**ScikitLab**

1. We learned in class that the XOR problem can't be solved using a single perceptron and requires a neural network to solve. In this part, you have to create the best possible neural net i.e. the one with the minimum number of layers and fewest number of parameters that will solve the XOR problem. Give the weights and intercepts for each neuron and any other parameters that you have used.

Attached File: XOR+Classification.py

In folder: XOR

Best Neural Network that classifies any XOR input:

**input layer -> 2 nodes in hidden layer 1 -> 4 nodes in hidden layer 2 -> output layer**

Loss Function used: **Solver: lbfgs**

Alpha used: **1e-5**

Random\_state used: **1**

Weights:

Between input layer and 1st hidden layer : (2\*2)

array([[-1.21612036, -1.29200374],

[-7.80305618, -1.40600775]])

Between 1st hidden layer and 2nd hidden layer: (2\*4)

array([[-5.97277229, 5.82914463, -0.41583443, -4.11574571],

[ 2.28059746, -0.91423299, -0.65766377, -0.4009252 ]])

Between 2nd hidden layer and output layer: (4\*1)

array([[ 5.4065043 ],

[-4.21971585],

[-0.70429851],

[-6.22691275]])

Intercepts:

To 1st hidden layer: (2)

array([ 1.22118787, 2.5197109 ])

To 2nd hidden layer: (4)

array([ 0.20718306, 0.89493171, -0.62604847, 1.09717845])

To output layer: (2)

array([-0.09372803])

2. In this part, you will create a neural net for a dataset chosen from the UCI ML repository. The repository is available at: <http://archive.ics.uci.edu/ml/datasets.html>

You will first have to read in the dataset using Pandas into a dataframe. The second step will involve pre-processing the dataset - analyze each of the attributes and scale them. Then you will randomly split the data into train and test parts – you are free to decide the split size. Next will be the model creation step – you will need to tune as many parameters as possible. Finally, evaluate the performance of the model using the best set of parameters.

Attached file: uci.py

Dataset used: ISOLET