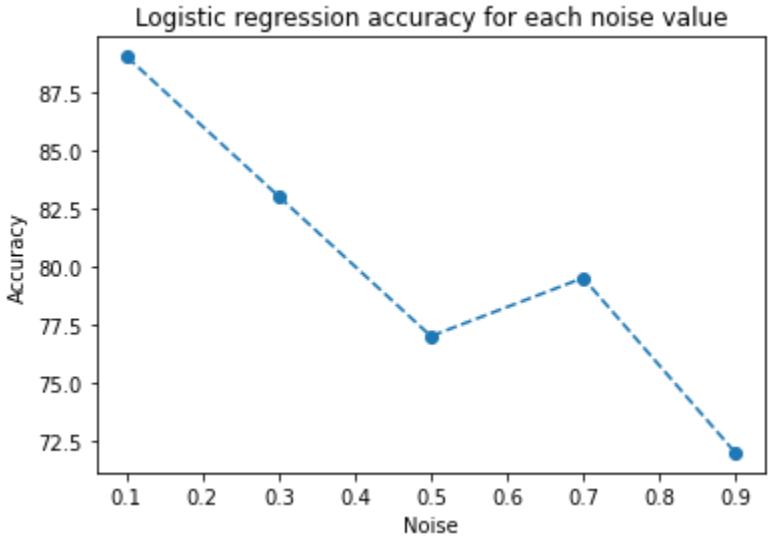
Gaetan Herry 10/11/2021

Deep learning first assignment report

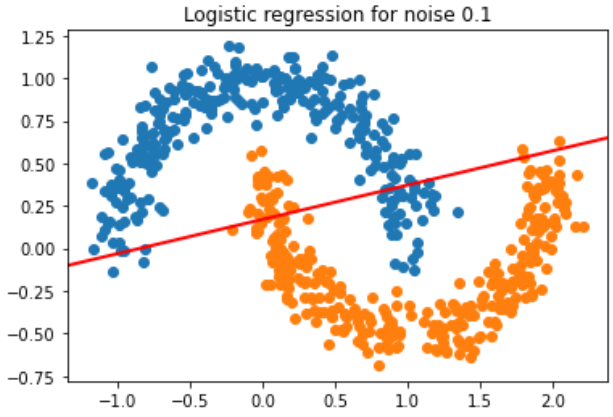
Generate two moon dataset

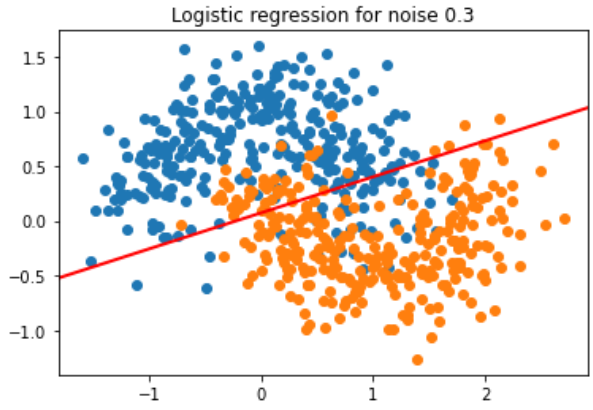
In order to generate the dataset we just need to run the command *python generate_dataset.py noise=0.1*, and to repeat that command for every desired noise.

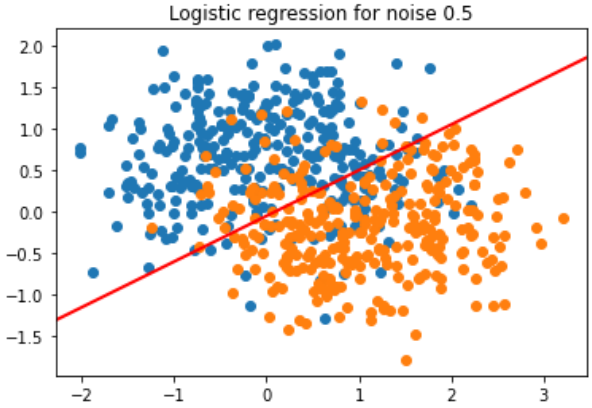
Logistic regression dataset

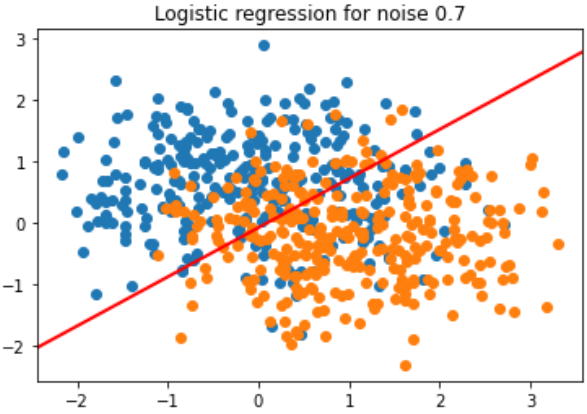


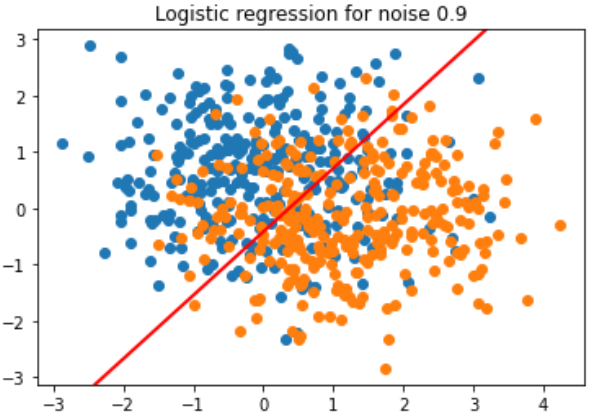
We can see with this graph that the accuracy of this classifier decreases when the noise increases, except for the noise 0.7 where it increases a bit from the 0.5 one. This is logical because by adding noise we can see that the data gets harder to separate with such a simple classifier.
From the plots below, we can see how well the classifier can separate the data. The difference between the noise 0.5 and 0.7 also seems minor, so variance could explain the small increase in accuracy for 0.7.


Logistic regression for noise 0.1

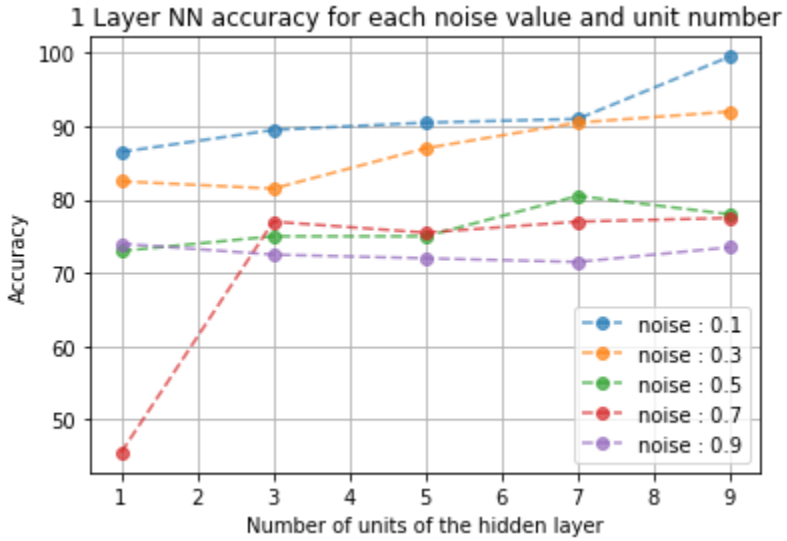

Logistic regression for noise 0.3


Logistic regression for noise 0.5


Logistic regression for noise 0.7

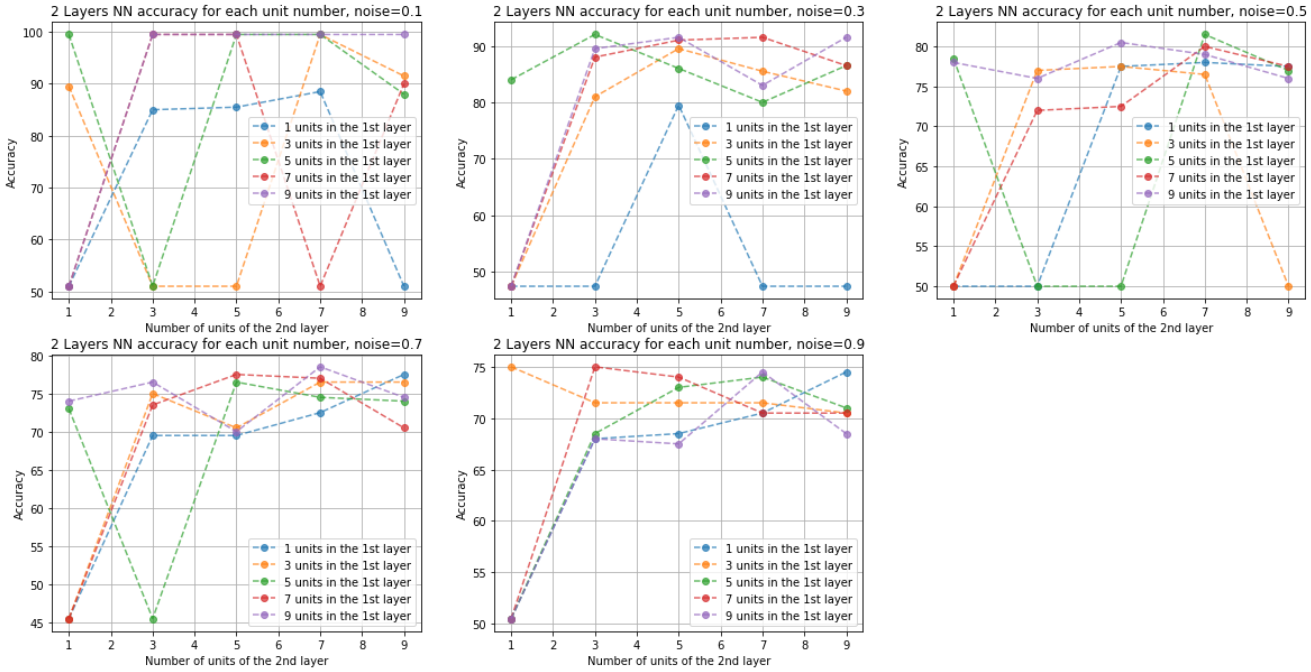

Logistic regression for noise 0.9

One layer NN classifier



From the plot above, we can see that the accuracy still decreases when the sound increases, almost always no matter how many units we have in the hidden layer.
When we add more units, it seems to globally increase the accuracy. For the noise 0.5, 7 units are more efficient than 9 but it is the only exception, that may be explained by variance.

2 hidden layers NN classifier



From the plots above, we can see the accuracy for each noise for each number of units for each layer. As for before, the accuracy decreases when the sound increases since the data is less easy to classify, but this time the accuracy gets more stable when the noise is higher. It may mean that when we use multiple layers with numerous units to explain easily classifiable data, we start to overfit the training data.
For complex data, a high number of units work a lot better, from 3 units and more. For less complex data, a high number of units seems to give more confidence in obtaining high accuracy. Otherwise, when the number of units lowers a bit, the accuracy can vary a lot.

During this project, I learned how we could implement in practice neural networks and how they work. I realized the importance of the hyperparameters: the difference in the accuracy between when we use the sigmoid and the ReLU, and when we initialize the weights at 0 or randomly before reducing them. I understood a bit more how to design the architecture of a neural network.