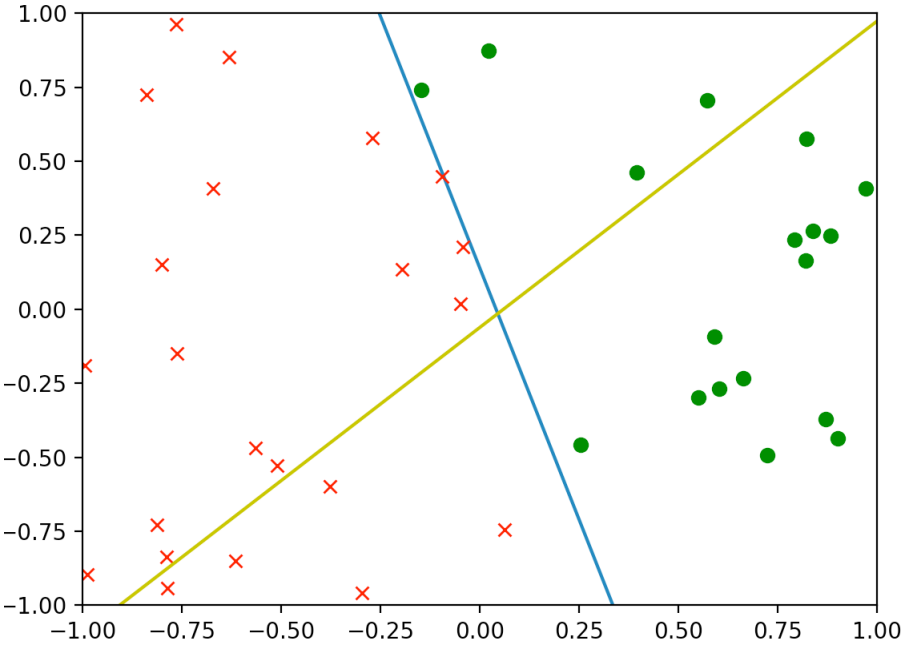
PseudoPLA

Greg Pennisi

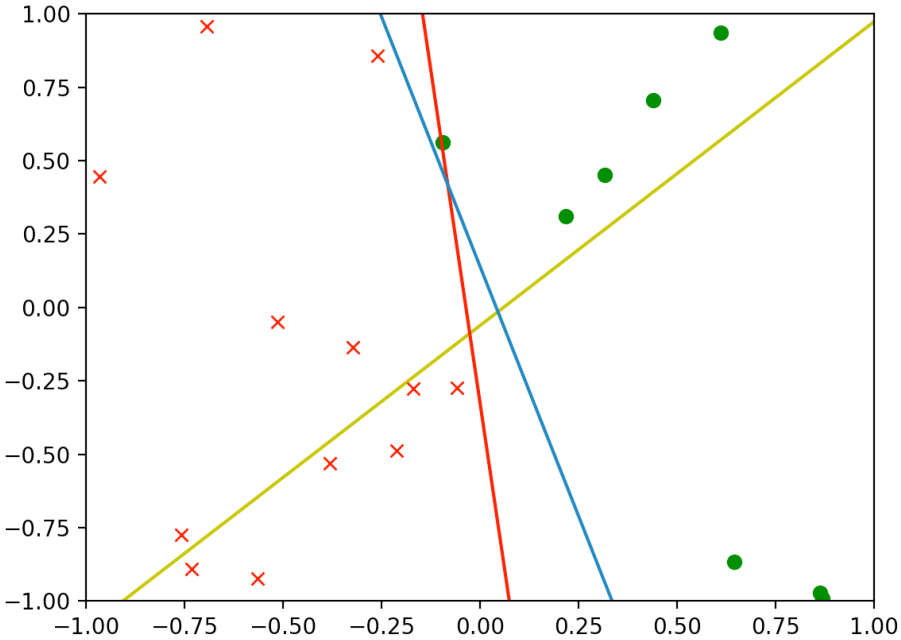
October 2018

I chose a random target function $f(x) = -3.4x + 0.14$ and built a program to simulate the PLA with the goal of nearing this target function. First, it plots 40 random points and labels them based on where they lie relative to the target function. A random starting hypothesis is then generated by plotting two random points and calculating the function that graphs the line segment connecting them. In this first image, the blue line is the target function and the yellow line is a randomly generated starting hypothesis. Red X's correspond to a -1 label and green circles correspond to +1.



Next, 20 new random points are generated and labelled according to our target function. At this stage, the PLA begins. Accessing the points in a random order, the program runs until it reaches a mislabelled point. Due to a lack of matplotlib knowledge, I was unable to figure out how to represent the data as vectors, which resulted in the need of an algorithm which mimics PLA. I decided to do the following:

1. For a mislabeled point x_n, measure the distances between x_n and the two points where the current hypothesis line intersects with the boundaries of the graph
2. Move 2 percent of the way between x_n and its closer boundary intercept, and save this coordinate pair
3. Move 0.5 percent of the way between x_n and its further boundary intercept, and save this coordinate pair
4. Connect these two new coordinate pairs with a line segment, and then calculate a point-slope formula for this line segment. This will be your adjusted hypothesis formula
5. Repeat the process until no mislabeled points exist



By using this algorithm, adjustments are made in a way that's almost identical to the PLA. After enough iterations, the hypothesis will correctly label all 20 points, and a new graph will be displayed (shown above). The blue line is the target function, the yellow line is the initial randomly generated hypothesis, and the red line is the final hypothesis reached after some number of iterations of the (pseudo) PLA. The points in this figure are the 20 new points and are not part of the originally generated set of 40 points.

The number of iterations varied greatly from run to run. The majority ranged from 10-90 iterations, but there were a noticeable number of outliers sitting within the 1-10 and 90-160 ranges. The high-iteration runs tended to have at

least a few points placed extremely close to the target function, and on both sides. This gives the final hypothesis very little room for error, resulting in many iterations.