Homework 01

Zach Stecher

Due: 9/20/16

Problem 1.2

1.2a)

We know that the regions represented h(x) = +1 and h(x) = -1 can be seperated by a line because h(x) = 0 seperates the two reasons, so the line that seperates the two regions is h(x) = 0.

h(x) = 0 can be represented as:

$$x_0 w_0 + x_1 w_1 + x_2 w_2 = 0 \tag{1}$$

By moving some of the equation to the other side, we end up with:

$$x_2 = \frac{-w_0 - x_1 w_1}{w_2} \tag{2}$$

From there, if we move $-w_0$ to the end and split this side of the equation up for clarity, we get:

$$x_2 = \frac{-x_1 w_1}{w_2} - \frac{w_0}{w_2} \tag{3}$$

So using $y = mx + b$ where m is the slope and b is the intersect, in terms of w and x the slope is $\frac{-x_1}{w_2}$ and the intersect is $\frac{-w_0}{w_2}$.

1.2b Draw figures for cases w = [1, 2, 3] and w = -[1, 2, 3]

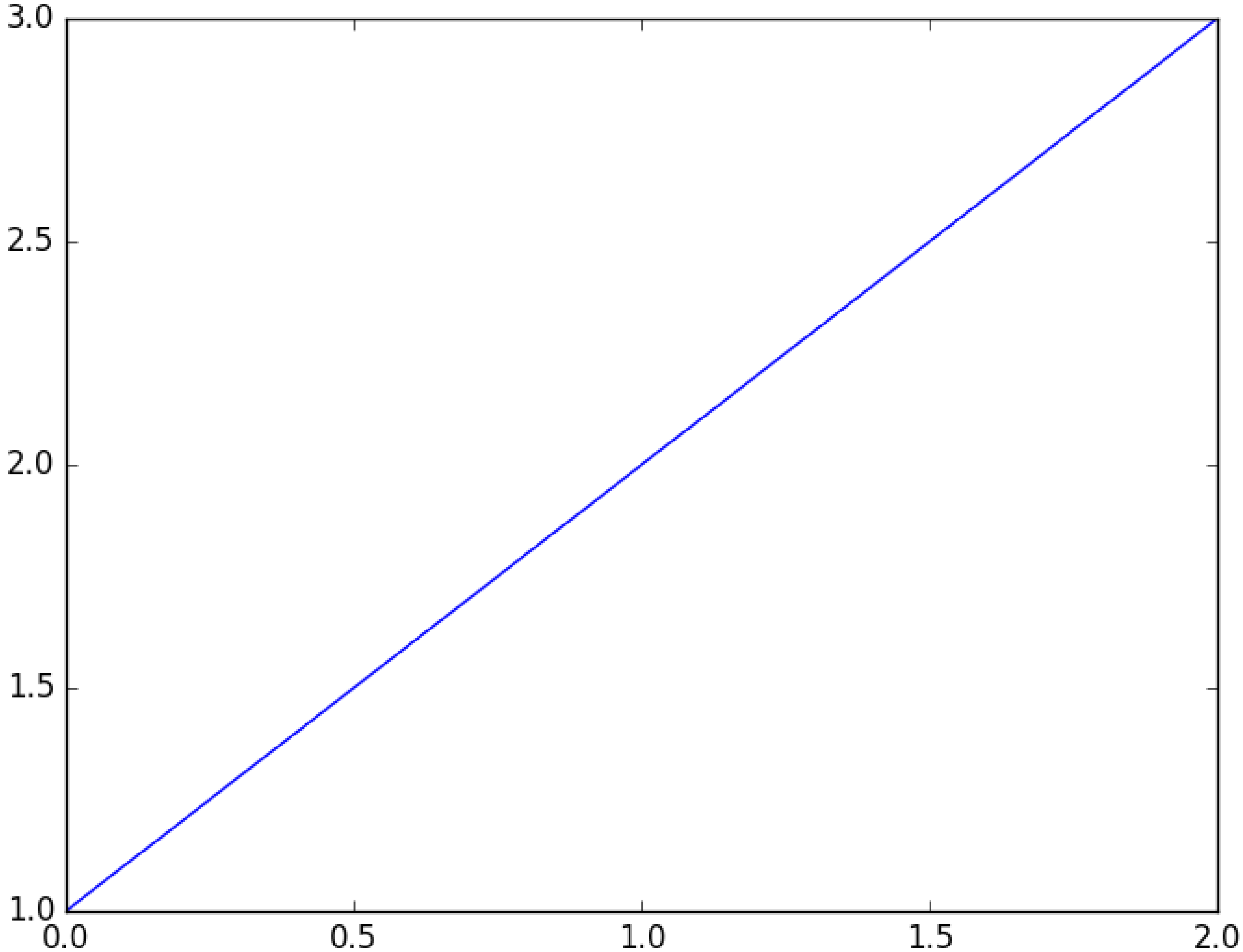

Figure 1: With positive weight values.

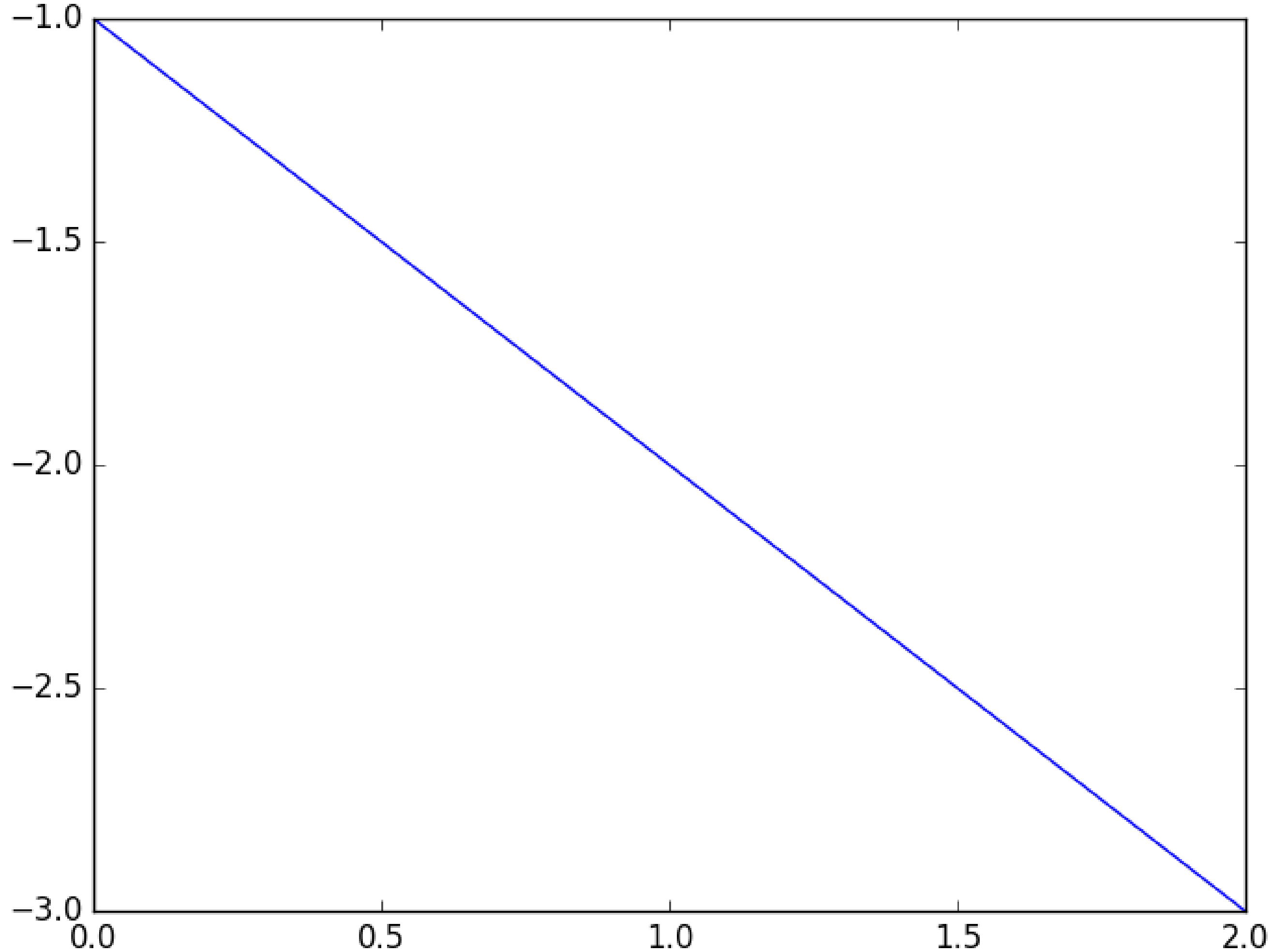

Figure 2: With negative weight values

.

Problem 1.4

For all sections of problem 1.4, we used a provided base version of the Perceptron learning algorithm with small modifications.

1.4a

For this section, the Perceptron was called on a deta set size 20 by the following code:

```
p = Perceptron(20)
p.plot()
```

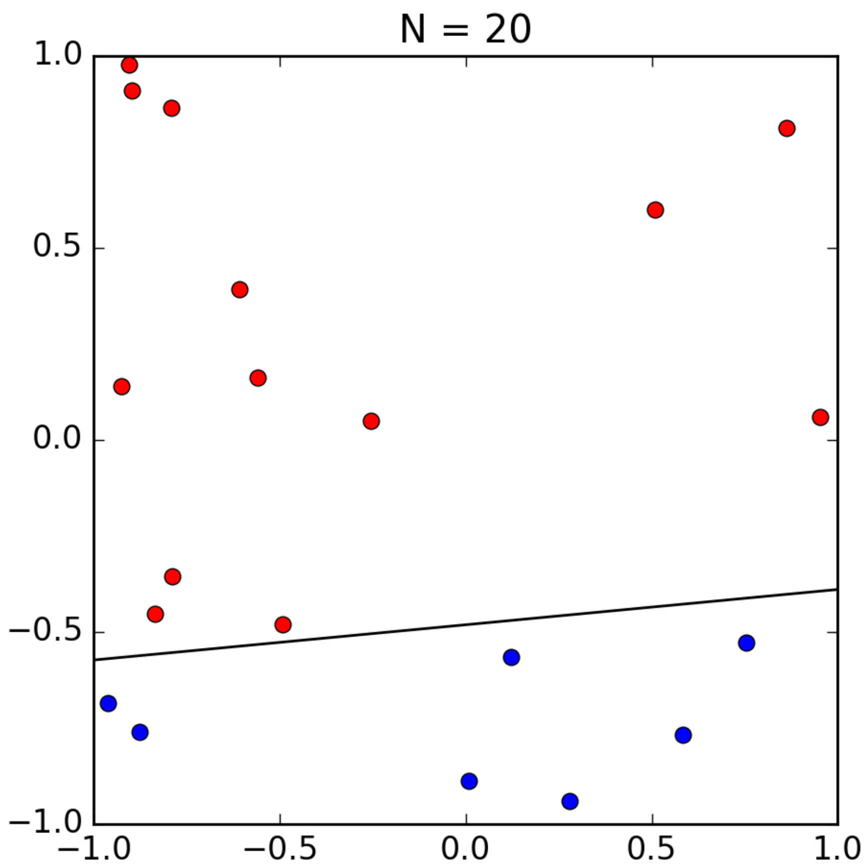

Figure 3: The final hypothesis $h(g)$.

1.4b

For this section, we ran the Perceptron algorithm again on a data set with a size of 20, taking note of the number of iterations necessary to reach $h(g)$. For this instance, we ended up with 17 iterations to reach $h(g)$. The code was modified to save all iterations as .png files:

```
p = Perceptron(20)
p.pla(save=True)
p.plot()
```

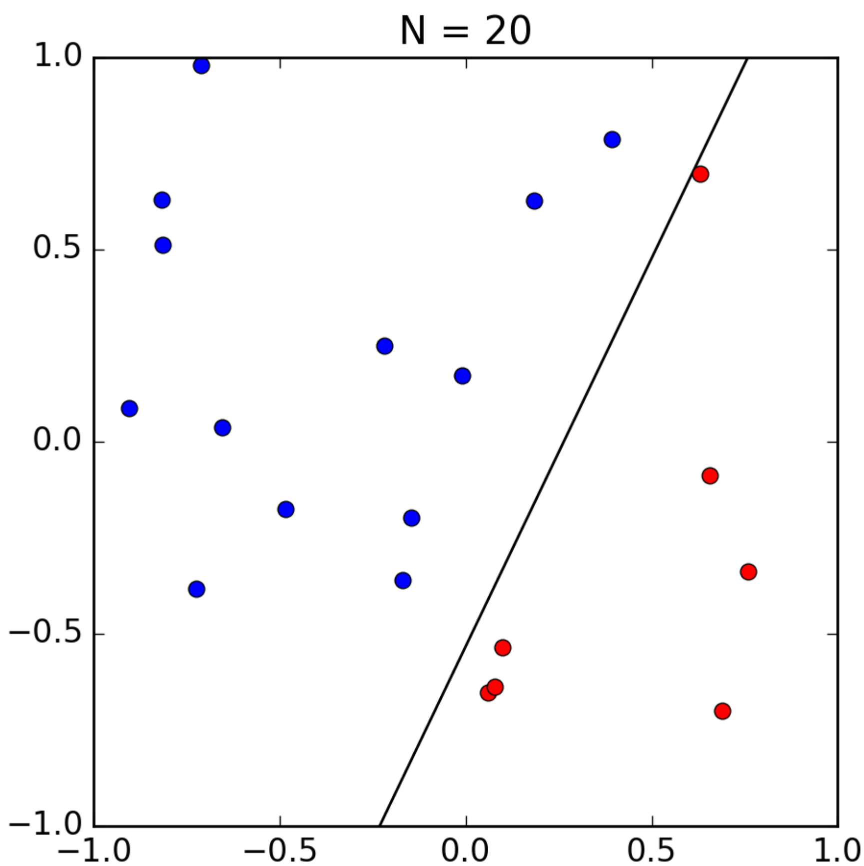

Figure 4: The final hypothesis $h(g)$ after 17 iterations.

1.4c Run the Perceptron again with a data set size of 20 and compare results with 1.4b

This time the Perceptron algorithm only required 14 iterations to find $h(g)$. The code was not modified at all for this section.

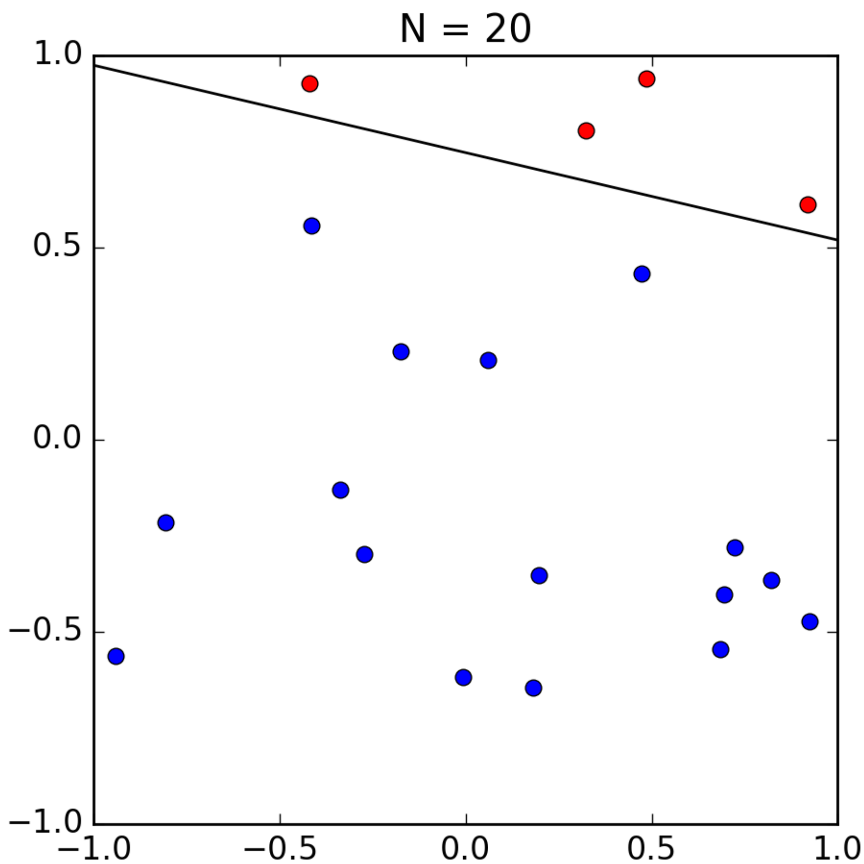

Figure 5: The final hypothesis h(g) after 14 iterations.

1.4d Run the Perceptron again with a data set size of 100 and compare results.

This time we ended up with 31 iterations and a much longer runtime to reach $h(g)$. It's looking like the runtimes get exponentially longer the more pieces of data introduced, as each entry may need to be re-checked for every iteration.

```
p = Perceptron(100)
p.pla(save=True)
p.plot()
```

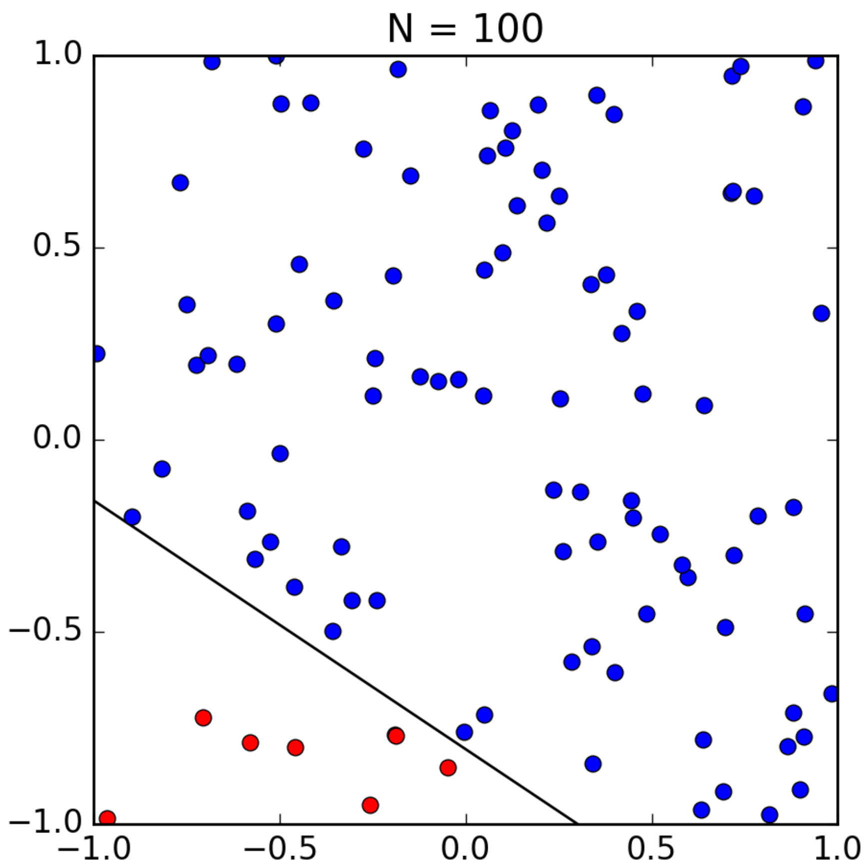

Figure 6: The final hypothesis $h(g)$ after 31 iterations.

1.4e Run the Perceptron again with data set size of 1000. Compare results

This section actually managed to freeze my computer after it had arrived at $h(g)$. It only required 49 iterations, but the runtime was significantly longer than any of the other experiments.

```
p = Perceptron(1000)
p.pla(save=True)
p.plot()
```

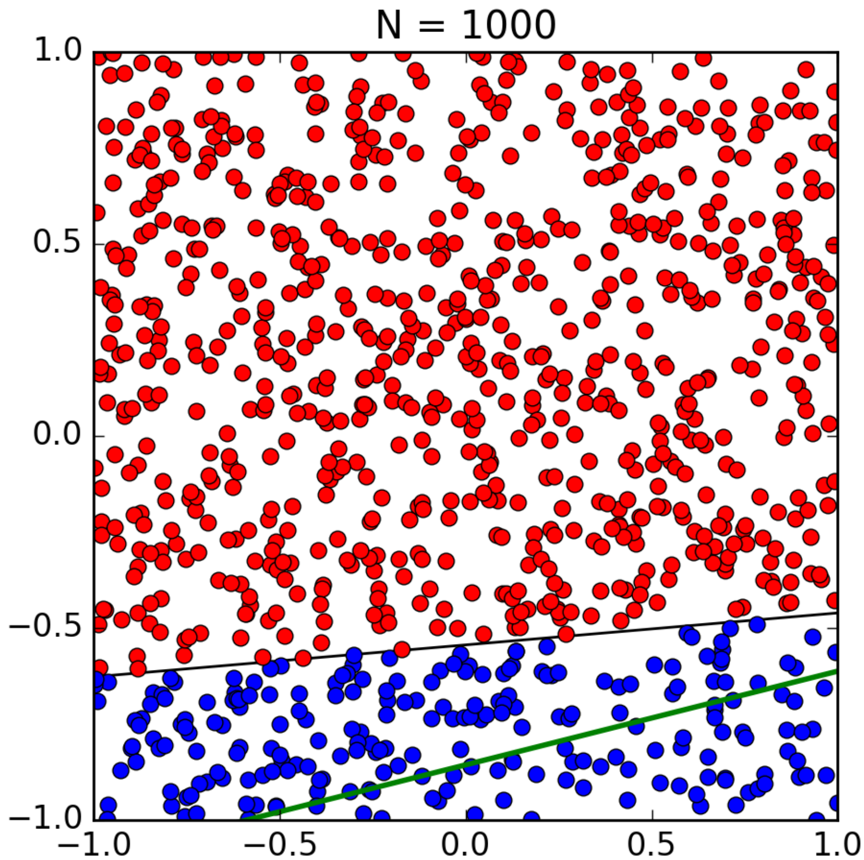

Figure 7: The final hypothesis $h(g)$ after 49 iterations.