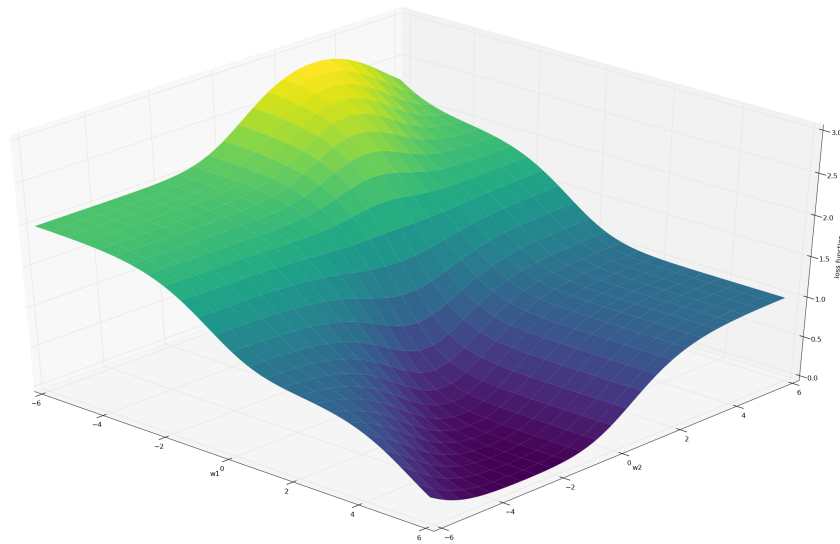


# Artificial Intelligence Methods, exercise 4

Rendell Cale

April 24, 2018

## Part A



From the plot it does not seem that there is a definite lowest point, but rather that the function decreases monotonically toward zero in the as  $w = (w_1, w_2) = (w, -w)$  and  $w \rightarrow \infty$ .

### Part A.1

### Part A.2

Using the definitions for  $\sigma(w, x)$  and  $L_{simple}(w)$ ,

$$\sigma(w, x) = \frac{1}{1 + e^{-w^T x}} \quad (1)$$

$$L_{simple} = [\sigma(w, [1, 0]) - 1]^2 + \sigma(w, [0, 1])^2 + [\sigma(w, [1, 1]) - 1]^2, \quad (2)$$

we can compute the gradient of  $L_{simple}$ ,  $\nabla_w L_{simple}$ .

We begin by writing out the expression using  $w = [w_1, w_2]^T$ .

$$L_{simple} = \left[ \frac{1}{1 + e^{-w_1}} - 1 \right]^2 + \left[ \frac{1}{1 + e^{-w_2}} \right]^2 + \left[ \frac{1}{1 + e^{-w_1 - w_2}} - 1 \right]^2 \quad (3)$$

$$= [\sigma(w_1) - 1]^2 + \sigma(w_2)^2 + [\sigma(w_1 + w_2) - 1]^2. \quad (4)$$

It is quite well known that

$$\frac{\partial \sigma(x)}{\partial x} = \sigma(x)(1 - \sigma(x)) \quad (5)$$

Using this and the chain rule, we find that

$$\frac{\partial L_{simple}}{\partial w_1}(w) = 2[\sigma(w_1) - 1] \frac{\partial \sigma(w_1)}{\partial w_1} + 2[\sigma(w_1 + w_2) - 1] \frac{\partial \sigma(w_1 + w_2)}{\partial w_1} \quad (6)$$

$$= 2(\sigma(w_1) - 1)\sigma(w_1)(1 - \sigma(w_1)) + 2(\sigma(w_1 + w_2) - 1)\sigma(w_1 + w_2)(1 - \sigma(w_1 + w_2)) \quad (7)$$

$$= -2[\sigma(w_1) - 1]^2 \sigma(w_1) - 2[\sigma(w_1 + w_2) - 1]^2 \sigma(w_1 + w_2). \quad (8)$$

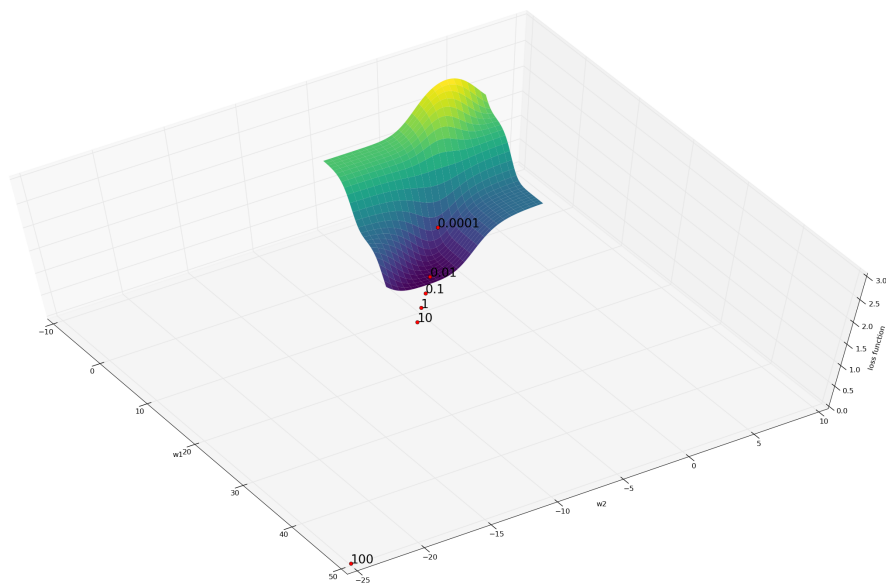
For  $w_2$  we get

$$\frac{\partial L_{simple}}{\partial w_2}(w) = 2\sigma(w_2)\sigma(w_2)(1 - \sigma(w_2)) + 2[\sigma(w_1 + w_2) - 1]\sigma(w_1 + w_2)(1 - \sigma(w_1 + w_2)) \quad (9)$$

$$= 2\sigma(w_2)^2(1 - \sigma(w_2)) - 2\sigma(w_1 + w_2)[\sigma(w_1 + w_2) - 1]^2. \quad (10)$$

### Part A.3

After implementing the update rule in Python, starting the weights at  $(w_1, w_2) = (0, 0)$ , running it for 10000 iterations and plotting the surface along with the search results for different  $\eta$ , I get:



Notice in the plot that there is quite a big difference between the different choices of  $\eta$ . With  $\eta = 100$  the weights ended up quite far away from the others, but actually the difference in the loss function values of the resulting points isn't that big when we compare  $\eta = 100$  to  $\eta = 0.01$ . The function does decrease in the direction of  $(w_1, w_2) = (\infty, -\infty)$ , but is essentially constantly zero from quite early on.

Another interesting thing we can see from this plot is that for  $\eta = 0.0001$ , the loss function is significantly larger. This is probably since the sequence converges slower for small values of  $\eta$ .

## Part B

### Part B.1 Derivative formula

Using the chain rule, we find

$$\frac{\partial L_n}{\partial w_i} = (\sigma(w, x_n) - y_n) \frac{\partial \sigma(w, x_n)}{\partial w_i} \quad (11)$$

Computing the derivative of  $\sigma(w, x_n)$  gives

$$\frac{\partial \sigma(w, x_n)}{\partial w_i} = \frac{\partial}{\partial w_i} \left( \frac{1}{1 + e^{-w^T x_n}} \right) \quad (12)$$

$$= \frac{-1}{(1 + e^{-w^T x_n})^2} \frac{\partial}{\partial w_i} (1 + e^{-w^T x_n}) \quad (13)$$

$$= \sigma(w, x_n)^2 x_i e^{-w^T x_n} \quad (14)$$

Using this in eq. (11) gives the final formula which we'll use for the remainder of the task.

$$\frac{\partial L_n}{\partial w_i} = (\sigma(w, x_n) - y_n) \frac{\partial \sigma(w, x_n)}{\partial w_i} \quad (15)$$

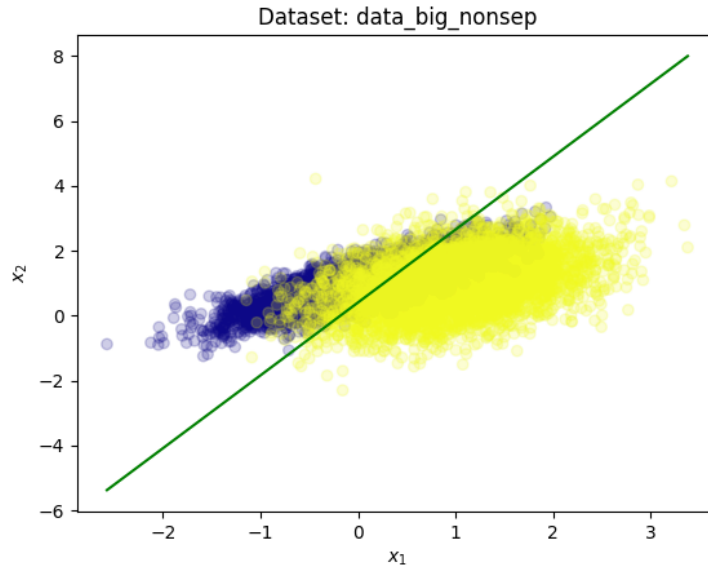
$$= x_i (\sigma(w, x_n) - y_n) \sigma(w, x_n)^2 e^{-w^T x_n} \quad (16)$$

## Part B.2 Experiments

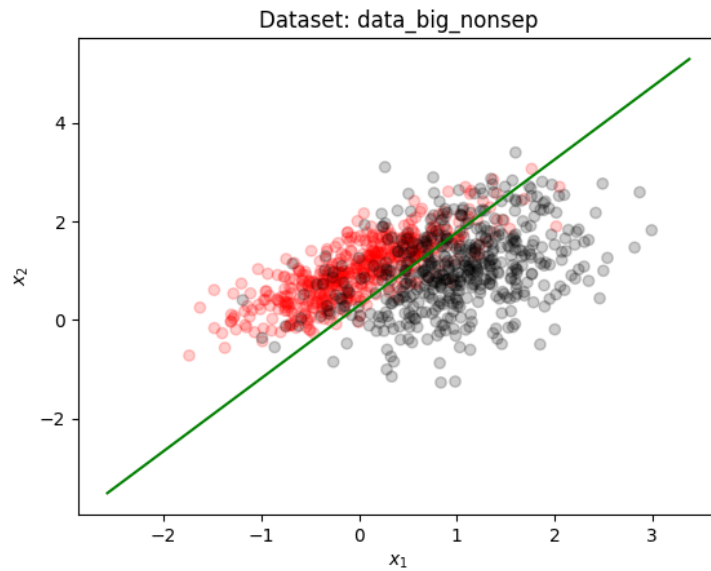
### Part B.2.1 General results

#### Part B.2.2 Scatter plot

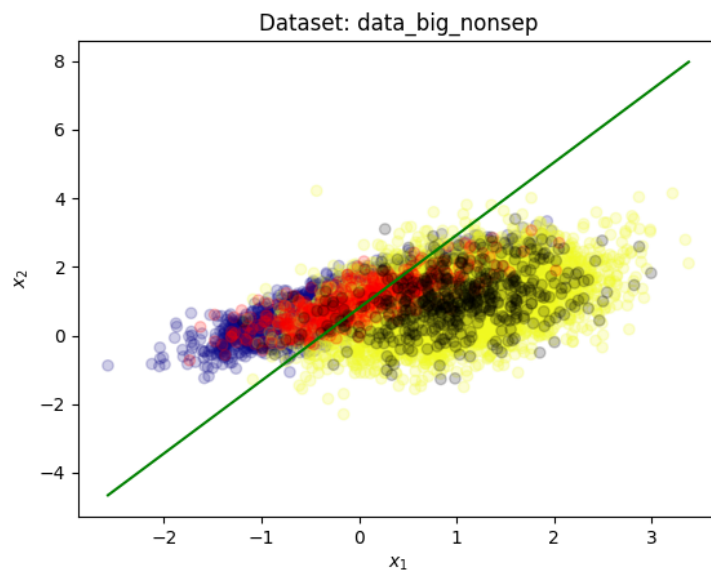
I trained the perceptron on the big non-seperable dataset using Stochastic Gradient Descent. The following scatter plot shows the two classes of points and the line which attempts to divide them.



Using the same line on the test data yields the following plot.



Combining both plots give the following.



### Part B.2.3 Effects of iterations

The general effect that increasing the number of iterations is that the error goes down, but only to a certain point. For the seperable datasets it was quite easy to get the error down to 0, which makes sense, but for the non-seperable datasets

the error will converge to something non-zero and bounce slightly up or down for each iteration.

To see this we will plot the error as a function of the number of iterations for the small non-separable dataset.