Task 1.1

a. X
    - The input data (the raw data set)
b. Input layer
    - Layer where the input data is passed down to
c. Weights w
    - Weights are the parameters of a neural network
    - It is used in the calculation of a hidden unit
d. Sum of weights function δ
    - Each node from the input layer is applied to the sum of weights in the hidden layer
    - Used in a neuron where it takes a group of weighted inputs, applies an activation function, and returns an output
e. Activation function α
    - Activation function of a node is the output of that node given an input or set of inputs
f. Hidden layer
    - Hidden layer consists some units that transform the input into something the output layer can use
    - The values are not observed in the training set
g. Output layer
    - The output layer produces the given outputs for the program.
    - Gets input from the hidden layer and optionally applies an activaion function
h. Y
    - The output data (result)

Task 1.2

For each layer, the error value is analyzed and used to adjust the threshold and weights for the next input. By doing this, the error will become lesser for each run as the nerual netowrk learns how to analyze values.
=> Adjust the the weight for each layer to minimize the error

Task 1.3

The stochastic gradient descent does not work well when the weights are initialized to the same constant. Initializing the nerual netowrk to the same weighted value makes the network take much longer to converge to an optimal solution. The nerual network will never learn if all weights start with the same value and if the soultion requires unequal weights.

Task 1.4

Actication function = Sigmoid

$o_1 * w_1 + o_2 * w_2 + o_3 = N_o$

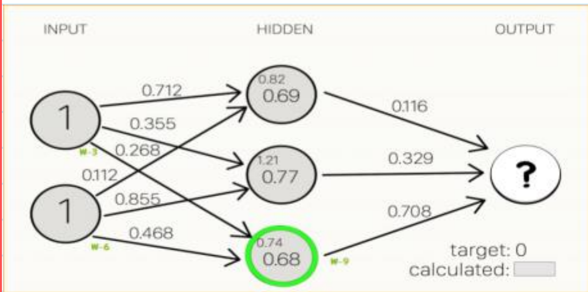
$Sig(N_o) = O_o$

$$0.69 \cdot 0.116 + 0.77 \cdot 0.329 + 0.68 \cdot 0.708 = 0.81481$$

$$x = 0.81481$$

$$S(x) = \frac{1}{1+e^{-x}} = \frac{1}{1+e^{-0.81481}} \approx 0.69313$$

Task 1.5



$$\frac{dE}{dout_H} = \frac{dE}{dnet_o} + \frac{dnet_o}{dout_H}$$

$$\frac{dE}{dw_6} = \frac{dE}{dout_H} \cdot \frac{dout_H}{dnet_H} \cdot \frac{dnet_H}{dw_6}$$

Finding net input and output for Output Layer:

$$net_o = out_{h_1} \cdot w_7 + out_{h_2} \cdot w_8 + out_{h_3} \cdot w_9 \simeq 0.81$$

$$out_o = \frac{1}{1- e^{-0.81}} \simeq 0.69$$

Calculating the total Error for the Output Layer:

$$E_{total} = \frac{1}{2}(target - out_o)^2$$

Calculating the change in w_9 using the total error:

$$\frac{\partial E_{total}}{\partial out_o} = -(target - out_o) = -(0 - 0.69) = 0.69$$

$$\frac{\partial out_o}{\partial net_o} = out_o(1 - out_o) = 0.69(1 - 0.69) \simeq 0.21$$

$$\frac{\partial net_o}{\partial w_9} = out_{h_3} = 0.68$$

$$\frac{\partial E}{\partial w_9} = \frac{\partial E}{out_o} \cdot \frac{\partial out_o}{\partial net_o} \cdot \frac{\partial net_o}{\partial w_9} = 0.69 \cdot 0.21 \cdot 0.69 \simeq 0.1$$

Update the weight:

$$w_9^+ = w_9 - \eta \cdot \frac{\partial E}{\partial w_9} = 0.708 - 0.5 \cdot 0.1 = 0.658$$

For hidden layer (h_3):

$$net_h = 1 \cdot w_3 + 1 \cdot w_6 = 0{,}268 + 0{,}468 = 0{,}74$$

$$out_h = \frac{1}{1 - e^{-0{,}74}} \simeq 0{,}68$$

Change in w_6 and w_3 with regarding to the total Error:

$$\frac{\partial E_{total}}{\partial w_6} = \frac{\partial E_{total}}{\partial out_{h_3}} \cdot \frac{\partial out_{h_3}}{\partial net_{h_3}} \cdot \frac{\partial net_{h_3}}{\partial w_6}$$

$$= \overbrace{\left(\frac{\partial E_{total}}{\partial net_o} \cdot \frac{\partial net_o}{\partial out_{h_3}} \right)} \cdot \frac{\partial out_{h_3}}{\partial net_{h_3}} \cdot \frac{\partial net_{h_3}}{\partial w_6}$$

$$\frac{\partial E_{total}}{\partial net_o} = \frac{\partial E_{total}}{\partial out_o} \cdot \frac{\partial out_o}{\partial net_o} = 0{,}69 \cdot 0{,}29 \simeq 0{,}2$$

$$net_o = out_{h_1} \cdot w_7 + out_{h_2} \cdot w_8 + out_{h_3} \cdot w_9$$

$$\frac{\partial net_o}{\partial out_{h_3}} = w_9 = 0{,}708$$

Plugging them in:

$$\frac{\partial E_{total}}{\partial out_{h_3}} = \frac{\partial E_{total}}{\partial net_o} \cdot \frac{\partial net_o}{\partial out_{h_3}} = 0{,}2 \cdot 0{,}708 \simeq 0{,}1416$$

$$\frac{\partial out_{h_3}}{\partial net_{h_3}} = out_{h_3} (1 - out_{h_3}) = 0{,}68 (-1 - 0{,}68) = 0{,}2176$$

$$\frac{\partial net_{h_3}}{\partial w_6} = out_h = 1$$

$$\frac{\partial E_{total}}{\partial w_6} = 0{,}1416 \cdot 0{,}2176 \cdot 1 \simeq 0{,}031$$

Update the weight:

$$w_6^+ = w_6 - \eta \cdot \frac{\partial E_{total}}{\partial w_6} = 0{,}468 - 0{,}5 \cdot 0{,}031 \simeq 0{,}4525$$

Same calculation for w_3:

$$w_3^+ = w_3^+ - \eta \cdot \frac{\partial E_{total}}{\partial w_3} = 0{,}268 - 0.5 \cdot 0{,}031 \simeq 0{,}2525$$

Task 2.1
The first (hidden) layer gets the (first) input data. The first layer will use some activation function on the input data. After that, the second layer will take the output data from the first (hidden) layer. The output data is already refined to a degree. The second (hidden) layer will then work with a much better data than the first layer. Therefore the shapes of the second layer is much more accurate than the shapes in the first layer.

Task 2.2
Changing the learning rate to 3 will make the network go crazy. This is because the learning rate is too high for converging. The weight will be too high for positive weights (and too low for negative) and therefore more difficult to classify the data.

Task 2.3
The network will not converge to a good solution because it has not be able to learn enough and need more neurons/layers.

Task 2.4
As mentioned in 2.3, the network will learn (converge) better if the network has more layers and neurons versus fewer layers and neurons. But having more layers and neurons will affect the time heavily.

Task: Using the keras package

Was not able to do this task because of installation problem: Tensorflow and Kensar