Name and surname: Sofya Aksenyuk   Index no.: 150284

I. Consider two examples ***a*** and ***b*** described in terms of three attributes x_1, x_2, and x_3 (see table below). Compute the respective excitations when weights of a unit are as follows: w_0 = -2, w_1 = -1, w_2 = 1 and w_3 = 2. Compute the activations obtained with two different types of functions: threshold and Leaky ReLU.

example	x_1	x_2	x_3	excitation	threshold	Leaky ReLU
a	2	1	1	2*(-1) + 1*1 + 1*2 + (-2) = -1	0	-0.01
b	1	2	1	1*(-1) + 2*1 + 1*2 + (-2) = +1	+1	+1

II. Compute the Mean Squared Error $E_{MSE} = \frac{1}{n}\sum_{j=1,\ldots,n}[z_j - y_j]^2$ (please use 1/n, not 1/2) based on the obtained z^j and desired y^j results for the three examples: ***a***, ***b*** and ***c***. When computing the error $\mathbf{E}^j$ for an individual example, please use: $\mathbf{E}^j = \frac{1}{2}[z_j - y_j]^2$

j	z^j	y^j	E^j
a	1	1	0
b	1	2	1/2
c	4	1	4.5

Answer: $\boldsymbol{E_{MSE}}$ = (1/3)*1*1*9 = 3

III. Consider a neuron with an excitation function *exc* = $3x_1 + 1x_2 + 2$ (w_1=3; w_2=1; w_0=2) and a **sigmoid activation function**. Assume we optimize the Minimal Square Error $E^j_{MSE} = 1/2[z^j – y^j]^2 = 1/2(\delta^j)^2$ (assume ½ is used when computing E^j_{MSE}, not 1/n). Propagate example x_i^j (*given below*) with desired output of z^j = 0.3 through the unit and apply the gradient descend algorithm to optimize the weights with the learning rate η=1. *Hint: you can use the equations you know from the lecture for the gradient descend algorithms and a sigmoid activation function; no need to derive them again.*

x_1^j	x_2^j	z^j	*exc*	y^j	δ^j
-1	1	0.3	3*(-1) + 1*1 + 2 = 0	1 / (1 + e^0) = 1/2	0.3 - 0.5 = -0.2

Δw_1^j = 1 * (-0.2) * (-1) * (1/2) * (1 - 1/2) = 0.05 $w_1^{j'} = w_1^j + \Delta w_1^j$ = 3 + 0.05 = 3.05

Δw_2^j = 1 * (-0.2) * 1 * (1/2) * (1 - 1/2) = -0.05 $w_2^{j'} = w_2^j + \Delta w_2^j$ = 1 + (-0.05) = 0.95

Δw_0^j = 1 * (-0.2) * 1 * (1/2) * (1 - 1/2) = -0.05 $w_0^{j'} = w_0^j + \Delta w_0^j$ = 2 + (-0.05) = 1.95

IV. Given the gray 3×3 matrix of inputs, perform **zeropadding** of size 1. Then, **convolve** the resulting matrix with the 2×2 filter defined by the green matrix (bias = 0) with a stride of 1.

Original matrix and matrix after zeropadding

0	0	0	0	0
0	-2	1	-1	0
0	0	-1	2	0
0	1	0	-2	0
0	0	0	0	0

Filter

0	1
-1	0

Matrix after convolution

0	2	-1	1
-2	1	0	-2
0	-2	2	2
1	0	-2	0

V. Apply **MAX pooling** with filter of size 2×2 and a stride of 2 on the gray matrix given to the left.

Original matrix

4	2	5	1
3	1	0	0
0	3	4	3
5	6	0	7

Matrix after MAX pooling

4	5
6	7