**Perceptron Learning Lab** Name and period: Justin Cai, 7

**For boolean functions on n variables, a perceptron can learn:**

1. N=2:
   1. Learned: 14/16 = 87.5%
   2. 2 example functions and the learned weight vectors

[[array([0, 0, 1]), 0], [array([0, 1, 1]), 0], [array([1, 0, 1]), 0], [array([1, 1, 1]), 0]]

[ 0.13962368 -0.50293019 -1.00346731]

[[array([0, 0, 1]), 0], [array([0, 1, 1]), 0], [array([1, 0, 1]), 0], [array([1, 1, 1]), 1]]

[ 1.86577908 1.37043942 -2.79988131]

1. N=3:
   1. Learned: 104/256= 40.6%
   2. 2 example functions and the learned weight vectors

[[array([0, 0, 0, 1]), 1], [array([0, 0, 1, 1]), 1], [array([0, 1, 0, 1]), 1], [array([0, 1, 1, 1]), 1], [array([1, 0, 0, 1]), 1], [array([1, 0, 1, 1]), 1], [array([1, 1, 0, 1]), 1], [array([1, 1, 1, 1]), 0]]

[-2.80838804 -1.76528958 -1.42225938 5.50709397]

[[array([0, 0, 0, 1]), 1], [array([0, 0, 1, 1]), 1], [array([0, 1, 0, 1]), 1], [array([0, 1, 1, 1]), 1], [array([1, 0, 0, 1]), 1], [array([1, 0, 1, 1]), 1], [array([1, 1, 0, 1]), 1], [array([1, 1, 1, 1]), 1]]

[ 0.08540325 0.55393406 0.79382258 0.04107043]

1. N=4:
   1. Learned: 1882/65536= 2.87%
   2. 2 example functions and the learned weight vectors

[[array([0, 0, 0, 0, 1]), 0], [array([0, 0, 0, 1, 1]), 0], [array([0, 0, 1, 0, 1]), 0], [array([0, 0, 1, 1, 1]), 0], [array([0, 1, 0, 0, 1]), 0], [array([0, 1, 0, 1, 1]), 0], [array([0, 1, 1, 0, 1]), 0], [array([0, 1, 1, 1, 1]), 0], [array([1, 0, 0, 0, 1]), 0], [array([1, 0, 0, 1, 1]), 0], [array([1, 0, 1, 0, 1]), 0], [array([1, 0, 1, 1, 1]), 0], [array([1, 1, 0, 0, 1]), 0], [array([1, 1, 0, 1, 1]), 0], [array([1, 1, 1, 0, 1]), 0], [array([1, 1, 1, 1, 1]), 0]]

[ 0.59969962 0.30462678 -0.36372338 -0.72717243 -1.87989632]

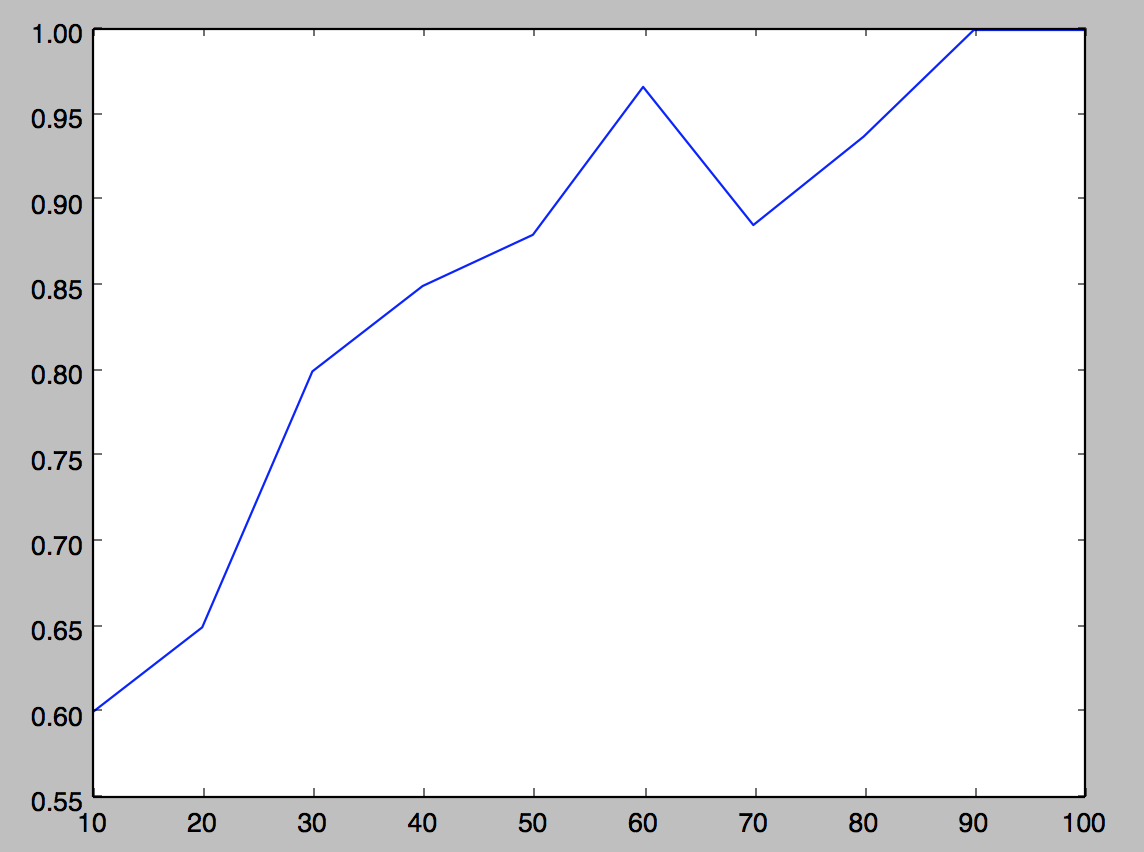
[[array([0, 0, 0, 0, 1]), 0], [array([0, 0, 0, 1, 1]), 0], [array([0, 0, 1, 0, 1]), 0], [array([0, 0, 1, 1, 1]), 0], [array([0, 1, 0, 0, 1]), 0], [array([0, 1, 0, 1, 1]), 0], [array([0, 1, 1, 0, 1]), 0], [array([0, 1, 1, 1, 1]), 0], [array([1, 0, 0, 0, 1]), 0], [array([1, 0, 0, 1, 1]), 0], [array([1, 0, 1, 0, 1]), 0], [array([1, 0, 1, 1, 1]), 0], [array([1, 1, 0, 0, 1]), 0], [array([1, 1, 0, 1, 1]), 0], [array([1, 1, 1, 0, 1]), 0], [array([1, 1, 1, 1, 1]), 1]]

[ 3.88381811 2.73413986 1.52099482 0.85944884 -8.87591384]

Create a training set and a testing set over 10 boolean inputs (x) where the function f(x) = majority. Use a training size of about 100 vectors. Plot the accuracy of a perceptron and of a decision tree, each on the testing set, for the target concept. The x-axis should be “training set size” and the y-axis “accuracy on test set”. Plot both functions on the same set of axes.

For the perceptron: use as many epochs as you deem necessary. For both: do NOT test on the training data!

**Perceptron**



**Decision Tree**

