Practical Session 7

Soft-Margin SVM and Kernels

1 Soft-Margin 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,

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

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

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

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

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

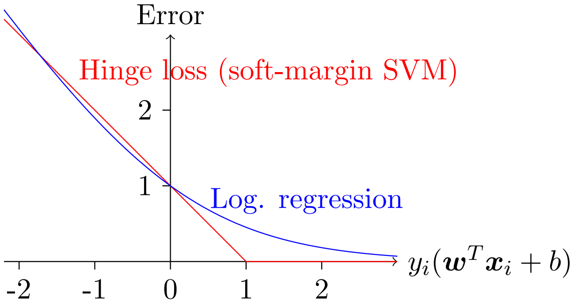
We use the negative log-likelihood as the error function, i.e.

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

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



2 Gaussian kernel

Problem 2: 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 the function

$$K(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{|\boldsymbol{x} - \boldsymbol{y}|^2}{2\sigma^2}\right)$$

is a kernel.

(Hint: Use the Taylor expansion of the exponential function to prove that $\exp(K_1(\boldsymbol{x}, \boldsymbol{y}))$ is a kernel if $K_1(\boldsymbol{x}, \boldsymbol{y})$ is a kernel.)

2nd hint: It might help to apply the rule $K_3(\boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{y}))$ with the linear kernel $K_3(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^T \boldsymbol{y}$ and consider a feature map $\phi(\boldsymbol{z})$ with only one feature.

First we prove that $\exp(K_1(\boldsymbol{x}, \boldsymbol{y}))$ is a kernel if $K_1(\boldsymbol{x}, \boldsymbol{y})$ is a kernel. The Taylor expansion of the exponential function is

$$\exp(K_1(\boldsymbol{x}, \boldsymbol{y})) = 1 + \sum_{n=1}^{\infty} \frac{1}{n!} K_1(\boldsymbol{x}, \boldsymbol{y})^n \,.$$

The power $K_1(\boldsymbol{x}, \boldsymbol{y})^n$ is a kernel by iterated application of rule 3 ($K_1(\boldsymbol{x}, \boldsymbol{y}) K_2(\boldsymbol{x}, \boldsymbol{y})$ is a kernel). The product $(1/n!) K_1(\boldsymbol{x}, \boldsymbol{y})^n$ is a kernel by rule 2 ($\alpha K_1(\boldsymbol{x}, \boldsymbol{y})$ if a kernel for $\alpha > 0$) because $(1/n!)$ is always positive. The sum $\sum_{n=1}^{\infty} 1/(n!) K_1(\boldsymbol{x}, \boldsymbol{y})^n$ is a kernel by iterated application of rule 1 ($K_1(\boldsymbol{x}, \boldsymbol{y}) + K_2(\boldsymbol{x}, \boldsymbol{y})$ is a kernel). The constant 1 is a kernel by rule 4 ($K_3(\boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{y}))$) with $K_3(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^T \boldsymbol{y}$ and $\phi(\boldsymbol{z}) = (1)$. Thus $1 + \sum_{n=1}^{\infty} \frac{1}{n!} K_1(\boldsymbol{x}, \boldsymbol{y})^n$ is a kernel by rule 1.

We expand the argument of the exponential function,

$$\exp\left(-\frac{|\boldsymbol{x}-\boldsymbol{y}|^2}{2\sigma^2}\right) = \exp\left(-\frac{(\boldsymbol{x}-\boldsymbol{y})^T(\boldsymbol{x}-\boldsymbol{y})}{2\sigma^2}\right) = \exp\left(-\frac{\boldsymbol{x}^T\boldsymbol{x} - 2\boldsymbol{x}^T\boldsymbol{y} + \boldsymbol{y}^T\boldsymbol{y}}{2\sigma^2}\right)$$
$$= \exp\left(-\frac{\boldsymbol{x}^T\boldsymbol{x}}{2\sigma^2}\right)\exp\left(-\frac{\boldsymbol{y}^T\boldsymbol{y}}{2\sigma^2}\right)\exp\left(\frac{\boldsymbol{x}^T\boldsymbol{y}}{\sigma^2}\right) .$$

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

The product of the first two terms $\exp(-\boldsymbol{x}^T\boldsymbol{x}/2\sigma^2)\exp(-\boldsymbol{y}^T\boldsymbol{y}/2\sigma^2)$ is a kernel by rule 4 with $K_3(x,y) = xy$ and the one-dimensional 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 Stacking feature maps

Suppose you have found a feature map $\boldsymbol{\theta} : \mathbb{R}^n \to \mathbb{R}^m$ that transforms your data into a feature space in which a SVM with a Gaussian kernel works well. However computing the feature map $\boldsymbol{\theta}(\boldsymbol{x})$ is computationally expensive and luckily you discover an efficient method to compute the scalar product $K(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{\theta}(\boldsymbol{x})^T\boldsymbol{\theta}(\boldsymbol{y})$ in your feature space without having to compute $\boldsymbol{\theta}(\boldsymbol{x})$ and $\boldsymbol{\theta}(\boldsymbol{y})$ explicitly.

Problem 3: Show how you can use the scalar product $K(\boldsymbol{x},\boldsymbol{y})$ to efficiently compute the Gaussian kernel in your feature space, that is

$$K_g(\boldsymbol{\theta}(\boldsymbol{x}),\boldsymbol{\theta}(\boldsymbol{y}))$$

where

$$K_g(\boldsymbol{a},\boldsymbol{b}) = \exp\left(-\frac{|\boldsymbol{a}-\boldsymbol{b}|^2}{2\sigma^2}\right)$$

is the Gaussian kernel.

By expanding the quadratic term and applying the definition of the $K(\boldsymbol{x},\boldsymbol{y})$ we get

$$\begin{aligned}
K_g(\boldsymbol{\theta}(\boldsymbol{x}),\boldsymbol{\theta}(\boldsymbol{y})) &= \exp\left(-\frac{|\boldsymbol{\theta}(\boldsymbol{x})-\boldsymbol{\theta}(\boldsymbol{y})|^2}{2\sigma^2}\right)\\
&= \exp\left(-\frac{(\boldsymbol{\theta}(\boldsymbol{x})-\boldsymbol{\theta}(\boldsymbol{y}))^T(\boldsymbol{\theta}(\boldsymbol{x})-\boldsymbol{\theta}(\boldsymbol{y}))}{2\sigma^2}\right)\\
&= \exp\left(-\frac{\boldsymbol{\theta}(\boldsymbol{x})^T\boldsymbol{\theta}(\boldsymbol{x}) - 2\boldsymbol{\theta}(\boldsymbol{x})^T\boldsymbol{\theta}(\boldsymbol{y}) + \boldsymbol{\theta}(\boldsymbol{y})^T\boldsymbol{\theta}(\boldsymbol{y})}{2\sigma^2}\right)\\
&= \exp\left(-\frac{K(\boldsymbol{x},\boldsymbol{x}) - 2K(\boldsymbol{x},\boldsymbol{y}) + K(\boldsymbol{y},\boldsymbol{y})}{2\sigma^2}\right) .
\end{aligned}$$

Thus we can compute $K_g(\boldsymbol{\theta}(\boldsymbol{x}),\boldsymbol{\theta}(\boldsymbol{y}))$ from $K(\boldsymbol{x},\boldsymbol{y})$ using the above equation.

4 Unsuitable Kernels

Consider a SVM without slack variables using the kernel $K(x, y) = |x||y|$ where $x, y \in \mathbb{R}$.

Problem 4: Show that the function $K(x, y)$ is indeed a kernel and write down a feature space $\phi(x)$ corresponding to this kernel.

We see that $\phi(x) = |x|$ is the corresponding feature space, since $\phi(x)^T\phi(y) = |x||y| = K(x, y)$. Since we managed to express the $K(x, y)$ as a scalar product this function is indeed a kernel.

Problem 5: Write down the set of classification functions that can be learned by an SVM using the kernel $K(x, y)$. Make sure that the set contains each classification function only once.

An SVM is a linear classifier. Thus from the feature space $\phi(x)$ it can be seen that the classification functions have the from

$$h_b(x) = \operatorname{sgn}(|x| + b) = \begin{cases} -1 & \text{if } |x| + b < 0 \\ 0 & \text{if } |x| + b = 0 \\ +1 & \text{if } |x| + b > 0 \end{cases} \tag{3}$$

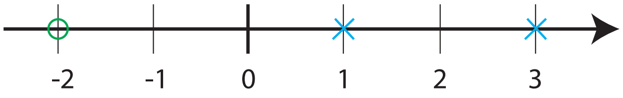
and one possible set is

$$S = \{h_b : b \leq 0\} \cup \{h_1\} \,. \tag{4}$$

Note that if $b > 0$ then $h_b(x) = 1$ for any $x \in \mathbb{R}$ and thus all classification functions h_b with $b > 0$ are equal to (for example) h_1.

Consider the following data set. The ith data point is given by x_i and the corresponding class is $y_i \in \{-1, +1\}$.

i	x_i	y_i
1	-2	-1
2	1	$+1$
3	3	$+1$



Problem 6: What would happen if you try to solve the dual problem for fitting an SVM without slack variables to this data set? Explain your answer.

Since the data set is not linearly separable in the feature space $\phi(x)$ the constraints of the primal problem are not fulfilled and therefore the dual optimization problem cannot have a solution. The dual constraints can always be fulfilled easily (for example $\alpha_i = 0, i = 1, 2, 3$); thus some α_i must diverge, i.e. $\alpha_i \to \infty$.

We can see this explicitly in our example. From the constraints we get

$$\sum_i \alpha_i y_i = \alpha_1 - \alpha_2 + \alpha_3 = 0$$
$$\Leftrightarrow \quad \alpha_2 = \alpha_1 + \alpha_3 \,.$$

We maximize the objective function

$$\begin{aligned} &\sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j y_i y_j \alpha_i \alpha_j x_i x_j \\ &= \alpha_1 + \alpha_2 + \alpha_3 - \frac{1}{2}\underbrace{\left(\alpha_1^2 + 4\alpha_2^2 + 9\alpha_3^2 - 2 \cdot 2 \cdot \alpha_1\alpha_2 + 2 \cdot 3 \cdot \alpha_1\alpha_3 - 2 \cdot 2 \cdot 3 \cdot \alpha_2\alpha_3\right)}_{=0 \text{ if } \alpha_1 + 3\alpha_3 = 2\alpha_2} \,. \end{aligned}$$

We can set the right-hand term to 0 by setting

$$\frac{1}{2}\alpha_1 + \frac{3}{2}\alpha_3 = \alpha_2 = \alpha_1 + \alpha_3 \,,$$

where the second equality is due to the constraints, as shown above. Thus, we get

$$\alpha_1 = \alpha_3 = \frac{1}{2}\alpha_2 \,.$$

Under these condition the right-hand term of the objective function becomes 0 and we only maximize

$$\alpha_1 + \alpha_2 + \alpha_3 = 4\alpha_1 \,,$$

which therefore goes to infinity.

5 Kernelized k-nearest neighbors

To classify the point $\boldsymbol{x}$ the k-nearest neighbors finds the k training samples $\mathcal{N} = \{\boldsymbol{x}^{(s_1)}, \boldsymbol{x}^{(s_2)}, \dots, \boldsymbol{x}^{(s_k)}\}$ that have the shortest distance $||\boldsymbol{x} - \boldsymbol{x}^{(s_i)}||_2$ to $\boldsymbol{x}$. Then the label that is mostly represented in the neighbor set $\mathcal{N}$ is assigned to $\boldsymbol{x}$.

Problem 7: Formulate the k-nearest neighbors algorithm in feature space by introducing the feature map $\boldsymbol{\phi}(\boldsymbol{x})$. Then rewrite the k-nearest neighbors algorithm so that it only depends on the scalar product in feature space $K(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{\phi}(\boldsymbol{y})$.

The distance to a training sample in feature space is given by

$$||\boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{\phi}(\boldsymbol{x}^{(s_i)})||_2 \,.$$

We can replace this by the squared distance because this will not change which points are nearest to

$\boldsymbol{x}$. Thus we have

$$\begin{aligned}||\boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{\phi}(\boldsymbol{x}^{(s_i)})||_2^2 &= (\boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{\phi}(\boldsymbol{x}^{(s_i)}))^T(\boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{\phi}(\boldsymbol{x}^{(s_i)})) \\ &= \boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{\phi}(\boldsymbol{x}) - 2\boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{\phi}(\boldsymbol{x}^{(s_i)}) + \boldsymbol{\phi}(\boldsymbol{x}^{(s_i)})^T\boldsymbol{\phi}(\boldsymbol{x}^{(s_i)})\,.\end{aligned}$$

The first term is a constant when searching for the k training samples that minimize this function. Hence we can drop the first term and must find the k training samples $\boldsymbol{x}^{(s_i)}$ that minimize

$$\boldsymbol{\phi}(\boldsymbol{x}^{(s_i)})^T\boldsymbol{\phi}(\boldsymbol{x}^{(s_i)}) - 2\boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{\phi}(\boldsymbol{x}^{(s_i)}) = K(\boldsymbol{x}^{(s_i)}, \boldsymbol{x}^{(s_i)}) - 2K(\boldsymbol{x}, \boldsymbol{x}^{(s_i)})\,.$$