Homework 02

Zach Stecher

Due: 9/27/16

Problem 2.1

2.1a) For $M = 1$, how many examples do we need to make $\varepsilon \leq 0.05$?

$$\varepsilon(M, N, \delta) = \sqrt{\frac{1}{2N} ln \frac{2M}{\delta}}$$

By subbing the provided numbers into the above forumla, we can begin solving for N. First we'll plug in the numbers:

$$\sqrt{\frac{1}{2N} ln \frac{2(1)}{0.03}} \leq 0.05$$

Then we math out the right side of the part of the forumla inside the square root:

$$\sqrt{\frac{1}{2N} x 4.1999} \leq 0.05$$

Then we start to isolate the N. First we square both sides of the equation to remove the square root, resulting in:

$$\frac{1}{2N} x 4.1999 \leq 0.0025$$

Then we move $\frac{1}{2N}$ to the other side and multiply by 2 to remove the fraction. Eventually we're left with:

$$\frac{4.199}{0.005} \leq N$$

So we get $N \geq 840$. We need at minimum 840 examples to achieve the desired error tolerance.

2.1b) For $M = 100$, how many examples do we need to make $\varepsilon \leq 0.05$?

Using the same equation and process, but subbing 100 in for M, we get the answer $N \geq 1761$.

2.1c) For $M = 100$, how many examples do we need to make $\varepsilon \leq 0.05$?

Using the same equation and process, but subbing 1000 in for M, we get the answer $N \geq 2683$.

Problem 2.11 Suppose $m_H(N) = N + 1$, so $d_{vc} = 1$. With 100 training examples, give a bound for E_{out} with confidence 90 percent. Repeat for $n = 10,000$.

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{2N} ln \frac{4m_h(N)}{\delta}}$$

By plugging the provided numbers into the equation, we get this (simplified):

$$\sqrt{\frac{8}{100} ln \frac{804}{0.1}}$$

By mathing this out, we get the bound (rounded up to the nearest hundreth):

$$E_{out}(g) \leq E_{in}(g) + 0.85.$$

Repeating for $N = 10,000$ we get:

$$E_{out}(g) \leq E_{in}(g) + 0.14$$

Problem 2.12 For an H with $d_vc = 10$, what sample size do you need to have a 95 percent confidence that your generalization error is at most 0.05?

$$N \geq \frac{8}{\varepsilon^2} ln(\frac{4((2N)^{d_{vc}} + 1)}{\delta})$$

Because N exists on both sides of the above equation, we can use it to iteratively solve this problem. First, we plug in the correct numbers:

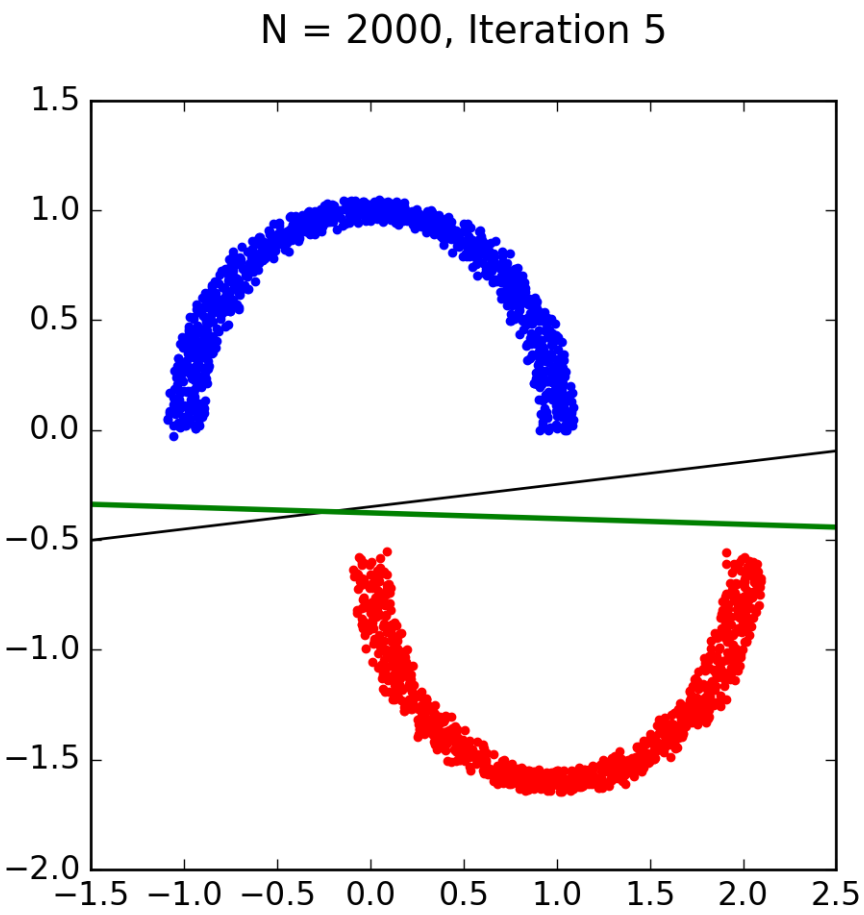
$$N \geq \frac{8}{0.05^2} ln(\frac{4((2N)^{10}+1)}{0.05})$$

Then we can begin iterating to find the answer. Let's start with N = 1,000.

$$N \geq \frac{8}{0.05^2} ln(\frac{4((2*1000)^{10}+1)}{0.05})$$

If we math this out, we get $N \geq 257251.363936$. From here, we plug $257,251.363936$ in for N in the equation we just used and do it again, arriving at $361,170.64168$. Eventually the results start evening out just below $453,000$, so we can say that $N \approx 453,000$.

Problem 3.1a Run the PLA from w=0 until it converges.



Problem 3.1b Repeat (a) using Linear Regression and note observations.

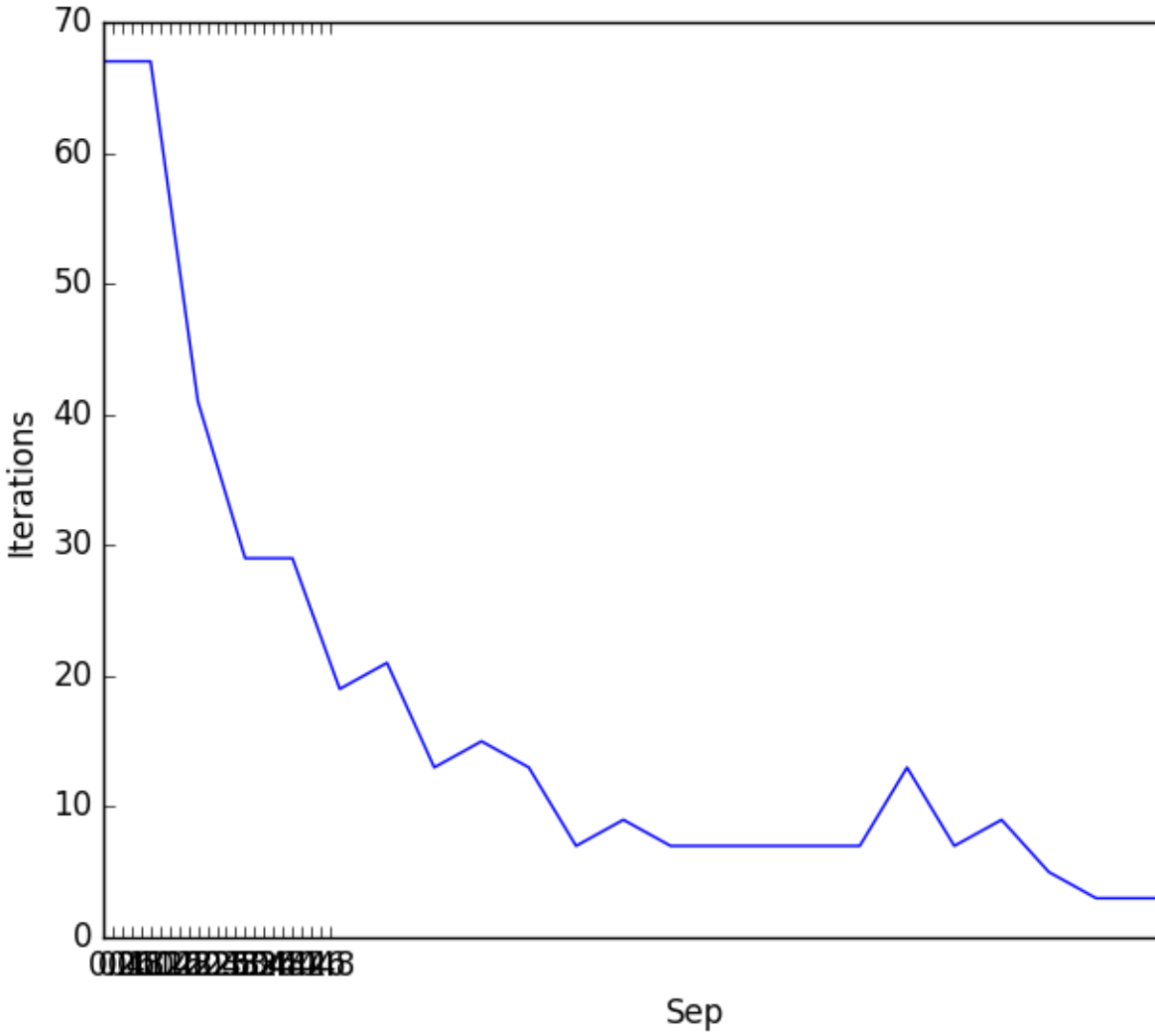
The black line in the previous graphic represents the linear regression line. This line seems to be correct from the outset as it didn't seem to change at all during the PLA iterations. I also tried running this again to see if it would change, however while our end hypothesis ended up being different, the linear regression line remained exactly the same.

Problem 3.2b For each sep in range [0.2, ..., 5] rerun the PLA task from 3.1, plot *sep* versus the number of iterations for PLA to converge.

For this problem, I modified the code as such:

```
def main():
    sepX = []
    se = 0.2
    it = np.zeros(1)
    while se <= 5:
        for x in range(0, 1):
            p = Perceptron(2000, se)
            it[x] = p.pla(save=False)
            print it
            sepX.append(it[0])
            se = se + 0.2
    print sepX
    plt.clf()
    plt.plot(sepX)
    plt.ylabel('Iterations')
    plt.xlabel('Sep')
    plt.xticks(np.arange(0.2, 5, 0.2))
plt.show()
```

This adds a variable to track each number of iterations the PLA takes. Just through observation, it became clear that the smaller the *sep* number, the higher the amount of iterations necessary to converge. I got the following plot from this experiment.


70
60
50
40
30
20
10
0
Iterations
Sep