Harsh Patel
CS 3600 – Introduction to AI
Project 4b
XOR Extra Credit

For the extra credit, I created a new method called testXORData in Testing.py. The training dataset was created by simply having a shuffled list of multiple repeats of the following four possible two-variable XOR examples: [([0, 0], [0]), ([0, 1], [1]), ([1, 0], [1]), ([1, 1], [0])]. The testing dataset was created in a similar way. The only difference was that the training dataset consisted of a total of 200 examples and the testing dataset consisted of a total of 800 examples. Below is a table of the average test accuracy for five runs of the testXORData method with a variable number of perceptrons in the hidden layers. Also provided is a graph plotting the number of perceptrons in the hidden layers against the average test accuracy. From the graph, you can see that the average test accuracy of 0.5 if there are no perceptrons in the hidden layer validating the relative non-linearity of the XOR function. However, as we add more and more hidden layers, it is clear to see the convergence to perfect test accuracy after roughly five perceptrons in the hidden layer. There is some noise in the accuracy when having four perceptrons, but that may be accounted for by the random shuffling of the examples and random initial values for the weights.

Number of Hidden Layer Perceptrons	Run 1	Run 2	Run 3	Run 4	Run 5	Mean	Max	Standard Deviation
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0
1	0.75	0.75	0.5	0.75	0.25	0.6	0.75	0.223607
2	1	0.75	1	0.5	1	0.85	1	0.223607
3	1	1	1	1	1	1	1	0
4	1	1	1	1	0.75	0.95	1	0.111803
5	1	1	1	1	1	1	1	0
6	1	1	1	1	1	1	1	0
7	1	1	1	1	1	1	1	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	1	1	1	0
10	1	1	1	1	1	1	1	0
15	1	1	1	1	1	1	1	0
20	1	1	1	1	1	1	1	0
25	1	1	1	1	1	1	1	0
30	1	1	1	1	1	1	1	0
35	1	1	1	1	1	1	1	0
40	1	1	1	1	1	1	1	0

