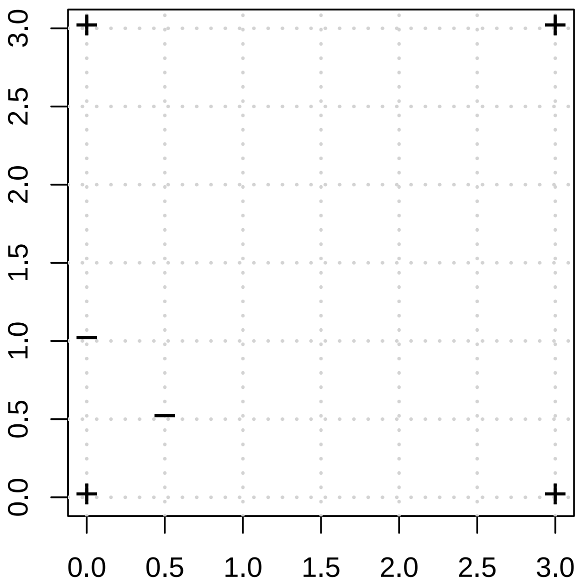
Supervised Learning **Exercise sheet 5**
`https://slds-lmu.github.io/i2ml/` **WS 2021/2022**

Exercise 1: SVM – Support Vectors and Separating Hyperplane

The primal optimization problem for the two class soft margin SVM classification is given by

$$\begin{aligned} \min_{\theta, \theta_0, \mathbf{x}^{(i)}} \quad & \frac{1}{2}||\theta||^2 + C \sum_{i=1}^{n} \zeta^{(i)} \\ \text{s.t. :} \quad & y^{(i)}(\theta^\top \mathbf{x}^{(i)} + \theta_0) \geq 1 - \zeta^{(i)}, \\ & \zeta^{(i)} \geq 0, \quad \forall i = 1, \dots, n. \end{aligned}$$



(a) Add the decision boundary to the figure for $\hat{\theta} = (1, 1)^T, \hat{\theta}_0 = -2$.

(b) Identify the coordinates of the support vector(s) and compute the values of their slack variables $\zeta^{(i)}$.

(c) Compute the euclidean distance of the non-margin-violating support vector(s) (i.e. support vector(s) that are located on the margin hyperplanes) to the decision boundary.

(d) What needs to be changed in the plot such that a hard margin SVM results into the same decision boundary?

Exercise 2: SVM – Optimization

Write your own stochastic subgradient descent routine to solve the soft-margin SVM in the primal formulation.

Hints:

- Use the regularized-empirical-risk-minimization formulation, i.e., an optimization criterion without constraints.
- No kernels, just a linear SVM.
- Compare your implementation with an existing implementation (e.g., `kernlab` in R). Are your results similar? Note that you might have to switch off the automatic data scaling in the already existing implementation.

Exercise 3: SVM – Kernel Trick

The polynomial kernel is defined as

$$k(x, \tilde{x}) = (x^T \tilde{x} + b)^d.$$

Furthermore, assume $x \in \mathbb{R}^2$ and $d = 2$.

(a) Derive the explicit feature map ϕ taking into account that the following equation holds:

$$k(x, \tilde{x}) = \langle \phi(x), \phi(\tilde{x}) \rangle$$

(b) Describe the main differences between the kernel method and the explicit feature map.

Exercise 4: Gaussian Processes

Assume your data follows the following law:

$$\boldsymbol{y} = \boldsymbol{f} + \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{\Psi}),$$

with $\boldsymbol{f} = f(\boldsymbol{x}) \in \mathbb{R}^n$ being a realization of a Gaussian process (GP), for which we a priori assume

$$f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')).$$

$\boldsymbol{x}$ here only consists of 1 feature that is observed for n data points.

(a) Derive / Define the prior distribution of $\boldsymbol{f}$.

(b) Derive the posterior distribution $\boldsymbol{f}|\boldsymbol{y}$.

(c) Derive the posterior predictive distribution $y_*|x_*, \boldsymbol{x}, \boldsymbol{y}$ for a new sample x_* from the same data generating process.

(d) Implement the GP with squared exponential kernel, zero mean function and $\ell = 1$ from scratch for $n = 2$ observations $(\boldsymbol{y}, \boldsymbol{x})$ and $\boldsymbol{\Psi} = \boldsymbol{I}$. Do this as efficiently as possible by explicitly calculating all expensive computations by hand. Do the same for the posterior predicitve distribution of y_*. Test your implementation using simulated data.