Lab 4: Perceptron Classification

Brian Brenner

Learning Algorithm

$$o^j = f(\sum_{i=1}^{n} w_i x_i) \text{ where } f(\theta) = \begin{cases} 1, & \text{if } \theta > 0. \\ 0, & \text{if } \theta \leqslant 0. \end{cases}$$

- Distance from correct y value which we define

$$d^j = y - o^j$$

- We use this distance to update the weights

$$\Delta W_i = \ell d^j x_i^j \text{ for } i = 1...n, w_i = w_i + \Delta w_i$$

- L is the learning rate, which lets us control how quickly the weights change
- Eventually we want the weights to be adjusted so that d will be minimized and our output will be the same as our desired y value

Three Classification Problems

- AND:

x_1	x_2	y
1	1	1
1	0	0
0	1	0
0	0	0

- OR:

x_1	x_2	y
1	1	1
1	0	1
0	1	1
0	0	0

- XOR:

x_1	x_2	y
1	1	0
1	0	1
0	1	1
0	0	0

Functions

- pdrive, driver function which calls perceptron

```
[w, w0] = perceptron(x,y)
```

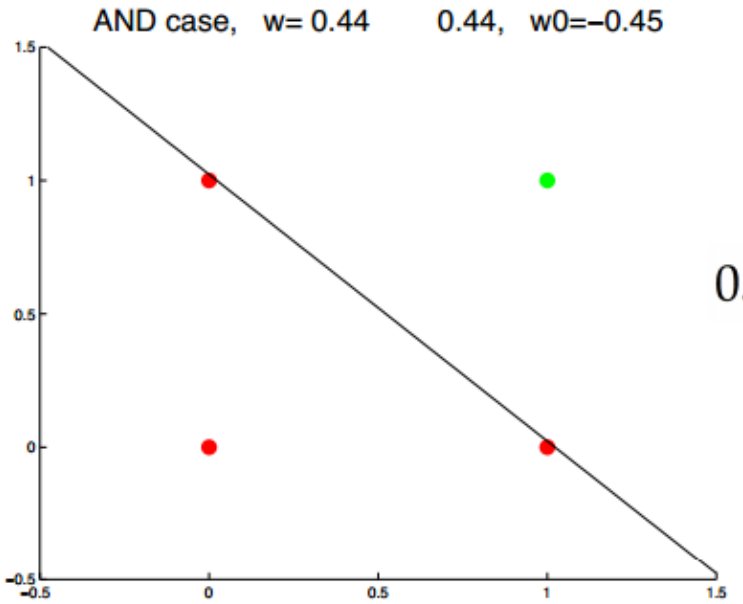
```
x =  1 1        y = 1
     1 0            0
     0 1            0
     0 0            0
```

- Input x which is set of inputs, remains constant across the different problems
- We change y to reflect different problems
- Outputs the weights that our learning algorithm found

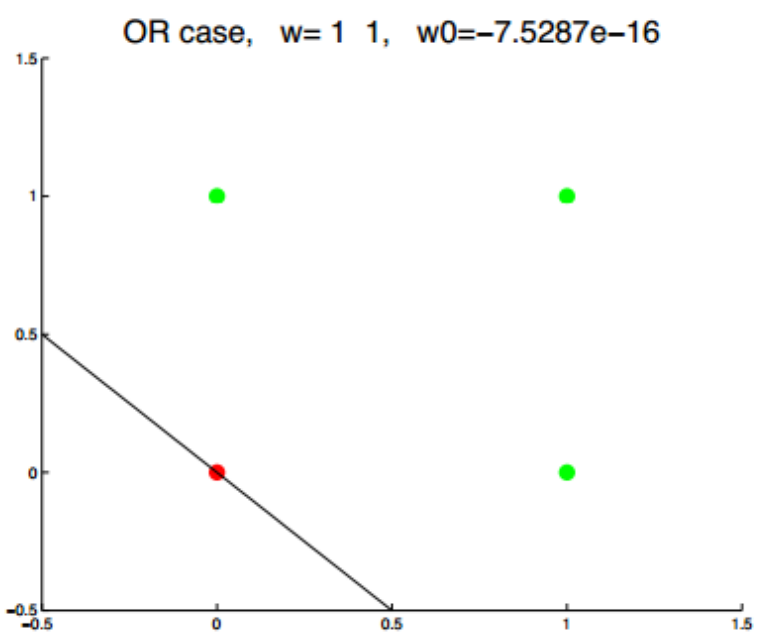
Graphs

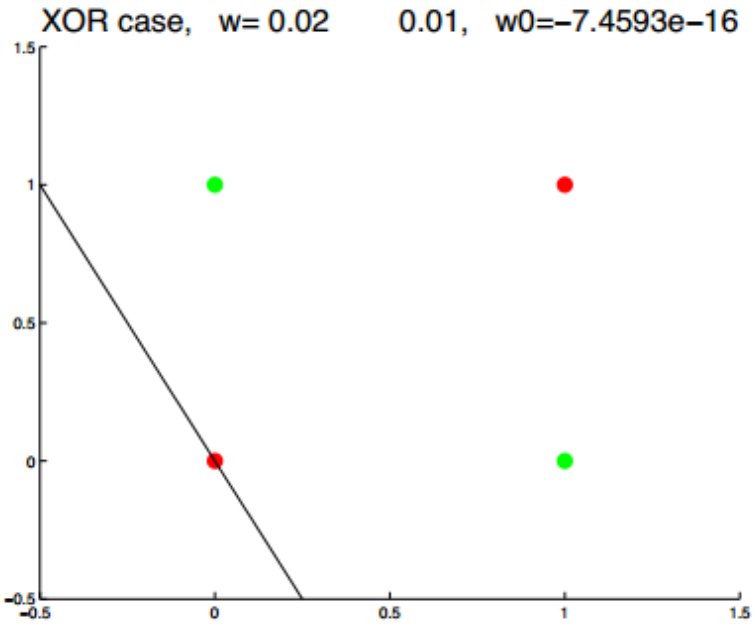
- Use outputted weights to make a plot

$$w_1 x_1 + w_2 x_2 - \theta = 0$$



$$0.44x_1 + 0.44x_2 - 0.45 = 0$$





- Use scatter to plot the points