2.2.1.1 Rosenblatt's perceptron

Rosenblatt's perceptron

The function $\hat{f}$ in 2.2.1 is also called the Perceptron and dates back to Warren McCulloch and Walter Pitts in 1943. Due to the discontinuity of $\hat{f}$, gradient methods for determining w and b based on a least squares method are not possible. An alternative method was proposed by Frank Rosenblatt in 1958. For this purpose, the perceptron loss is defined as

2.2.3

$$\mathcal{L}(w,b) := \frac{1}{N}\sum_{i=1}\max(0, -y_i(\langle w, x_i\rangle + b))$$

Intuition

- For correctly classified points ($y_i(\langle w, x_i\rangle) \geq 0$), the loss is 0
- For misclassified points ($y_i(\langle w, x_i\rangle) + b) < 0$) the loss is $-y_i(\langle w, x_i\rangle + b)$, which is proportional to the (unnormalized) distance from x_i to the hyperplane. Points farther from the hyperplane incur large penalties

Back to the script

Note that in the case where $\langle w, x_i\rangle + b$ and y_i have the same sign, the i-th term in this sum is equal to zero. If the signs are different, the corresponding summand is equal to $\langle w, x_i\rangle + b$ and thus proportional to the distance from x_i to the hyperplane.

Explanation:

If y_i is negative (e.g. -1) and $\langle w, x_i\rangle + b$ is also negative (e.g. -2) then:

$$-y_i(\langle w, x_i\rangle + b) = -(-1)\cdot(-2) = -2$$

and thus the max is 0!
Now if the signs are different e.g. y_i is 1 and $\langle w, x_i\rangle + b$ is still -2 then:

$$-y_i(\langle w, x_i\rangle + b) = -(1)\cdot(-2) = 2$$

(which is equal to $\langle w, x_i\rangle + b$ and thus proportial to the distance from x_i to the hyperplane)

Back to the script

Therefore, misclassified points x_i that are further away from the hyperplane are penalized more heavily than those that are close.
We note that 2.2.3 is a convex function in the sought variables w and b. Unfortunately is is not differentiable in w and b at $y_i(\langle w, x_i \rangle + b) = 0$.
However, we still calculate the derivative of the function

$$f_i(w, b) := \max(0, -y_i(\langle w, x_i \rangle) + b)$$

with respect to the two variables $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$ and ignore the fact that the maximum function is not differentiable. At such points, we set the derivative to the value 0. The resulting derivative is called a *subgradient* of f_i, and is given by

$$\partial_w f_i(w, b) := -y_i x_i \mathbf{1}_{y_i(\langle w, x_i \rangle + b) < 0}$$

$$\partial_b f_i(w, b) := -y_i \mathbf{1}_{y_i(\langle w, x_i \rangle + b) < 0}$$

How are those computed?

Explanation

Case Analysis for Gradients

The loss has two cases depending on whether the point is **correctly classified** or **misclassified**
Case 1: correctly classified:
If $y_i(\langle w, x_i \rangle + b) \geq 0$

- The term inside $\max(0, -y_i(\langle w, x_i \rangle + b))$ is ≤ 0 (because of the negative sign in front of the y_i), so $f_i(w, b) = 0$
- **Gradients**: Since the loss is a constant (0), the gradients are

$$\partial_w f_i(w, b) = 0, \quad \partial_b f_i(w, b) = 0$$

Case 2: Misclassified
If $y_i(\langle w, x_i \rangle + b) < 0$:

- The term inside $\max(0, -y_i(\langle w, x_i \rangle + b))$ is > 0 (because of the negative sign in front of the y_i), so $f_i(w, b) = -y_i(\langle w, x_i \rangle + b)$
- **Gradients**: The loss is linear in w and b, so we differentiate
  - With respect to w

$$\partial_w f_i(w, b) = \partial_w(-y_i(\langle w, x_i \rangle + b)) = -y_i x_i$$

  - With respect to b

$$\partial_b f_i(w, b) = \partial_b(-y_i(\langle w, x_i \rangle) + b) = -y_i$$

Now combining the cases and using the indicator function $\mathbf{1}$:

$$\partial_w f_i(w,b) = -y_i x_i \cdot 1_{y_i(\langle w, x_i \rangle + b) < 0}$$

$$\partial_w f_i(w,b) = -y_i \cdot 1_{y_i(\langle w, x_i \rangle + b) < 0}$$

Back to the script

Rosenblatt's algorithm is now a stochastic subgradient method with step size $\tau > 0$ and batch size $B \in \{1, \ldots, N\}$ applied to $L(w,b)$:

$$\begin{cases} \text{Initialize } w_0 \in \mathbb{R}^d, b_0 \in \mathbb{R} \\ \text{For } k \in \mathbb{N} \text{ iterate:} \\ \text{Choose a random B-element subset } \mathcal{I} \subset \{1, \ldots, N\}, \\ \text{Define } \mathcal{M} := \{i \in \mathcal{I} : y_i(\langle w, x_i \rangle + b) < 0\} (\text{misclassified points}), \\ w_k := w_{k-1} + \frac{\tau}{B} \sum_{i \in \mathcal{M}} y_i x_i, \\ b_k := b_{k-1} + \frac{\tau}{B} \sum_{i \in \mathcal{M}} y_i \end{cases}$$

The main problem with Rosenblatts algorithm is that we have no control over which hyperplane it converges to. In the case that the data are linearly separable, i.e. there exist $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$ such that $y_i(\langle w, x_i \rangle + b) \geq 0$ for all $i = 1, \ldots, N$, every hyperplane of this form is a fixed point of the algorithm (the algorithm stops updating once all points are correctly classified!)