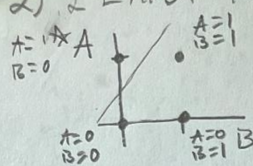
I did problems 2,3,5,6,7

2: Design a two-input perceptron that implements the boolean function $A \wedge \neg B$. Design a two-layer network of perceptrons that implements $A \oplus B$ (where $\oplus$ is XOR).

2) 2 Input Perceptron for $A \cap \overline{B}$



$w_a = 1$

$w_b = -1$

$\theta = 0$

$w_a \cdot A \geq \theta$

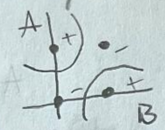
$1 \cdot 1 \geq 0$ ✓

$w_b \cdot B \leq \theta$

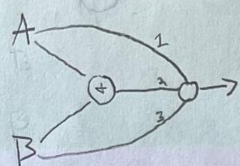
$-1 \cdot 1 \leq 0$

A	B	$w_a A$	$w_b B$	out
0	0	0	0	0
1	0	1	0	1
0	1	0	-1	0
1	1	1	-1	0

2 layer network of Perceptrons $A \oplus B$



$w_1 \cdot A + w_2 \cdot A \cap B + w_3 \cdot B \geq \theta$



$w_1 = 1$

$w_2 = -3$

$w_3 = 1$

$\theta = 0 \angle x \angle 1$

A	B	+	$w_1 A$	$w_2 +$	$w_3 B$	Σ	out
0	0	0	0	0	0	0	0
0	1	0	0	0	1	1	1
1	0	0	1	0	0	1	1
1	1	1	1	-3	1	-1	0

3: Derive the perceptron training rule and gradient descent training rule for a single unit with output o, where $o = w_0 + w_1x_1 + w_1x_1^2 + \ldots + w_nx_n + wnx_n^2$. What are the advantages of using gradient de- scent training rule for training neural networks over the perceptron training rule?

t = target

η = learning rate

3) Perceptron rule: $w_i \leftarrow w_i + \Delta w_i$ where $\Delta w_i = \eta(t-o)x_i$

Gradient rule: $\frac{\partial E}{\partial w_i} = \sum_{d \in D}(t_d - o_d)(-x_{id})$

where $o = w_0 + w_1x_1 + w_1x_1^2 \ldots w_nx_n + w_nx_n^2$

Perceptron: we need to adjust the weight up or down depending on the output, thus giving us "t-o" so we can compare then we need to adjust by our learning rate $\eta(t-o)$ & take into account the x_i giving us the change in the weight $\Delta w_i = \eta(t-o)x_i$

Gradient: start w/ $\frac{1}{2}\sum(t_d - o_d)^2 = E(\vec{w})$ we need to find $\frac{\partial E}{\partial w_i} =$

$$= \frac{\partial}{\partial w_i}\frac{1}{2}\sum(t_d - o_d)^2$$

$$= \frac{1}{2}\sum\frac{\partial}{\partial w_i}(t_d - o_d)^2$$

$$= \frac{1}{2}\sum 2(t_d - o_d)\frac{\partial}{\partial w_i}(t_d - o_d)$$

$$= \sum(t_d - o_d)\frac{\partial}{\partial w_i}(t_d - \vec{w}\vec{x_i})$$

$$= \sum(t_d - o_d)(0 - 1\vec{x_i})$$

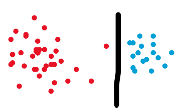
$$= \sum(t_d - o_d)(-\vec{x_i})$$

5: Suggest a lazy version of the eager decision tree learning algorithm ID3. What are the advantages and disadvantages of your lazy algorithm compared to the original eager algorithm?

A lazy version of ID3 could be where the tree isn't created at training time, but created with each value we try to predict on. So we aren't concerned with where the other points go, we only care about our single point and try to increase the purity of that node with each split. Instead of a tree looking algorithm we could call it the big stick algorithm as it will be one looooooong line of splitter points. This would be EXTREMELY computationally expensive though as some trees need a big depth to reach optimal purity. For tress that have a small depth though this wouldn't be bad.

6: Imagine you had a learning problem with an instance space of points on the plane and a target function that you knew took the form of a line on the plane where all points on one side of the line are positive and all those on the other are negative. If you were constrained to only use decision tree or nearest-neighbor learning, which would you use? Why?

I would use a decision tree. A decision tree works well with linearly separable data, that doesn't deal with proximity. The decision tree would be able to form those splitting boundaries quite well, as apposed to the nearest neighbor model. I wouldn't choose a nearest neighbor model because even though the split creates pure boundaries, we could have one point that is very close to the line, yet far away from the main cluster but extremely close to the other cluster of the opposite classification. Something like below:



7: Give the VC dimension of these hypothesis spaces, briefly explaining your answers:

1. An origin-centered circle (2D)
   The VC dimension will be 3, when we are in 2D we deal with parameters of w and theta where w is in 2 dimensions or a vector with 2 rows. The VC dimension is the number of parameters in which the hypothesis lives or the number of your dimension + 1

2. An origin-centered sphere (3D)
   The VC dimension is 4, see above