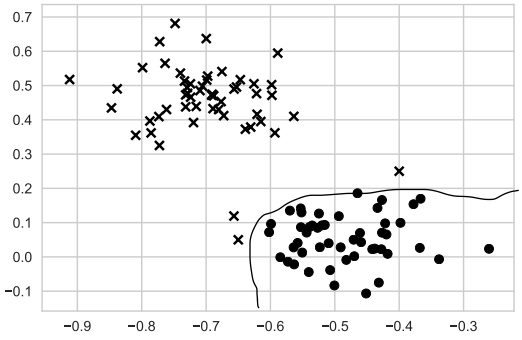
because of yes, ∪ linearly separable dataset

$$\text{minimize}_{\boldsymbol{w},b} \quad \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{N}\max\left(0, 1 - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)\right)$$

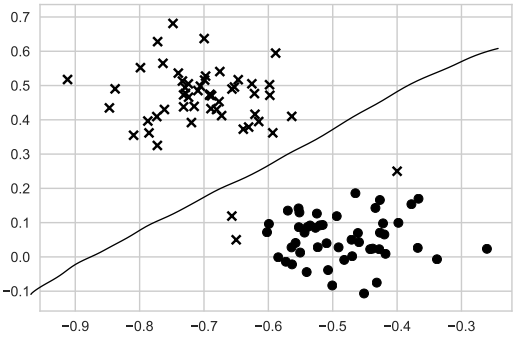
a) Is it guaranteed that all training samples in $\mathcal{D}$ will be assigned the correct label by the model?

b) Prove that if for some $C_0 \geq 0$ the resulting model classifies all training samples correctly and all samples lie outside of the margin then it will also be the case if we train the model with any larger $C > C_0$.

Problem 3: Sketch the decision boundary of an SVM with a quadratic kernel (polynomial with degree 2) for the data in the figure below, for two specified values of the penalty parameter C. The two classes are denoted as •'s and ×'s.



(a) $C = 10^{10}$



(b) $C = 10^{-10}$

1. Soft margin.

$$\min \quad f_0(w, b, \xi) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i$$

$$\text{sub to} \quad y_i (w^T x_i + b) - 1 + \xi_i \geq 0 \qquad i = 1, \ldots, N$$

$$\xi_i \geq 0 \qquad i = 1, \ldots, N$$

Loss Function

$$E(w, b, C) = \frac{1}{2C} w^T \cdot w + \sum_{i=1}^{N} \text{Hinge}\left(y_i \cdot (w^T x_i + b)\right)$$

λ

Loss Regretion

$$p(y_i = 1 \mid w, x) = \sigma(w^T x_i + b)$$

$$p(y_i = -1 \mid w, x) = \sigma(-(w^T x_i + b)) = 1 - \sigma(w^T x_i + b)$$

$$p(y \mid w, x) = \prod_{i=1}^{N} p(y_i \mid w, x_i) = \frac{1}{1 + e^{-(w^T x_i + b)}} \cdot \frac{1}{1 + e^{(w^T x_i + b)}}$$

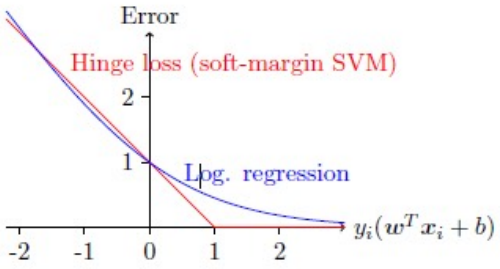
$$E(w, b) = -\ln p(y \mid w, x) = -\sum_{i=1}^{N} \ln\left(1 + e^{-y_i (w^T x_i + b)}\right)^{-1}$$

$$= \sum_{i=1}^{N} \ln\left(1 + e^{-y_i (w^T x_i + b)}\right)$$

② $$E(w, b, \lambda) = \lambda w^T \cdot w + \sum_{i=1}^{N} \ln\left(1 + e^{-y_i (w^T x_i + b)}\right)$$

① $$E(w, b, C) = \frac{1}{2C} w^T \cdot w + \sum_{i=1}^{N} \text{Hinge}\left(y_i \cdot (w^T x_i + b)\right)$$

Hence, we see the close relationship between the soft-margin SVM (Eq. 1) and logistic regression (Eq. 2). While soft-margin SVM uses the hinge function for its loss, logistic regression uses $\ln(1 + e^{-x})$. For better comparison, we can plot these two (with logistic regression rescaled by $\frac{1}{\ln 2}$):



2

a) no, because of misclassifying a few point nearly by the decision boundary will lead to significantly increasing margin.

large $C \rightarrow$ Hard SVM $\rightarrow$ less behaviour

b) let $h(w,b) = \sum_{i=1}^{N} \max(0, 1 - y_i(w^T x + b))$

when right classified $h(w,b) = 0$

$$(w^*, b^*) = \arg\min_{w,b} \frac{1}{2} w^T w + C_0 h(w,b)$$

$$(v^*, d^*) = \arg\min_{v,d} \frac{1}{2} w^T w + C h(w,b)$$

$$\frac{1}{2}\|w^*\|^2 + C_0 h(w^*, b^*) \leq \frac{1}{2}\|v^*\|^2 + C_0 h(v^*, d^*)$$

$$\frac{1}{2}\|v^*\| + C h(v^*, d^*) \leq \frac{1}{2}\|w^*\|^2 + C h(w^*, b^*)$$

$$+ \qquad +$$

$$(C_0 - C)(h(w^*, b^*) - h(v^*, d^*)) \leq 0$$

$\because C > C_0 \quad \therefore h(w^*, b^*) \leq h(v^*, d^*)$

That means the overall Hinge loss decreases for the optimal solution if we solve the SVM fitting problem with a larger value of C. Since we could separate the data perfectly (with no points within the margin) with the solution (w^*, b^*) for C_0 we have $h(w^*, b^*) = 0$. As shown above the Hinge loss can not get worse for larger C, hence also $h(v^*, d^*)$ has to be 0 and the solution (v^*, d^*) also corresponds to a decision boundary with no error on training data and no points lying inside the margin.

4. ① $k_G(\theta(x_1), \theta(x_2)) = \exp\left(-\frac{\|\theta(x_1) - \theta(x_2)\|^2}{2\sigma^2}\right)$

$= \exp\left(-\frac{\theta(x_1)^T\theta(x_1) - 2\theta(x_1)^T\theta(x_2) + \theta(x_2)^T\theta(x_2)}{2\sigma^2}\right)$

$= \exp\left(-\frac{k(x_1,x_1) - 2k(x_1,x_2) + k(x_2,x_2)}{2\sigma^2}\right)$

2 Kernels

Problem 4: Consider the Gaussian kernel

$$k_{\mathrm{G}}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp\left(-\frac{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|^2}{2\sigma^2}\right), \text{ with } \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d.$$

a) Suppose you have found a feature map $\theta : \mathbb{R}^d \to \mathbb{R}^{d_1}$ that transforms your data into a feature space in which a SVM with a Gaussian kernel works well. However, computing $\theta(\boldsymbol{x})$ is computationally expensive and luckily you discover an efficient method to compute the scalar product

$$k(\boldsymbol{x}_1, \boldsymbol{x}_2) = \theta(\boldsymbol{x}_1)^T \theta(\boldsymbol{x}_2)$$

in your feature space without having to compute $\theta(\boldsymbol{x}_1)$ and $\theta(\boldsymbol{x}_2)$ explicitly. Show how you can use the scalar product $k(\boldsymbol{x}_1, \boldsymbol{x}_2)$ to efficiently compute the Gaussian kernel $k_G(\theta(\boldsymbol{x}_1), \theta(\boldsymbol{x}_2))$ in your feature space.

b) One of the nice things about kernels is that new kernels can be constructed out of already given ones. Use the five kernel construction rules from the lecture to prove that k_G is a kernel.

Hint: Use the Taylor expansion of the exponential function to prove that $\exp \circ k_1$ *is a kernel if* k_1 *is a kernel. Also, consider* $k_2(\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2))$ *with the linear kernel* $k_2(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{x}_1^T \boldsymbol{x}_2$ *and a feature map* ϕ *with only one feature.*

c) Can any finite set of points be linearly separated in the feature space of the Gaussian kernel if σ can be chosen freely?

Problem 5: Let $\mathcal{M} = \cup_{n\in\mathbb{N}} \cup_{m\in\mathbb{N}} \mathbb{R}^{n\times m}$ denote the set of all real-valued matrices of arbitrary size. Prove that the function $k : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ is a valid kernel, where

$$k(\boldsymbol{X}, \boldsymbol{Y}) = \min(\mathrm{rank}(\boldsymbol{X}), \mathrm{rank}(\boldsymbol{Y})).$$

Construct a function

$\phi(x) = \{1, 1, 1, \ldots, 0, 0, 0\}$

$\phi^T(x)\,\phi(y) = \sum_{j=1}^{\infty} \phi_j(x)\,\phi_j(y) = \min(\mathrm{rank}(x), \mathrm{rank}(Y)) \in k, Y)$

4 (2) prove $\exp \circ k_1$ is a kernel if $k(x_1, x_2)$ is a kernel

Tyler expansion :

$$\exp(k(x_1, x_2)) = 1 + \sum_{n=1}^{\infty} \frac{1}{n!} k(x_1, x_2)^n$$

rule 3 $k_1(x_1, x_2) \cdot k_2(x_1, x_2)$

rule 2 $c \cdot k_1(x_1, x_2)$

rule 1 $k_1(x_1, x_2) + k_2(x_1, x_2)$

rule 4 $k(\phi(x_1), \phi(x_2))$

$\phi(z) = 1$

$$\exp(k_G(x_1, x_2)) = \exp\left(-\frac{\|x_1 - x_2\|^2}{2\sigma}\right) = \exp\left(-\frac{x_1^T x_1 - 2x_1^T x_2 + x_2^T x_2}{2\sigma}\right)$$

$$= \exp\left(-\frac{x_1^T x_1}{2\sigma}\right) \cdot \exp\left(\frac{x_1^T x_2}{\sigma}\right) \exp\left(-\frac{x_2^T x_2}{2\sigma}\right)$$

↑ use rule

Consider the last term first. The scalar product $\boldsymbol{x}_1^T \boldsymbol{x}_2$ is the linear kernel and by rule 2 the product $\boldsymbol{x}_1^T \boldsymbol{x}_2/\sigma^2$ is a kernel because σ^2 is positive. As proved above the term $\exp(\boldsymbol{x}_1^T \boldsymbol{x}_2/\sigma^2)$ is then a kernel as well.

The product of the first two terms $\exp(-\boldsymbol{x}_1^T \boldsymbol{x}_1/2\sigma^2) \exp(-\boldsymbol{x}_2^T \boldsymbol{x}_2/2\sigma^2)$ is a kernel by rule 4 with $k_3(x_1, x_2) = x_1 x_2$ and the feature map $\phi(\boldsymbol{z}) = \exp(-\boldsymbol{z}^T \boldsymbol{z}/2\sigma^2)$.

Finally, by rule 3 the product of the first two terms with the third term is a kernel.

---

(3) $k_G(x_1, x_2, \sigma) \xrightarrow{\sigma=0} k(x_1, x_2) = \begin{cases} 1 & x_1 = x_2 \\ 0 & x_1 \neq x_2 \end{cases}$

All training sample are correctly classified if $y_i(w^T \phi_G(x_i) + b) > 0$

$$y_i\left(\sum_j y_j \alpha_j k_G(x_i, x_j, \sigma) + b\right) > 0 \xrightarrow{\sigma=0} y_i^2 \alpha_i + y_i b$$

for small σ. By choosing $b = 0$ we see that the resulting condition is fulfilled for all training samples, since $y_i^2 = 1$ for all i and we can simply set all $\alpha_i > 0$ (note, that this means that every sample is a support vector in this case). Hence all finite sets of points can be linearly separated using the Gaussian kernel if the variance σ is chosen small enough.

Homework

3 SVM

Problem 6: Explain the similarities and differences between the SVM and perceptron algorithms. How do they perform classification? In what way do they differ?

Problem 7: Recall that the dual function in the setting of the SVM training task (Slide 17) can be written as

$$g(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{Q}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T \mathbf{1}_N \,.$$

(a) Write down the matrix $\boldsymbol{Q}$ using the vector of labels $\boldsymbol{y}$ and feature matrix $\boldsymbol{X}$. Denote the element-wise product between two matrices (in case you want to use it) by $\odot$ (also known as Hadamard product or Schur product).

(b) Prove that we can search for a *local* maximizer of g to find its *global* maximum (don't forget to prove properties of $\boldsymbol{Q}$ that you decide to use in this task).

Problem 8: Consider training a standard hard-margin SVM on a linearly separable training set of N samples. Let s denote the number of support vectors we would obtain if we would train on the entire dataset. Furthermore, let ε denote the leave-one-out cross validation (LOOCV) misclassification rate. Prove that the following relation holds:

$$\varepsilon \leq \frac{s}{N} \,.$$

Problem 9: Load the notebook `exercise_09_notebook.ipynb` from Moodle. Fill in the missing code and run the notebook. Convert the evaluated notebook to pdf and add it to the printout of your homework.

Note: We suggest that you use Anaconda for installing Python and Jupyter, as well as for managing packages. We recommend that you use Python 3.

For more information on Jupyter notebooks and how to convert them to other formats, consult the Jupyter documentation and nbconvert documentation.

4 Kernels

Problem 10: Show that for $N \in \mathbb{N}$ and $a_i \geq 0$ for $i = 0, \ldots, N$ the following function k is a valid kernel.

$$k(\boldsymbol{x}_1, \boldsymbol{x}_2) = \sum_{i=1}^{N} a_i \left(\boldsymbol{x}_1^T \boldsymbol{x}_2\right)^i + a_0, \text{ with } \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d \,.$$

Problem 11: Find the feature transformation $\boldsymbol{\phi}(x)$ corresponding to the kernel

$$k(x_1, x_2) = \frac{1}{1 - x_1 x_2}, \text{ with } x_1, x_2 \in (0, 1) \,.$$

Hint: Consider an infinite-dimensional feature space and infinite series.

10

$$k(\boldsymbol{x}_1, \boldsymbol{x}_2) = \left(\sum_{i=1}^{N} a_i \left(\boldsymbol{x}_1^T \boldsymbol{x}_2\right)^2\right) + a_0, \text{ with } \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d.$$

rule 3 ↓ (on $\boldsymbol{x}_1^T \boldsymbol{x}_2$)

rule 3 ↙ (on the exponent 2)

rule 1 ↑ (on $+ a_0$)

rule ↑ (on $\sum$)

$a_i \geq 0$ ↑ rule 2

$k_1 = x_1^T x_2$ is kernel because of scale product of vector

$k_c(x_1, x_2)$ is kernel, because we can define feature map $\phi(x) = \sqrt{a_0}$

$\phi^T(x_1)\,\phi(x_2) = a_0$

---

11 geometric transformation

$$k(x_1, x_2) = \frac{1}{1 - x_1 x_2} = \sum_{0}^{\infty} x_1^i x_2^i = \phi^T(x_1)\,\phi(x_2)$$

$$\phi(x) = (1, x, x^2, \cdots x^n)^T$$

7

$$g(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i y_i y_j x_i^T x_j \alpha_j$$

$$Q = -y_i y_j x_i^T x_j$$

$$A = y_i \cdot y_j \quad B = x_i^T x_j \qquad y \in (N \times 1)$$

$$A = \begin{pmatrix} y_1 y_1 & y_1 y_2 & \cdots & y_1 y_N \\ \vdots & & & \\ y_N y_1 & \cdots & & y_N y_N \end{pmatrix} = y y^T \qquad B = \begin{pmatrix} x_1^T x_1 & \cdots & x_1^T x_N \\ \vdots & & \vdots \\ x_N^T x_1 & \cdots & x_N^T x_N \end{pmatrix} = X \cdot X^T$$

$$Q = -yy^T \odot X \cdot X^T$$

(b) $$\alpha^T (A \odot B) \alpha = \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i y_i y_j x_i^T x_j \alpha_j$$

$$= \left(\sum_{i=1}^{N} \alpha_i y_i x_i \right)^T \left(\sum_{j=1}^{N} \alpha_j y_j x_j \right)$$

$$= \left\| \sum_{i=1}^{N} \alpha_i y_i x_i \right\|^2 \geq 0$$

$\therefore A \odot B$ is PSD $\rightarrow Q = -A \odot B$ is negative semi-definite

$\Rightarrow$ Function g is concave, every local maximum is global maximum

---

8 for support vector $\alpha_i^* f_i(\theta^*) = 0$

$\alpha_i^* \neq 0$

for training set

$\alpha_i^* = 0 \Rightarrow w^*$ and b^* will not change

直观地讲，这个结果来自于下面的主张（我们在下面证明）：如果在整个训练集上进行训练时，xi不是支持向量，那么当把xi从训练集中排除时，最优的w∗和b∗不会改变。训练集之外的xi不会改变。
由于原始数据是线性可分离的，而且我们使用的是硬边际分类器，所以 由原始w∗和b∗给出的假设不会在xi上出错，因此，在第i步中也不会出错。因此，在LOOCV的第i步中不会出现错误。等价地，LOOCV程序中唯一可能的错误 是在整个训练集上训练时对支持向量的xi产生的，因此ε ≤ s N .

Intuitively, the result follows from the following claim (which we prove below): if $\boldsymbol{x}_i$ is not a support vector when training on the entire training set, then the optimal $\boldsymbol{w}^*$ and b^* do not change when leaving $\boldsymbol{x}_i$ out of the training set.

Since the original data are linearly separable and since we are using a hard-margin classifier, the hypothesis given by the original $\boldsymbol{w}^*$ and b^* will not make an error on $\boldsymbol{x}_i$, and hence, no error will be made in the i-th step of the LOOCV. Equivalently, the only *possible* errors in the LOOCV procedure are made on $\boldsymbol{x}_i$'s which are support vectors when training on the entire training set, and hence $\varepsilon \leq \frac{s}{N}$.

(The following details are not required for full points).

Formally, let $(\boldsymbol{w}^*_{\mathcal{D}}, b^*_{\mathcal{D}})$ and $\boldsymbol{\alpha}^*_{\mathcal{D}}$ denote the optimal primal and dual solutions for the SVM when training on the entire $\mathcal{D}$. Also let, $\mathcal{D}_i = \mathcal{D} \backslash \{(\boldsymbol{x}_i, y_i)\}$ be the set of training examples when omitting the i-th example, and let $(\boldsymbol{w}^*_{\mathcal{D}_i}, b^*_{\mathcal{D}_i})$ and $\boldsymbol{\alpha}^*_{\mathcal{D}_i}$ be the optimal primal and dual variables of the optimization problem when training on $\mathcal{D}_i$.

$\boldsymbol{\alpha}_{\mathcal{D}_i}$ consists of only $n-1$ variables, namely $\boldsymbol{\alpha}_{\mathcal{D}_i,1}, \ldots, \boldsymbol{\alpha}_{\mathcal{D}_i,i-1}, \boldsymbol{\alpha}_{\mathcal{D}_i,i+1}, \ldots, \boldsymbol{\alpha}_{\mathcal{D}_i,N}$. Now consider setting the dual variables as follows $\boldsymbol{\alpha}_{\mathcal{D}_i,j} = \boldsymbol{\alpha}^*_{\mathcal{D},j}$ for $j \neq i$. Note that, if $\boldsymbol{x}_i$ is not a support vector when training on $\mathcal{D}$ then $\boldsymbol{\alpha}^*_{\mathcal{D},i} = 0$. We can verify that $(\boldsymbol{w}^*_{\mathcal{D}}, b^*_{\mathcal{D}})$ and $\boldsymbol{\alpha}_{\mathcal{D}_i}$ satisfy the KKT conditions for the SVM optimization problem when training on $\mathcal{D}_i$ (e.g. the condition regarding the derivatives of the Lagrangian with respect to the primal variables is guaranteed to hold by our construction). From this, and the fact that $\boldsymbol{w}^*_{\mathcal{D}}, b^*_{\mathcal{D}}$ are unique since the objective function is strictly convex we can conclude that $\boldsymbol{w}^*, b^*$ do not change when omitting $\{(\boldsymbol{x}_i, y_i)\}$ as desired.