SILVIA LAVAGNINI

LECTURE 4: MINIMIZING LOSS FUNCTIONS WITH GRADIENT DESCENT

31 JAN 2023

GRA 4150

A key ingredient for supervised machine learning algorithms is the objective function: this measures the loss or a cost that we want to minimize.

When deciding the loss function for our algorithm, we want (at least) the following property:

- it must measure the distance between predictions and labels;
- easy to implement;
- differentiable.

We can for example use the squared distance

$$\mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = (y^{(i)} - \hat{y}^{(i)})^2$$

Where $\hat{y}^{(i)}$ is the prediction for the i-th feature vector $x^{(i)}$.

For the Adaline algorithm we have that

$$\hat{y}^{(i)} = \sigma(z^{(i)}) = w^T x^{(i)} + b .$$

The loss will then depend on the choice of w and b.

We consider the mean over all the observations of the squared distance (MEAN SQUARED ERROR)

$$L(w, b) := \frac{1}{2n} \sum_{i=1}^{n} \mathcal{L}(y^{(i)}, \hat{y}^{(i)})$$

$$= \frac{1}{2n} \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2$$

$$= \frac{1}{2n} \sum_{i=1}^{n} (y^{(i)} - w^T x^{(i)} - b)^2$$

GOAL: we want to find w and b so that $L(w, b)$ is as small as possible.

IDEA: we use a gradient method:

the gradient of a function $F: \mathbb{R}^d \to \mathbb{R}$ in a point $p = \begin{pmatrix} p_1 \\ \vdots \\ p_d \end{pmatrix} \in \mathbb{R}^d$ is given by the vector

$$\nabla F(p) = \begin{bmatrix} \frac{\partial F}{\partial x_1}(p) \\ \frac{\partial F}{\partial x_2}(p) \\ \vdots \\ \frac{\partial F}{\partial x_d}(p) \end{bmatrix} \in \mathbb{R}^d$$

Example: $F(x_1, x_2, x_3) = x_1 \cdot x_2 + x_3^4$ and $p = \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}$

(here $d = 3$)

$$\nabla F(p) = \begin{bmatrix} x_2 \\ x_1 \\ 4x_3^3 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 4 \end{bmatrix}$$

HOW: we go in the opposite direction of the gradient.

$\boxed{d=1}$ (only one feature)

Take $n=1$ (one observation) then $L(w) = \frac{1}{2}(y - wx)^2$

and $\nabla L(w) = -(y - wx) \cdot x$

Take $n=2$ (two observation) then $L(w) = \frac{1}{4}\left[(y^{(1)} - wx^{(1)})^2 + (y^{(2)} - wx^{(2)})^2\right]$

and $\nabla L(w) = -\frac{1}{2}\left[(y^{(1)} - wx^{(1)}) \cdot x^{(1)} + (y^{(2)} - wx^{(2)}) \cdot x^{(2)}\right]$

Take n (n observation) then $L(w) = \frac{1}{2n} \sum_{i=1}^{n} (y^{(i)} - wx^{(i)})^2$

and $\nabla L(w) = -\frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - wx^{(i)}) \cdot x^{(i)}$

What if now $\boxed{d > 1}$? (It means we have more features, hence more parameters)

We repeat the same for each parameter, and we then obtain a vector of partial derivatives.

In the Adalina method we have that $d = m + 1$ where

- m is the number of features, hence the number of weights, $w_1, w_2, \dots, w_m$;
- "+1" is the bias b.

Hence the gradient is a vector with $m+1$ components:

- the first m components are

$$\nabla_w L(w,b) = \begin{bmatrix} \frac{\partial L}{\partial w_1} \\ \frac{\partial L}{\partial w_2} \\ \vdots \\ \frac{\partial L}{\partial w_m} \end{bmatrix}$$

- and the last component is $\nabla_b L(w,b) = \frac{\partial L}{\partial b}$.

Remember that $\hat{y}^{(i)} = w^T x^{(i)} + b$

$$= w_1 x_1^{(i)} + w_2 x_2^{(i)} + \dots + w_m x_m^{(i)} + b$$

hence

$$L(w,b) = \frac{1}{2n} \sum_{i=1}^{n} (y^{(i)} - w^T x^{(i)} - b)^2$$

$$= \frac{1}{2n} \sum_{i=1}^{n} (y^{(i)} - w_1 x_1^{(i)} - w_2 x_2^{(i)} - \dots - w_m x_m^{(i)} - b)^2$$

Then

- $$\frac{\partial L}{\partial w_j} = -\frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)}) x_j^{(i)}$$
- $$\frac{\partial L}{\partial b} = -\frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)}) .$$

We now update the weights and bias by taking a step in the opposite direction of the gradient of the loss function:

$$w = w + \Delta w \quad \text{where} \quad \Delta w = -\eta \nabla_w L(w, b)$$

$$b = b + \Delta b \quad \text{where} \quad \Delta b = -\eta \nabla_b L(w, b)$$

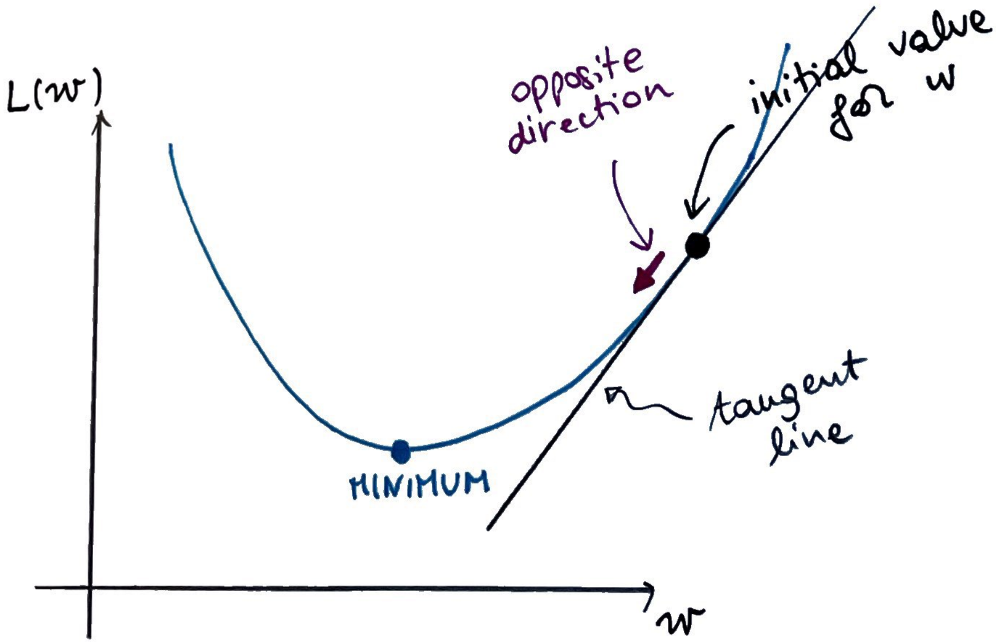
INTERPRETATION

(!) The derivative of a function in a point when $d = 1$ tells what is the <u>slope</u> of the <u>tangent line</u> to the function in that point

- positive derivative → positive slope → increasing line;
- negative derivative → negative slope → decreasing line.

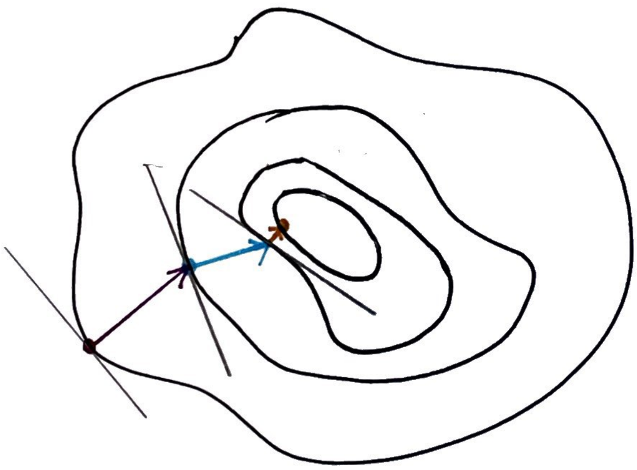
With gradient descent we go in the direction opposite to the gradient / derivative:

- positive derivative → we go "negative"
- negative derivative → we go "positive"



When we are in higher dimension $d > 1$, it is not easy to visualize the situation.

Imagine to have some contour lines, such as the ones in a geographic map: here each line represent the value of the loss function given some parameters (some values for w and b).
Suppose we want to "reach the valley" (namely the minimum value of the loss function)



starting values

$w^{(0)}, b^{(0)}$

$\downarrow$ update 1

$w^{(1)}, b^{(1)}$

$\downarrow$ update 2

$w^{(2)}, b^{(2)}$

$\downarrow$ update 3

$w^{(3)}, b^{(3)}$

The idea of gradient descent is that at each step, you "move" by going in the ~~direction~~ steepest direction. The steepest direction is defined by the gradient of the loss function.