**How many hidden layers have you used? And why?**

The dataset contains only 5 features. Since it has fewer features, it is better to go with 1 or 2 hidden layers. We also need to remember that using not enough hidden layer can lead to underfitting. For this reason, we decided to go with 2 hidden layers.

**How many nodes in each hidden layer and why that number of nodes in particular?**

The number of nodes in the input layer is the same as the number of feature the dataset has. In our case, we have 5 nodes in the input layer. Furthermore, our dataset is binary. We should be using 2 nodes for the output layers because of it, but one of them has all weights equal to 0 and the output will always equal to 0 as well. We decided to use only 1 node

There are multiple ways to find how many nodes to uses in each hidden layer. The one we decided to go with is “The number of hidden neurons should be 2/3 the size of the input layer, plus the size of the output layer”

The first hidden layer will have 4 nodes. For the second hidden layer, we decided divide the number of nodes in the first hidden layer by 2. This resulted in 2 nodes for the second hidden layer.

**What is the activation function that you used and why? Did you use the same activation function in all layers? Why?**

For the output layer, we used Sigmoid. Since we are using only one node for the output layer for the reason we explained above, we selected Sigmoid and not Softmax, because it will update faster.

For the 2 hidden layers, we used ReLU, because is the default activation function for the hidden layer and also it is less susceptible to vanishing gradients that prevent deep models from being trained

**What learning algorithm did you use to train the neural net and why?**

We have a binary dataset without an output of 1 or 0. The classification that make sense to use is either 'binary classification' or 'multiclass classification'. We decided to go with 'binary classification', because is the simplest.

**Can you use one hidden layer only to solve this problem? If yes, how many nodes are you going to have in it? And why?**

Yes. The dataset is a simple form of only 5 features. There won’t be any problem by using only one hidden layer. We just need to select carefully the number of nodes. If we use more than necessary nodes, it can cause overfitting. If we use lass than necessary nodes, it can cause underfitting. We can calculate the number of nodes by take the mean between the input layer and the output layer.

The hidden layer will have 3 nodes.

**Can we use 5 hidden layers? Is that a good idea? Justify your answer.**

Yes. It is possible to use 5 hidden layers, but it will be a bad idea. It has the same problem as the number of nodes for each layer. If we use more than necessary layers, it can cause overfitting. If we use lass than necessary layers, it can causer underfitting. With this problem, the test dataset’s accuracy will decrease, because the model won’t be able to generalize the neu unseen data. Furthermore, the increase in the number of the hidden layer will also increase the complexity of the model.

**How did the neural net do in classifying the testing set? Comment on how good or bad it learned the function from the training set.**

The latest number for the accuracy is 80% for the training set and 75% for the testing set. The training set accuracy changes between 60% and 90% on each run, but the testing set accuracy stays at 75% on all runs. The reason for the change in percentage in the training set is the evaluation set change every time, because use a random split function of it.

In our point of view, the model learns good using the training set. In the training set, we have 18/6 of 1 and 0 respectively. In the testing set, we have 6/2 of 1 and 0 respectively. The ratio of the 1 to 0 is the same for the training and the testing set. This explains why we got this big of a accuracy in both dataset