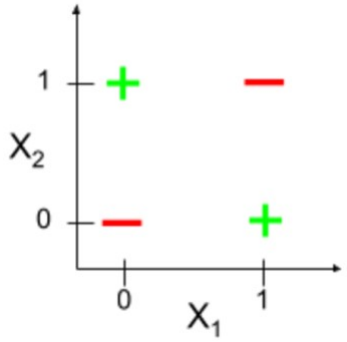
CSE516/ECE559: Theories of Deep Learning
Problem Sheet 1

Dr. Vinayak Abrol

September 6, 2023

- XOR Problem: The truth table for this problem and a graphical illustration is given below. XOR (exclusive OR) in digital electronics denotes the addition operations on bits e.g., 1+1=2 in decimal but 10 in binary representation, and since y here has single bit precision $y = 0$.



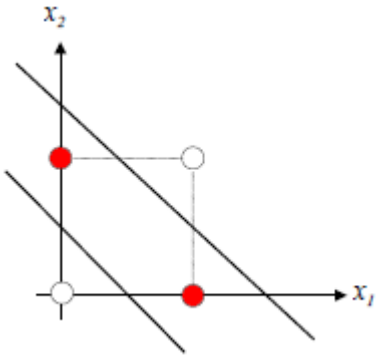
x1	x2	y
0	1	1
1	0	1
0	0	0
1	1	0

It can be observed that it is not possible to draw a linearly separating hyperplane (a line in two dimensions) which can classify these points in two classes.

1. Solution1: Project to higher dimensional space- Consider the modified truth table in 3-dimensions below. Here, $x3 = y^{'}$ i.e., complement of target output y.
   Notice that you can now draw a 2-dimensional plane to linearly separate the two classes in 3-dimensions.

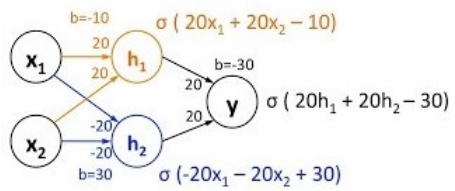
x1	x2	x3	y
0	1	0	1
1	0	0	1
0	0	1	0
1	1	1	0

   This idea can be extended for higher dimensional XOR problem or even for any classification problem where the goal is to find a suitable transformation in higher dimensions where data is linearly separable.
2. Solution2: Use a single hidden layer neural network with two neurons to get the classification.



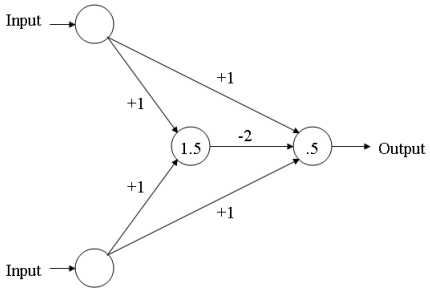
$h1 = a_{11}x_1 + a_{12}x_2 + b1$
$h2 = a_{21}x_1 + a_{22}x_2 + b2$
$y = c_1h_1 + c_2h_2 + b3$

$h1, h2$ are outputs of two nodes in hidden layer and define two approx. parallel hyper-planes. Note that the hidden layer is still 2-dimensional and we are able to solve the problem without going to higher dimensional space.



One such configuration using sigmoid activation is shown in figure with network parameters.

3. Solution3: My favourite! Think differently. Let's use a single hidden neuron with a threshold on output as shown in figure.



$y = \sigma(x_1 + x_2 - c\sigma(x_1 + x_2))$
σ denotes activation above a threshold. Notice that this configuration also solves the problem in 3-dimensions and is motivated from solution 1. The key point to note here is that when we added the 3rd dimension in solution (a) it was dependent on output y which depends on inputs $x1, x2$, hence, we can directly manipulate the inputs via just a single neuron instead of two as in solution (b) shown above.

- Solution 2(a)
  Let $y_1 = x^2$ and a linear approximation $y_2 = ax + b$ which interpolates y_1 at points x_0 and x_1. Hence,

$$\begin{aligned} x_0^2 &= ax_0 + b \quad \implies a = x_1 + x_0; b = -x_0x_1 \\ x_1^2 &= ax_1 + b \end{aligned}$$

  The ℓ_∞ error in interval $[x_0, x_1]$ is

$$y_1 - y_2 = x^2 - (x_1 + x_0)x + x_0x_1$$

  The maximizer of this loss is computed as

$$\begin{aligned} \frac{\partial(y_1 - y_2)}{\partial x} &= 2x - (x_1 + x_0) = 0 \\ x^* &= \frac{(x_1 + x_0)}{2} \end{aligned}$$

  and thus the ℓ_∞ loss is

$$(y_1 - y_2)|_{x^*} = \frac{-(x_1 + x_0)^2}{4} + x_0x_1$$

Now let's consider two adjacent points $x_0 = k/2^m$ and $x_1 = (k+1)/2^m$ to compute the maximum loss and we have

$$\begin{aligned}
&\Big| -\frac{(2k+1)^2}{4(2^{2m})} + \frac{(k^2+k)}{2^{2m}} \Big| \\
&= \Big| -\frac{1}{4(2^{2m})} \Big| = \Big| -2^{-2m-2} \Big| = 2^{-2m-2}
\end{aligned}$$