

# Linear Models and SVMs

## Optimization for Machine Learning — Homework #1

Monday 8<sup>th</sup> May, 2023

The theory part can be handed-in physically during the exercise session

### Part I: Theory

12+2 points

#### I.1. Convexity

**Exercise I.1** (2+1 points). Let  $f: \mathbb{R}^d \rightarrow \mathbb{R}$  and  $g: \mathbb{R}^d \rightarrow \mathbb{R}$  be two convex functions. Show that  $f + g$  is convex.

*Bonus:* Show that  $x \mapsto \max(f(x), g(x))$  is convex.

**Exercise I.2** (2+1 points). Let  $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$  be a linear function (i.e.  $f(x) = Ax$ , for  $A \in \mathbb{R}^{m \times n}$ ) and  $g: \mathbb{R}^m \rightarrow \mathbb{R}$  be convex functions. Show that  $g \circ f: \mathbb{R}^n \rightarrow \mathbb{R}$  is convex.

*Bonus:* What about if  $f$  is affine, i.e.  $f(x) = Ax + b$ , with  $b \in \mathbb{R}^m$ ?

#### I.2. (Sub-) Gradients

**Exercise I.3** (2 points). Let  $\phi: \mathbb{R}^d \rightarrow \mathbb{R}^p$  be a differentiable basis function. Define the model  $f: \mathbb{R}^d \times \mathbb{R}^p \rightarrow \mathbb{R}$  as, for  $b \in \mathbb{R}$ ,

$$\begin{aligned} f &: \mathbb{R}^d \times \mathbb{R}^p \longrightarrow \mathbb{R} \\ (x, w) &\longmapsto \langle w, \phi(x) \rangle + b \end{aligned}$$

1. What is  $\nabla_w f(x, w)$ ?
2. What is  $\nabla_x f(x, w)$ ?

**Exercise I.4** (2 points). For  $\lambda \geq 0$ , let  $E(w) = E_s(w) + \lambda \|w\|_1$ , where  $E_s$  is assumed to be convex and everywhere differentiable, and

$$\|w\|_1 := \sum_{i=1}^p |w_i|$$

is the 1-norm of  $w$ .

1. Is  $E$  convex?
2. Where is  $w \mapsto \|w\|_1$  (not) differentiable?
3. What is  $\partial E(w)$  (as a function of  $\nabla E_s(w)$ )? (*Hint*: see what happens for  $p = 1, 2$  first.)

### I.3. Support Vector Machines

**Exercise I.5** (4 points). The Support Vector Machines (SVM) algorithm solves a binary classification task. Given  $N$  couples samples / targets  $\{x_i, t_i\}_{i \in [N]}$ , with  $x_i \in \mathbb{R}^d$  and  $t_i \in \{-1, 1\}$  for each  $i \in [N] := \{1, \dots, N\}$ , the goal is to classify the samples, i.e. find the regions where the positive (resp. negative) samples lie. We will do that by finding a **hyperplane that separates** (or **splits**) the dataset, with positive samples on one side of the hyperplane and the negative on the other. We assume that **such an hyperplane exists** (the samples are said to be *linearly separable*). One can picture the case for  $d = 2$  or  $d = 3$ , where points are clustered in two groups and can be separated by a straight line ( $d = 2$ ) or a plane ( $d = 3$ ), see Figure 1a.

A hyperplane in  $\mathbb{R}^d$  is represented with a vector  $w \in \mathbb{R}^d$  and a *bias*  $b \in \mathbb{R}$  with the equation

$$y(x, w) = \langle w, x \rangle + b. \quad (1)$$

For  $i \in [N]$ , denote  $y_i := y(x_i, w) = \langle w, x_i \rangle + b$ . Note that  $y_i$  **still depends on**  $(x, w)$  even if the notation is dropped.

The hyperplane equation (1) splits  $\mathbb{R}^d$  into three regions:

- points  $x$  such that  $y(x) > 0$ ,
- points  $x$  such that  $y(x) = 0$  (the hyperplane itself),
- points  $x$  such that  $y(x) < 0$ .

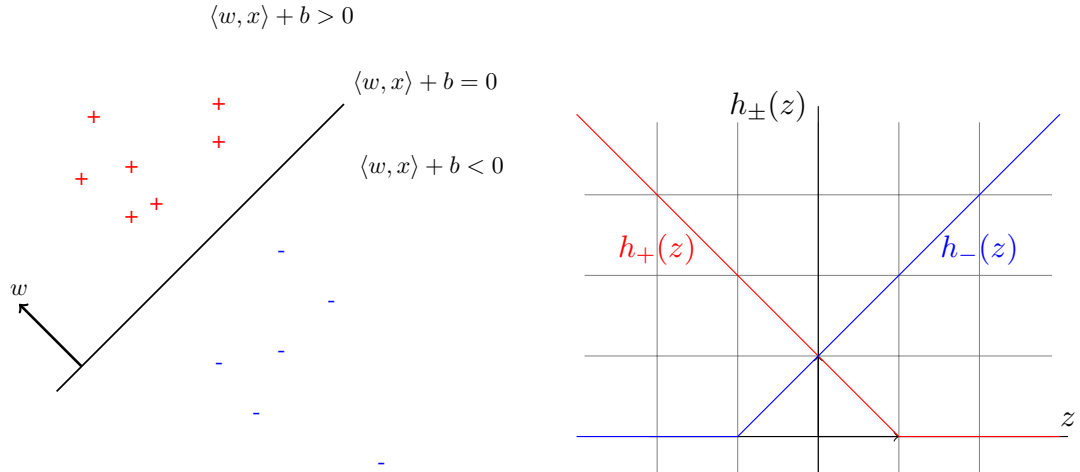
Therefore, we would like to find an hyperplane such that the samples  $x_i$  that have a positive target  $t_i = 1$  all lie on the side of the hyperplane where  $y(x) > 0$ , i.e.  $t_i = 1 \implies y_i > 0$ , and reciprocally all samples  $x_i$  such that  $t_i = -1$  should be on the side where  $y(x) < 0$ , i.e.  $t_i = -1 \implies y_i < 0$ . Then, the target  $t_i$  could simply be read from  $y_i$  by looking at its sign.

With this requirement, the product  $t_i y_i$  should **always be positive**, and the loss we define is

$$E(w) = \sum_{i=1}^N \max(0, 1 - t_i y_i) = \sum_{i=1}^N \max(0, 1 - t_i (\langle w, x_i \rangle + b)), \quad (2)$$

which has the effect of pushing the product  $t_i y_i$  towards the greatest value possible, for all  $i \in [N]$ .

Let  $\mathcal{I}_+ = \{i \in [N] \mid t_i = +1\}$  and  $\mathcal{I}_- = \{i \in [N] \mid t_i = -1\}$  be the sets of the indices of the positive and negative samples. The loss can be further written as



(a) Possible solution found by SVM. Points with  $t_i = +1$  (in red) are on the side where  $y(x) > 0$ , points with  $t_i = -1$  (in blue) are on the side where  $y(x) < 0$ .

(b) In red, the loss function for the samples that have  $t_i = +1$ . In blue, the loss function for the samples with  $t_i = -1$ .

Figure 1: SVM illustration,  $d = 2$ .

$$E(w) = \sum_{i \in \mathcal{I}_+} \max(0, 1 - y_i) + \sum_{i \in \mathcal{I}_-} \max(0, 1 + y_i) =: \sum_{i \in \mathcal{I}_+} h_+(y_i) + \sum_{i \in \mathcal{I}_-} h_-(y_i)$$

The functions  $h_+$  and  $h_-$  are plotted in Figure 1b.

1. Why is the loss  $w \mapsto E(w)$  convex? (Hint: if  $f$  and  $g$  are convex, then  $\max(f, g)$  is convex).

Recall that the function  $h_+ : \mathbb{R} \ni z \mapsto \max(0, 1 - z)$  has the following subgradient (see Exercise Sheet #2, I.2):

$$\partial h_+(z) = \begin{cases} \{-1\} & \text{if } z < 1, \\ [-1, 0] & \text{if } z = 1, \\ \{0\} & \text{if } z > 1. \end{cases}$$

2. Show that the subgradient of the function  $h_- : \mathbb{R} \ni z \mapsto \max(0, 1 + z)$  is

$$\partial h_-(z) = \begin{cases} \{0\} & \text{if } z < -1, \\ [0, 1] & \text{if } z = -1, \\ \{1\} & \text{if } z > -1. \end{cases}$$

3. Compute the subgradient of the loss  $E$  at  $w \in \mathbb{R}^p$ .
4. What should be the (sub-) gradient descent algorithm to minimize  $E$ ?

## Part II: Programming

8+2 points

**Exercise II.1** (8+2 points). This exercise implements some material from Exercise I.5. The data is generated with the helper function `gen_binary_data` in `ex02/utils.py`. The dimension is set to  $d = 2$  in order to visualize the result at the end. The generation simply draws some random points on the plane, draws an hyperplane, and classify the points depending on the sign of  $\langle w, x \rangle + b$ . Therefore, the training data is linearly separable.

1. Implement the loss from (2).
2. Implement the (sub-) gradient algorithm derived in I.5.4
3. Visualize the solution found by the algorithm, as well as its convergence.
4. *Bonus*: What happens if the data is not linearly separable?