Exercise 5.1

Suppose we have a classification prediction problem where the dependent variable y have three possible values: 0, 1 or 2.

Our data has five samples for which the y has the following values

$$\begin{bmatrix} 0 \\ 2 \\ 2 \\ 1 \\ 0 \end{bmatrix}$$

Our model, e.g. artificial neural network, outputs three values which represent the probalities $P(y = 0)$, $P(y = 1)$ and $P(y = 2)$. Outputs for the five samples are:

$$\begin{bmatrix} 0.6 & 0.15 & 0.25 \\ 0.1 & 0.2 & 0.7 \\ 0.2 & 0.35 & 0.45 \\ 0.1 & 0.5 & 0.4 \\ 0.5 & 0.2 & 0.3 \end{bmatrix}$$

What the loss value when the loss function is cross-entropy? Give the answer rounded to three decimals.

Suppose we train the model and afterwards the outputs for the five samples are:

$$\begin{bmatrix} 0.8 & 0.15 & 0.05 \\ 0.1 & 0.15 & 0.75 \\ 0.1 & 0.15 & 0.75 \\ 0.1 & 0.7 & 0.2 \\ 0.6 & 0.2 & 0.2 \end{bmatrix}$$

How much the loss value decreases in percentage terms?

Exercise 5.2

In the lecture example in Colab (Artificial Neural Network (Part 1)) the class ANN was implemented. Copy this code and make a new class where the constructor (i.e. function `__init__`) has also a parameter `random_state` with default value `None`. Change the initialization of biases and weights so that random numbers generated are based on the seed defined by the parameter `random_state`. The idea is that one can produce a certain randomness repeatedly.

Make the following artifical neural network:

- input layer has 12 values (i.e. identity neurons)
- hidden layer that has 10 neurons
- hidden layer that has 8 neurons
- output layer that has 5 neurons

Hidden layers have activation function ReLU and output layer have activation function softmax. Use parameter `random_state=123` when creating the ANN object.

For the following input $[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.1, 0.2, 0.3, 0.4]$ predict the output. The prediction of the model is the index of the biggest value in the output vector.

What is the predicted value?