Machine Learning Exercise Sheet 09

SVM and Kernels

Important

In-Class

1 SVM

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

The soft-margin SVM is defined via the minimization problem

$$\begin{aligned} &\text{minimize} && f_0(\boldsymbol{w}, b, \boldsymbol{\xi}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{N}\xi_i && \\ &\text{subject to} && y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - 1 + \xi_i \geq 0 && i = 1, \dots, N\,, \\ & && \xi_i \geq 0 && i = 1, \dots, N\,. \end{aligned}$$

Due to the complementary slackness conditions, we have

$$\alpha_i\left(y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - 1 + \xi_i\right) = 0\,.$$

Thus, for points that lie inside or beyond the margin (i.e. where $\alpha_i > 0$ and $y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) < 1$), it must hold that $\xi_i = 1 - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)$. Otherwise, ξ_i is minimized and therefore 0. In other words,

$$\xi_i = \begin{cases} 1 - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) & \text{if } y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) < 1, \\ 0 & \text{otherwise.} \end{cases} = \max\left(0, 1 - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)\right) = \text{Hinge}\big(y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)\big)\,.$$

Dividing by C, the error (or loss) function is

$$E(\boldsymbol{w}, b, C) = \underbrace{\frac{1}{2C}}_{\lambda}\boldsymbol{w}^T\boldsymbol{w} + \sum_{i=1}^{N}\text{Hinge}\big(y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)\big) \qquad (1)$$

Applying logistic regression to the same classification problem, we have

$$\begin{aligned} p(y_i = 1|\boldsymbol{x}_i, \boldsymbol{w}) &= \sigma(\boldsymbol{w}^T\boldsymbol{x}_i + b)\,, \\ p(y_i = -1|\boldsymbol{x}_i, \boldsymbol{w}) &= \sigma(-(\boldsymbol{w}^T\boldsymbol{x}_i + b)) = 1 - \sigma(\boldsymbol{w}^T\boldsymbol{x}_i + b)\,, \end{aligned}$$

where the last equality holds due to the definition of the sigmoid function, $\sigma(z) = \frac{1}{1+e^{-z}}$. Keep in mind that in this case $y_i \in \{-1, 1\}$ and not $\{0, 1\}$ as we used originally. Altogether, we have

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}) = \prod_{i=1}^{N}\sigma(y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b))\,.$$

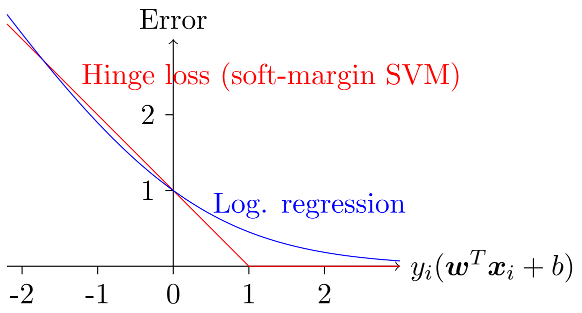
We use the negative log-likelihood as the error function, i.e.

$$E(\boldsymbol{w}, b) = -\ln(p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w})) = -\sum_{i=1}^{N} \ln((1 + e^{-y_i(\boldsymbol{w}^T\boldsymbol{x}_i+b)})^{-1}) = \sum_{i=1}^{N} \ln(1 + e^{-y_i(\boldsymbol{w}^T\boldsymbol{x}_i+b)})$$

Additionally, we introduce an L_2 regularization term and get

$$E(\boldsymbol{w}, b, \lambda) = \lambda \boldsymbol{w}^T\boldsymbol{w} + \sum_{i=1}^{N} \ln(1 + e^{-y_i(\boldsymbol{w}^T\boldsymbol{x}_i+b)}) \tag{2}$$

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 $\frac{1}{\ln 2}$):



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

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

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 $C > C_0$.

a) No, as it might happen that misclassifying a few points that are very close to the decision boundary would lead to a significantly increasing the margin. In general, larger values of C make this behavior less likely.

b) Denote by $h(\boldsymbol{w}, b) = \sum_{i=1}^{N} \max\left(0, 1 - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)\right)$ the overall Hinge loss and let $C > C_0 \geq 0$ be as in the task formulation. Note that the hyperplane defined by $\boldsymbol{w}$ and b assigns the correct labels to all samples such that all of them also lie outside of the margin if and only if $h(\boldsymbol{w}, b) = 0$.

We also define the corresponding solutions as

$$(\boldsymbol{w}^*, b^*) = \arg\min_{\boldsymbol{w},b} \quad \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C_0 h(\boldsymbol{w}, b) \quad \text{and}$$
$$(\boldsymbol{v}^*, d^*) = \arg\min_{\boldsymbol{w},b} \quad \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C h(\boldsymbol{w}, b)\,.$$

$\ast$ Then the following inequality holds due to optimality of these points:

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

fixing C_0

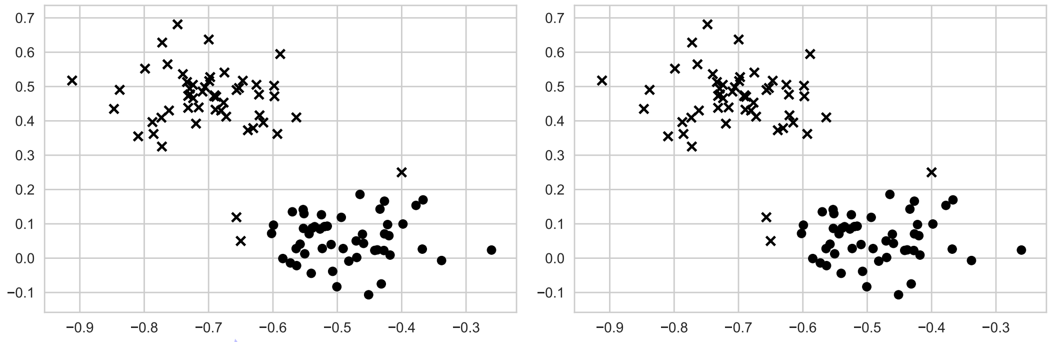
fixing C

Summing the left and right hand side of these inequalities we get

$$\frac{1}{2}\|\boldsymbol{w}^*\|^2 + \frac{1}{2}\|\boldsymbol{v}^*\|^2 + C_0 h(\boldsymbol{w}^*, b^*) + C h(\boldsymbol{v}^*, d^*) \le \frac{1}{2}\|\boldsymbol{v}^*\|^2 + \frac{1}{2}\|\boldsymbol{w}^*\|^2 + C_0 h(\boldsymbol{v}^*, d^*) + C h(\boldsymbol{w}^*, b^*)$$
$$\Leftrightarrow$$
$$(C - C_0)(h(\boldsymbol{v}^*, d^*) - h(\boldsymbol{w}^*, b^*)) \le 0 \quad \text{and since } C - C_0 > 0$$
$$\Rightarrow$$
$$h(\boldsymbol{v}^*, d^*) \le h(\boldsymbol{w}^*, b^*)\,.$$

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 $(\boldsymbol{w}^*, b^*)$ for C_0 we have $h(\boldsymbol{w}^*, b^*) = 0$. As shown above the Hinge loss can not get worse for larger C, hence also $h(\boldsymbol{v}^*, d^*)$ has to be 0 and the solution $(\boldsymbol{v}^*, d^*)$ also corresponds to a decision boundary with no error on training data and no points lying inside the margin.

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 $\bullet$'s and $\times$'s.

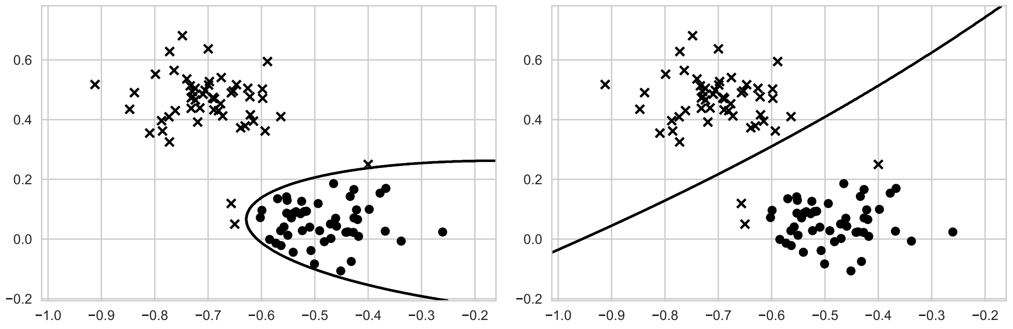

(a) $C = 10^{10}$ (b) $C = 10^{-10}$

large C

$\Rightarrow$ tighter margin

Small C

$\Rightarrow$ relaxed margin


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

a) With such a large penalty SVM will try to correctly classify all of the instances in the training set.

b) Given the small penalty, we can allow few misclassified instances, and obtain a larger margin between the two classes.

2 Kernels

Problem 4: Consider the Gaussian kernel

Important

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

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?

a) By expanding the quadratic term and applying the definition of $k(\boldsymbol{x}_1, \boldsymbol{x}_2)$ we get

$$\begin{aligned}
k_{\mathrm{G}}(\theta(\boldsymbol{x}_1), \theta(\boldsymbol{x}_2)) &= \exp\left(-\frac{\|\theta(\boldsymbol{x}_1) - \theta(\boldsymbol{x}_2)\|^2}{2\sigma^2}\right) \\
&= \exp\left(-\frac{(\theta(\boldsymbol{x}_1) - \theta(\boldsymbol{x}_2))^T(\theta(\boldsymbol{x}_1) - \theta(\boldsymbol{x}_2))}{2\sigma^2}\right) \\
&= \exp\left(-\frac{\theta(\boldsymbol{x}_1)^T\theta(\boldsymbol{x}_1) - 2\theta(\boldsymbol{x}_1)^T\theta(\boldsymbol{x}_2) + \theta(\boldsymbol{x}_2)^T\theta(\boldsymbol{x}_2)}{2\sigma^2}\right) \\
&= \exp\left(-\frac{k(\boldsymbol{x}_1, \boldsymbol{x}_1) - 2k(\boldsymbol{x}_1, \boldsymbol{x}_2) + k(\boldsymbol{x}_2, \boldsymbol{x}_2)}{2\sigma^2}\right) .
\end{aligned}$$

Thus we can compute $k_{\mathrm{G}}(\theta(\boldsymbol{x}_1), \theta(\boldsymbol{x}_2))$ from $k(\boldsymbol{x}_1, \boldsymbol{x}_2)$ using the above equation.

b) First we prove that $\exp(k(\boldsymbol{x}_1, \boldsymbol{x}_2))$ is a kernel if $k(\boldsymbol{x}_1, \boldsymbol{x}_2)$ is a kernel. The Taylor expansion of the exponential function is

$$\exp(k(\boldsymbol{x}_1, \boldsymbol{x}_2)) = 1 + \sum_{n=1}^{\infty} \frac{1}{n!} k(\boldsymbol{x}_1, \boldsymbol{x}_2)^n .$$

The power $k(\boldsymbol{x}_1, \boldsymbol{x}_2)^n$ is a kernel by iterated application of rule 3 ($k_1(\boldsymbol{x}_1, \boldsymbol{x}_2)k_2(\boldsymbol{x}_1, \boldsymbol{x}_2)$ is a kernel). The product $(1/n!)k(\boldsymbol{x}_1, \boldsymbol{x}_2)^n$ is a kernel by rule 2 ($\alpha k_1(\boldsymbol{x}_1, \boldsymbol{x}_2)$ if a kernel for $\alpha > 0$) because $(1/n!)$ is always positive. The sum $\sum_{n=1}^{N} 1/(n!)k(\boldsymbol{x}_1, \boldsymbol{x}_2)^n$ is a kernel for arbitrary $N \in \mathbb{N}$ by iterated application of rule 1 ($k_1(\boldsymbol{x}_1, \boldsymbol{x}_2) + k_2(\boldsymbol{x}_1, \boldsymbol{x}_2)$ is a kernel). Therefore, the pointwise limit of kernels

$$\sum_{n=1}^{\infty} \frac{1}{n!} k(\boldsymbol{x}_1, \boldsymbol{x}_2)^n = \lim_{N\to\infty} \sum_{n=1}^{N} \frac{1}{n!} k(\boldsymbol{x}_1, \boldsymbol{x}_2)^n$$

is again a kernel. The constant 1 is a kernel by rule 4: $k_3(\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2))$ with $k_3(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{x}_1^T \boldsymbol{x}_2$ and $\phi(\boldsymbol{z}) = 1$. Thus $1 + \sum_{n=1}^{\infty} \frac{1}{n!} k(\boldsymbol{x}_1, \boldsymbol{x}_2)^n$ is a kernel by rule 1.

We expand the argument of the exponential function,

$$\begin{aligned}
\exp\left(-\frac{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|^2}{2\sigma^2}\right) &= \exp\left(-\frac{(\boldsymbol{x}_1 - \boldsymbol{x}_2)^T(\boldsymbol{x}_1 - \boldsymbol{x}_2)}{2\sigma^2}\right) = \exp\left(-\frac{\boldsymbol{x}_1^T\boldsymbol{x}_1 - 2\boldsymbol{x}_1^T\boldsymbol{x}_2 + \boldsymbol{x}_2^T\boldsymbol{x}_2}{2\sigma^2}\right) \\
&= \exp\left(-\frac{\boldsymbol{x}_1^T\boldsymbol{x}_1}{2\sigma^2}\right) \exp\left(-\frac{\boldsymbol{x}_2^T\boldsymbol{x}_2}{2\sigma^2}\right) \exp\left(\frac{\boldsymbol{x}_1^T\boldsymbol{x}_2}{\sigma^2}\right) .
\end{aligned}$$

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.

b) Consider the limit $\sigma \to 0$. Then

$$k_G(\boldsymbol{x}_1, \boldsymbol{x}_2; \sigma) \xrightarrow{\sigma \to 0} k(x_1, x_2) = \begin{cases} 1 & \text{if } x_1 = x_2 \\ 0 & \text{if } x_1 \neq x_2 \end{cases}.$$

All training samples are correctly classified if

i.e Zero loss

$$y_i(\boldsymbol{w}^T \phi_G(\boldsymbol{x}_i) + b) > 0 \quad \text{for all } i\,.$$

We will construct a classifier with the desired property by using the dual representation of $\boldsymbol{w} = \sum_j y_j \alpha_j \phi_G(\boldsymbol{x}_j)$. Substituting it into the above expression and replacing the scalar product $\phi_G(\boldsymbol{x}_i)^T \phi_G(\boldsymbol{x}_j)$ by the kernel function $k_G(\boldsymbol{x}_i, \boldsymbol{x}_j; \sigma)$ we get the same condition in the following form

$$y_i \left(\sum_j y_j \alpha_j k_G(\boldsymbol{x}_i, \boldsymbol{x}_j; \sigma) + b \right) > 0\,.$$

Using the limit kernel function k in particular we can see that the left hand side will converge to

$$\xrightarrow{\sigma \to 0} {y_i}^2 \alpha_i + y_i b$$

$\alpha_i = 0$ for correctly classified

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.

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

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

We will prove it by constructing the corresponding feature map. Consider the function

$$\phi(\boldsymbol{X}) = (\underbrace{1, 1, 1, ..., 1}_{\text{rank}(\boldsymbol{X}) \text{ ones}}, \underbrace{0, 0, 0, 0, 0, 0, 0, 0...}_{\text{zeros everywhere else}})\,.$$

Then $\phi(\boldsymbol{X})^T \phi(\boldsymbol{Y}) = \sum_{j=1}^{\infty} \phi_j(\boldsymbol{X}) \phi_j(\boldsymbol{Y}) = \min(\text{rank}(\boldsymbol{X}), \text{rank}(\boldsymbol{Y})) = k(\boldsymbol{X}, \boldsymbol{Y})$ is the inner product. Hence, $k(\boldsymbol{X}, \boldsymbol{Y})$ is a valid kernel.

Homework

3 SVM

Problem 6: Explain the similarities and differences between the SVM and perceptron algorithms.

Both supervised classification methods separate two classes by a hyperplane. The difference is that SVM also tries to maximize the margin between the hyperplane and data, while perceptron only cares about separation.

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).

(a) Recall that the dual function g is

$$g(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j \alpha_j$$

and on the other hand

$$\frac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{Q}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T \mathbf{1}_N = \sum_{i=1}^{N} \alpha_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i Q_{ij} \alpha_j$$

meaning that $Q_{ij} = -y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j$. From that we see that $\boldsymbol{Q}$ can be decomposed as (-1 times) the element-wise product of two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ with $A_{ij} = y_i y_j$ and $B_{ij} = \boldsymbol{x}_i^T \boldsymbol{x}_j$ so that

$$\boldsymbol{A} = \begin{pmatrix} y_1y_1 & y_1y_2 & \cdots & y_1y_N \\ y_2y_1 & y_2y_2 & \cdots & y_2y_N \\ \vdots & \vdots & & \vdots \\ y_Ny_1 & y_Ny_2 & \cdots & y_Ny_N \end{pmatrix} = \boldsymbol{y}\boldsymbol{y}^T \text{ and } \boldsymbol{B} = \begin{pmatrix} \boldsymbol{x}_1^T\boldsymbol{x}_1 & \boldsymbol{x}_1^T\boldsymbol{x}_2 & \cdots & \boldsymbol{x}_1^T\boldsymbol{x}_N \\ \boldsymbol{x}_2^T\boldsymbol{x}_1 & \boldsymbol{x}_2^T\boldsymbol{x}_2 & \cdots & \boldsymbol{x}_2^T\boldsymbol{x}_N \\ \vdots & \vdots & & \vdots \\ \boldsymbol{x}_N^T\boldsymbol{x}_1 & \boldsymbol{x}_N^T\boldsymbol{x}_2 & \cdots & \boldsymbol{x}_N^T\boldsymbol{x}_N \end{pmatrix} = \boldsymbol{X}\boldsymbol{X}^T.$$

Finally, we get $\boldsymbol{Q} = -\boldsymbol{y}\boldsymbol{y}^T \odot \boldsymbol{X}\boldsymbol{X}^T$.

(b) We will first show that $\boldsymbol{A} \odot \boldsymbol{B} = \boldsymbol{y}\boldsymbol{y}^T \odot \boldsymbol{X}\boldsymbol{X}^T \in \mathbb{R}^{N \times N}$ is a positive semi-definite matrix by definition. For that consider an arbitrary $\boldsymbol{\alpha} \in \mathbb{R}^N$ and

$$\begin{aligned}\boldsymbol{\alpha}^T(\boldsymbol{A} \odot \boldsymbol{B})\boldsymbol{\alpha} &= \sum_{i=1}^N \sum_{j=1}^N \alpha_i y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j \alpha_j \\ &= \sum_{i=1}^N \sum_{j=1}^N (\alpha_i y_i \boldsymbol{x}_i^T)(\alpha_j y_j \boldsymbol{x}_j) = \left(\sum_{i=1}^N \alpha_i y_i \boldsymbol{x}_i\right)^T \left(\sum_{j=1}^N \alpha_j y_j \boldsymbol{x}_j\right) = \left\| \sum_{i=1}^N \alpha_i y_i \boldsymbol{x}_i \right\|^2 \geq 0\,.\end{aligned}$$

✗ As $\boldsymbol{A} \odot \boldsymbol{B}$ is PSD, it follows that $\boldsymbol{Q} = -\boldsymbol{A} \odot \boldsymbol{B}$ is negative semi-definite. This means that the function g is **concave**, thus every local maximum is a global maximum.

Important

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:

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

✗ 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.

Problem 9: Load the notebook `09_homework_svm_kernels.ipynb` from Piazza. 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.

The solution notebook is uploaded on Piazza.

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.

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

The function $k_l(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{x}_1^T \boldsymbol{x}_2$ is a kernel because it is the scalar product of the input vectors. The constant function $k_c(\boldsymbol{x}_1, \boldsymbol{x}_2) = a_0 \geq 0$ is a kernel because we can define the feature map $\phi(\boldsymbol{x}) = \sqrt{a_0}$ and obtain this kernel by calculating the scalar product in feature space $\phi(\boldsymbol{x}_1)^T \phi(\boldsymbol{x}_2) = \sqrt{a_0}^2 = a_0$. Overall, k is a kernel since it is built up of sums and products of kernels and multiplication with non-negative scalars.

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.

We use the geometric series to transform k:

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

with the feature transformation

$$\boldsymbol{\phi}(x) = \left(1, x, x^2, x^3, x^4, \ldots\right)^T .$$