09/02/24 Neural Networks Page (1) 675086474

1) a) Given:- For some j, $w_{ji} = 1$ & $b_j = 0$

To prove:- z_j is equivalent to logical OR of x_i

Inference:-

$$z_j = \text{step}_1\left(b_j + \sum_{i=1}^{q} w_{ji} x_i\right) \quad \text{step}_1(t) = \begin{cases} 0, & t < 1 \\ 1, & t \geq 1 \end{cases}$$

Since $b_j = 0$ & $w_{ji} = 1$

$$\boxed{z_j = \text{step}_1\left(\sum_{i=1}^{q} x_i\right)}$$

According to the step fn if there is atleast one $x_i = 1$ then $z_j = 1$, or if no $x_i = 1$, then $z_j = 0$.

Proof:-

i) Let us consider $x_4 = 1$ and rest as zero.

Thus $\sum_{i=1}^{q} x_i = 0+0+0+1+0+0+0+0+0+0$

$= 1$

$z_j = \text{step}_1(1) = 1$

$\boxed{z_j = 1}$

ii) Consider all x_i as zeros,

Thus $\sum_{i=1}^{q} x_i = 0+0+0+0+0+0+0+0+0+0$

$= 0$

$z_j = \text{step}_1(0) = 0$

$\boxed{z_j = 0}$

Thus with given condition it is logical OR.

b) Given :- If $w_{ji} = -1$ and $b_j = N$ (no. of such weights)

To prove :- z_j is inverted in these cases.

Inference :-

Consider $b_j = N$, where N is the no. of such inverted weights.

So, $z_j = \text{Step}_1\left(N + \sum_{i=1}^{9} w_{ji} x_i\right)$

Case 1 :- For $x_i = 1$, $w_{ji} = 1$, $b_j = 0$

$$z_j = \text{Step}_1\left(0 + \sum_{i=1}^{9} x_i\right) = \text{Step}(1)$$

$$= 1$$

$\boxed{z_j = 1}$ For unit weights. works as logical OR.

Case 2 :-

For $x_i = 0$, $w_{ij} = 1$, $b_j = 0$

$$z_j = \text{Step}_1\left(0 + \sum_{i=1}^{9} x_i\right)$$

$$= \text{Step}(0)$$

$$= 0$$

$\boxed{z_j = 0}$ works as logical OR

Case 3 :- Here $w_{ij} = -1$, $b_j = N$

$$z_j = \text{Step}_1\left(N + \sum_{i=1}^{9} w_{ij} x_i\right)$$

Say, consider $x_2 = 1$, $w_{2j} = -1$

then $b_j = 1$

$$z_j = \text{Step}_1\left(1 + \sum_{i=1}^{9} w_{ij} x_i\right)$$

for, $\sum_{i=1}^{9} w_{ij} x_i = 0 + (-1)(1) + 0 + 0 + 0 + 0 + 0 + 0 + 0$

$= -1$

$= \text{Step}_1(1 - 1) = \text{Step}_1(0)$

$z_j = 0.$

So if atleast one $x_i = 1$ and its weight is negative, it acts as a Inverted logical OR

Case 4:- Here $w_{ij} = -1$, $b_j = N$

Say, consider $x_2 = 0$, $w_{ij} = -1$

Then $b_j = 1$

$$z_j = \text{Step}_1\left(1 + \sum_{i=1}^{9} w_{ij} x_i\right)$$

for, $\sum_{i=1}^{9} w_{ij} x_i = 0 + \overset{(-1)}{(0)} + 0 + 0 + 0 + 0 + 0 + 0 + 0$

$= 0$

$= \text{Step}_1(1 + 0) = \text{Step}(1)$

$z_j = 1$

So, if the weight is negative and all $x_i = 0$ then, it acts as a Inverted logical OR

Thus when weight is negative it acts as a Inverted logical OR

c) Case 1:

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}_{3\times 3}$$

$$z_1 = \text{not}(x_1 \cdot x_2 \cdot \overline{x_3} \cdot x_4 \cdot x_5 \cdot \overline{x_6} \cdot \overline{x_7} \cdot \overline{x_8} \cdot \overline{x_9})$$

using DeMorgan's law

$$= (\overline{x_1} + \overline{x_2} + x_3 + \overline{x_4} + \overline{x_5} + x_6 + x_7 + x_8 + x_9)$$

(using inverted logical or)

$$= (w_{11}\overline{x_1} + w_{12}\overline{x_2} + w_{13}x_3 + w_{14}\overline{x_4} + w_{15}\overline{x_5} + w_{16}x_6 + w_{17}x_7 + w_{18}x_8 + w_{19}x_9)$$

where $w_{ij} = [-1, -1, 1, -1, -1, 1, 1, 1, 1]$

and $b = 4$

According to fn

$$z_1 = 4 + \sum_{i=1}^{9} x_j w_{ij}$$

$$= 4 + (-1 - 1 + 0 - 1 - 1 + 0 + 0 + 0 + 0)$$

$$= 4 + (-4)$$

$$z_1 = 0$$

Case 2:

$$\begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}_{3\times 3}$$

$$z_2 = \text{not}\left(\bar{x}_1 \cdot x_2 \cdot x_3 \cdot \bar{x}_4 \cdot x_5 \cdot x_6 \cdot \bar{x}_7 \cdot \bar{x}_8 \cdot \bar{x}_9\right)$$

using De Morgan law

$$= x_1 + \bar{x}_2 + \bar{x}_3 + x_4 + \bar{x}_5 + \bar{x}_6 + x_7 + x_8 + x_9$$

using inverted logical or

$$= (w_{11}x_1 + w_{12}\bar{x}_2 + w_{13}\bar{x}_3 + w_{14}x_4 + w_{15}\bar{x}_5 + w_{16}\bar{x}_6 + w_{17}x_7 + w_{18}x_8 + w_{19}x_9)$$

when $= w_{ij} = [1, -1, -1, 1, -1, -1, 1, 1, 1]$

and $b = 4$

According to f_4

$$z_2 = 4 + \sum_{i=1}^{9} x_i w_{ij}$$

$$= 4 + (0 - 1 - 1 + 0 - 1 - 1 + 0 + 0 + 0)$$

$$= 4 + (-4)$$

$z_2 = 0$

$z_2 = 0$

Case 3:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}_{3\times 3}$$

~~$z_3 = \text{not}(\bar{x}_1 \cdot \bar{x}_2 \cdot \bar{x}_3 \cdot \bar{x}_4 \cdot x_4 \cdot x_5 \cdot x_6$~~

$$z_3 = \text{not}\left(\bar{x}_1 \cdot \bar{x}_2 \cdot \bar{x}_3 \cdot \bar{x}_4 \cdot x_5 \cdot x_6 \cdot \bar{x}_7 \cdot x_8 \cdot x_9\right)$$

using Demorgan laws

$$= \left(x_1 + x_2 + x_3 + x_4 + \bar{x}_5 + \bar{x}_6 + x_7 + \bar{x}_8 + \bar{x}_9\right)$$

using inverted logic gates

$$= w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + w_{14}x_4 + w_{15}\bar{x}_5 + w_{16}\bar{x}_6 + w_{17}x_7 + w_{18}\bar{x}_8 + w_{19}\bar{x}_9)$$

where $w_{ij} = [1, 1, 1, 1, -1, -1, 1, -1, -1]$

According to fn

$$z_3 = 4 + \sum_{i=1}^{9} x_i w_{ij}$$

$$= 4 + (\text{~~4~~}) - 4$$

~~$z_3 = 4$~~ $\quad z_3 = 0$

Case 4:

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}_{3\times 3}$$

$$Z_4 = \text{not}\left(\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot x_4 \cdot x_5 \cdot \overline{x_6} \cdot x_7 \cdot x_8 \cdot \overline{x_9}\right) \quad \text{(using demorgan laws)}$$

$$= (x_1 + x_2 + x_3 + \overline{x_4} + \overline{x_5} + x_6 + \overline{x_7} + \overline{x_8} + x_9)$$

$\Rightarrow$ (using inverted logical or)

$$= w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + w_{14}\overline{x_4} + w_{15}\overline{x_5} + w_{16}x_6 + w_{17}\overline{x_7} + w_{18}\overline{x_8} + w_{19}\overline{x_9}$$

where $w_{ij} = [1, 1, 1, -1, -1, 1, -1, -1, 1]$

According to y_4

$$z_4 = 4 + \sum_{i=1}^{9} x_{4i}$$

$$= 4 + (-4)$$

$$\boxed{z_4 = 0}$$

$$w_{ji} = \begin{bmatrix} -1 & -1 & 1 & -1 & -1 & 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & -1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & 1 & -1 & -1 \\ 1 & 1 & +1 & -1 & -1 & +1 & -1 & -1 & 1 \end{bmatrix}$$

$$z_4 = [0, 0, 0, 0]$$

$$b = [4, 4, 4, 4]$$

$= z_1 \cdot z_2 \cdot z_3 \cdot z_4$

apply demorgan law

$= \overline{z_1} + \overline{z_2} + \overline{z_3} + \overline{z_4}$

So here

bias = 4

v_j = weight = $[-1, -1, -1, -1]$

$C = 4$

2) a) $g(x) = x^3$

$f(x, w) = wx$

given composite fn

$$l(f(x,w), g(x)) = (f(x,w) - g(x))^2$$

chain rule :-

$$\boxed{\frac{\partial l}{\partial w} = \frac{\partial l}{\partial f} \cdot \frac{\partial f}{\partial w}}$$

b) In chain rule,

$$\frac{\partial l}{\partial w} = \frac{\partial l}{\partial f} \cdot \frac{\partial f}{\partial w}$$

where

$$\frac{\partial l}{\partial f} = 2\left(f(x,w) - g(x)\right) = 2(wx - x^3)$$

$$\frac{\partial f}{\partial w} = x$$

Substitute in chain rule

~~$\frac{\partial l}{\partial w} = 2(f(x,w)$~~

$$\frac{\partial l}{\partial w} = 2(wx - x^3) \cdot x$$

$$= 2x(wx - x^3)$$

$$= 2(wx^2 - x^4)$$

So now integrate to $\frac{dL}{dw}$

$$\frac{dL}{dw} = \int_{-1}^{1} 2(wx^2 - x^4)\,dx$$

$$= \int_{-1}^{1} (2wx^2 - 2x^4)\,dx$$

$$= \int_{-1}^{1} 2wx^2 \cdot dx - \int_{-1}^{1} 2x^4 \cdot dx$$

$$= 2w\int_{-1}^{1} x^2\,dx - 2\int_{-1}^{1} x^4\,dx$$

Case 2. 2 & w is constant

$$= 2w\left[\frac{x^3}{3}\right]_{-1}^{1} - 2\left[\frac{x^5}{5}\right]_{-1}^{1}$$

$$= 2w\left[\frac{1}{3}\right] - 2w\left[\frac{-1}{3}\right] - \left[2\left[\frac{1}{5}\right] - 2\left[\frac{-1}{5}\right]\right]$$

$$= \frac{2w}{3} + \frac{2w}{3} - \left[\frac{2}{5} + \frac{2}{5}\right]$$

$$\frac{dL}{dw} = \frac{4w}{3} - \frac{4}{5}$$

To find optimal of w

$$\frac{dL}{dw} = 0$$

$$\frac{4w}{3} - \frac{4}{5} = 0$$

$$\frac{20w - 12}{15} = 0$$

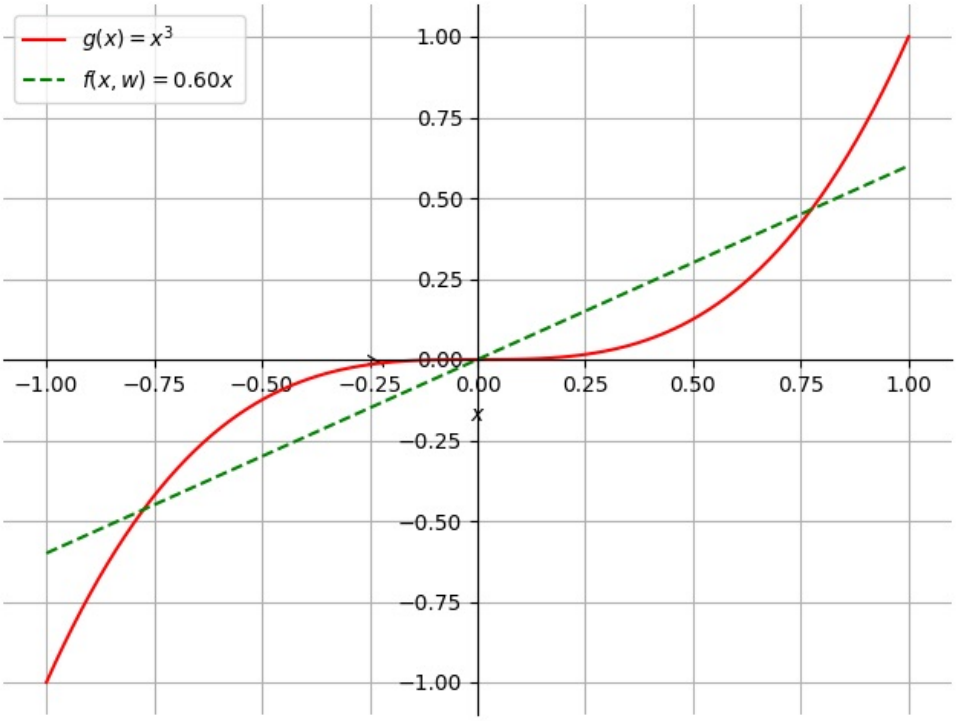
$$20w - 12 = 0$$

$$20w = 12$$

$$w = \frac{12}{20} = \frac{3}{5}$$

$$w = 3/5$$

2) c) graph plotted.



①P

NN - HW2

Grakhulnath Thirumaran | 675086474

1) a) Given: To design a feed fwd NN for the below logical statement.

$$(x_1 \wedge x_2 \wedge \neg x_3) \vee (\neg x_2 \wedge x_3)$$

+1 ⇒ True -1 ⇒ False

$$\varphi(v) = sign(v) = \begin{cases} +1 & ; \ v > 0 \\ 0 & ; \ v = 0 \\ -1 & ; \ v < 0 \end{cases}$$

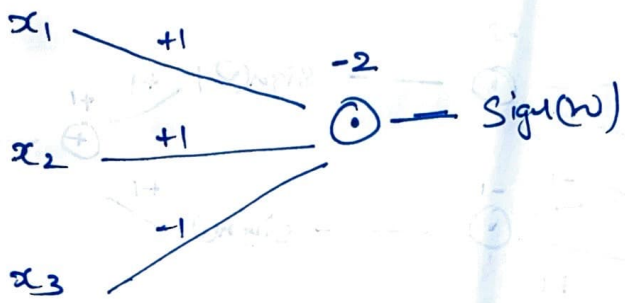
Inference:

$$(x_1 \wedge x_2 \wedge \neg x_3) \vee (\neg x_2 \wedge x_3)$$

↓ ① ↓ ②

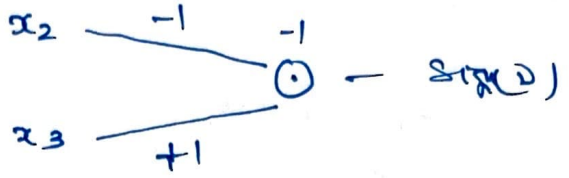
① ⇒ $(x_1 \wedge x_2 \wedge \neg x_3)$

We can choose the $w = [1, 1, -1]$ since x_3 is a not operation and x_1, x_2 is a And operation. Let the bias $b_1 = -2$, so that it activates only when it satisfies the logical statement.

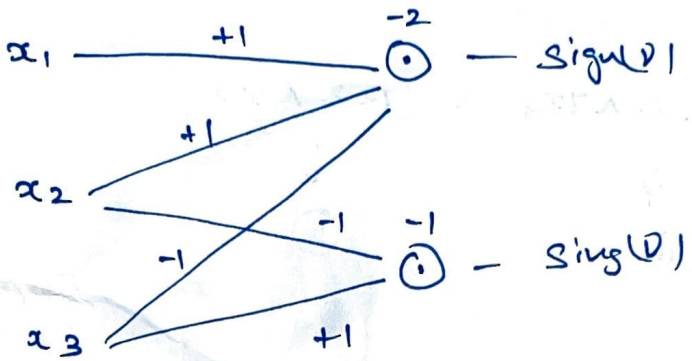


(2) $\Rightarrow (\neg x_2 \wedge x_3)$

Similarly we assign the weights as $[-1, 1]$ and bias $b_2 = -1$ so that it activates satisfying the logical statement.



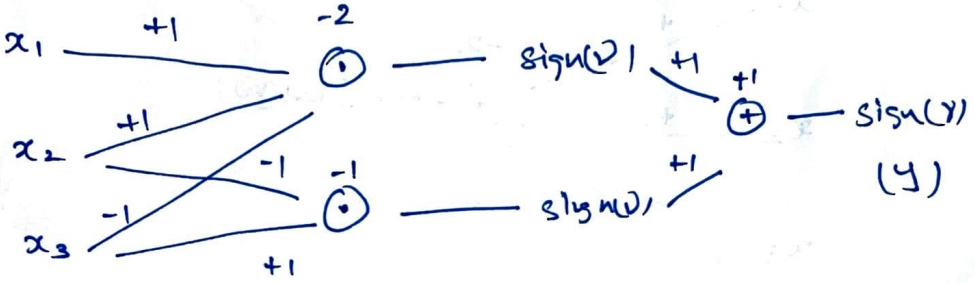
Combining (1) & (2)



Second layer:- Since its an OR operation, we need to make sure that it satisfies the condition for the given bias & weight. Thus we choose

$$\text{weight} = [1, 1]$$

$$\text{bias} = +1$$



(3) P

Thus we have 3 — input

2 — hidden layer

1 — output

3-2-1 NN.

b) First layer (hidden)

$$z_j = \text{sign}\left(\sum_{i=1}^{3} w_{ji} x_i + b_j\right) - (1)$$

Second layer (output)

$$y = \text{sign}\left(\sum_{i=1}^{2} v_{ji} z_j + c_j\right) - (2)$$

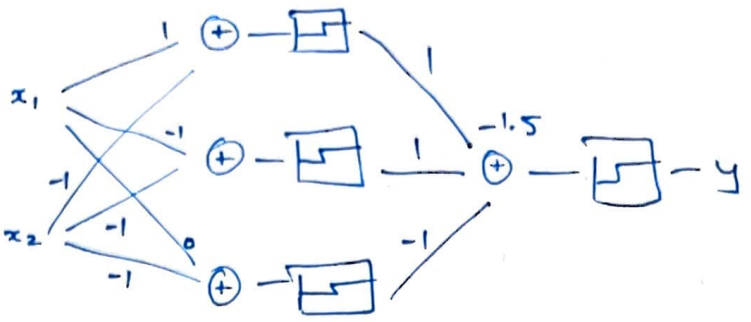
(2) in (1)

$$y = \text{sign}\left(\sum_{i=1}^{2} v_{ji} \left(\text{sign}\left(\sum_{i=1}^{3} w_{ji} x_i + b_j\right)\right) + c_j\right)$$

c)

x1	x2	x3	Logical	NN
1	1	1	False	-1
1	1	-1	True	1
1	-1	1	True	1
1	-1	-1	False	-1
-1	1	1	False	-1
-1	1	-1	False	-1
-1	-1	1	True	1
-1	-1	-1	False	-1

2) a) Given: (4p)



$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_{2\times1} \quad \text{input}$$

$$w = \begin{bmatrix} 1 & -1 \\ -1 & -1 \\ 0 & -1 \end{bmatrix}_{3\times2} \quad \text{weight}$$

$$b = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}_{3\times1} \quad \text{bias}$$

$$v = \begin{bmatrix} 1 & 1 & -1 \end{bmatrix}_{1\times3} \qquad c = \begin{bmatrix} -1.5 \end{bmatrix}_{1\times1}$$

b) $$z_j = \text{step}\left(\sum_{i=1}^{2} w_{ji} x_i + b_j \right) \quad -(1)$$

$$y = \text{step}\left(\sum_{i=1}^{1} v_{ji} z_j + c_i \right) \quad -(2)$$

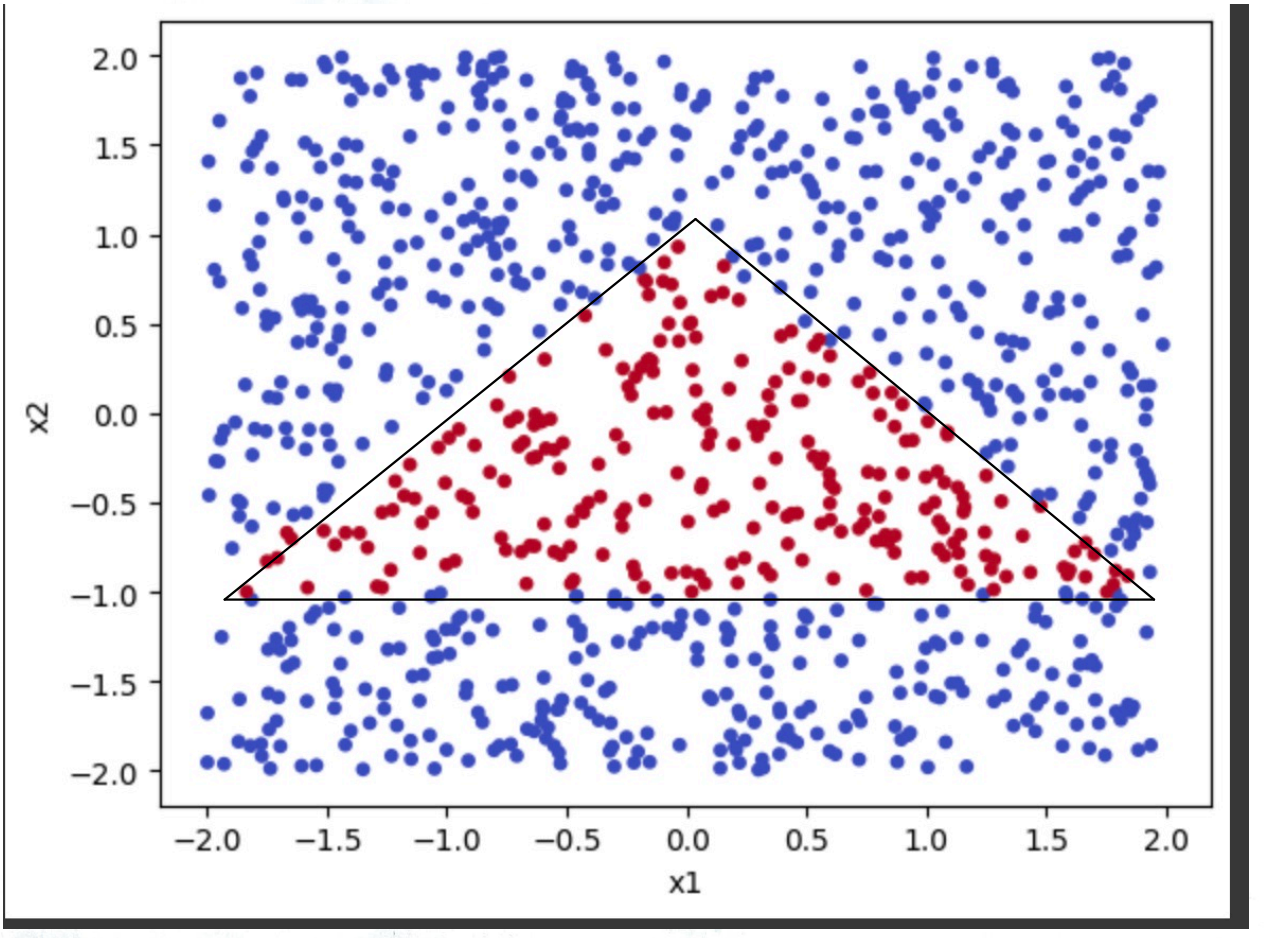
(2) in (1)

$$y = \text{step}\left(\sum_{i=1}^{1} v_{ji} \left(\text{step}\left(\sum_{i=1}^{2} w_{ji} x_i + b_j \right) \right) + c_j \right)$$

$$y = \text{step}\left(\begin{bmatrix} 1 & 1 & -1 \end{bmatrix} \left(\text{step}\left(\begin{bmatrix} 1 & 1 \\ -1 & -1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} \right) \right) + \begin{bmatrix} -1.5 \end{bmatrix} \right).$$

c) code submitted

d)



09/17/24

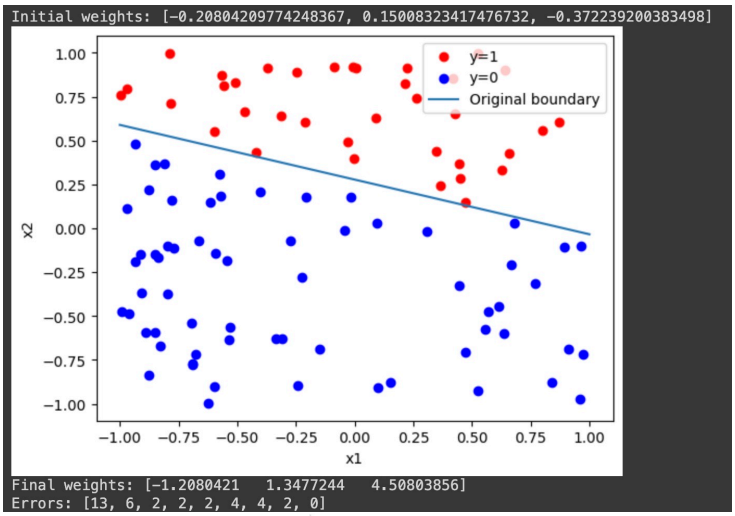
Neural Network - HW3

Gokhulnath Thirumaran
675086474

1) a) The code has been submitted Seperately.

weight Vector = [-0.20804, 0.150083, -0.37223]

b) Code has been submitted. The below is the output.



c) Given and verified that $\begin{bmatrix} \omega_1^* \\ \omega_2^* \end{bmatrix}$ is normal to the line $\omega^{*T} x$.

To prove: distance between origin and $\omega^{*T} x$

$$= \frac{|\omega_0^*|}{\sqrt{\omega_1^{*2} + \omega_2^{*2}}}.$$

Inferences: Shortest distance between a point and a line

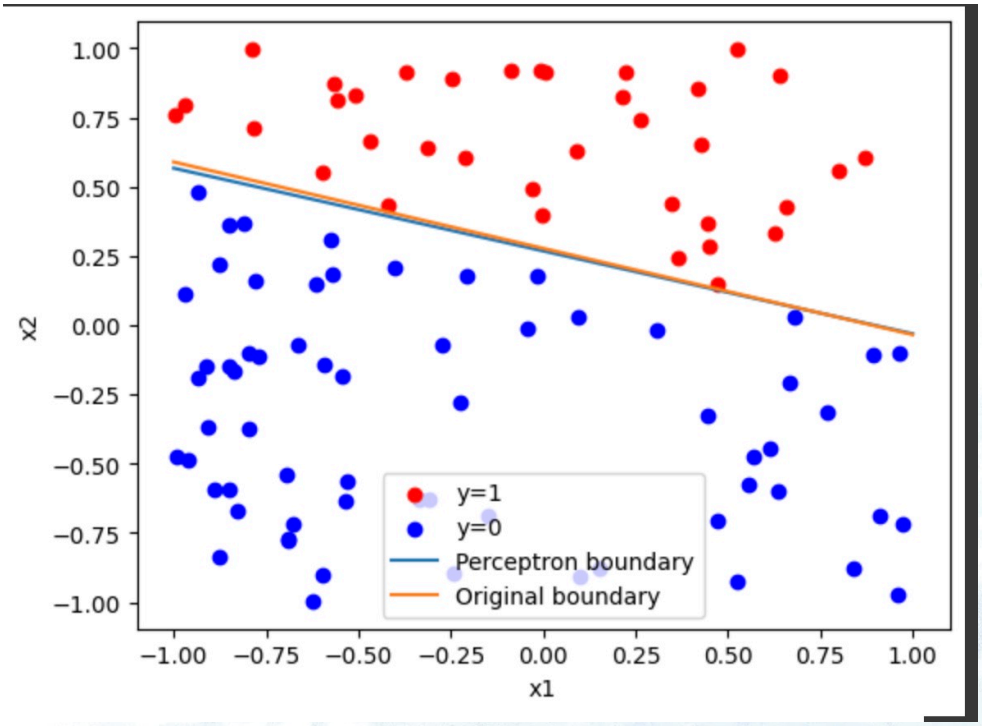
$$= \frac{|\omega_0 + \omega_1 x_1 + \omega_2 x_2|}{\sqrt{\omega_1^2 + \omega_2^2}}$$

Since $x_1 = 0$, $x_2 = 0$ for origin.

$$= \frac{|\omega_0^*|}{\sqrt{\omega_1^{*2} + \omega_2^{*2}}}$$

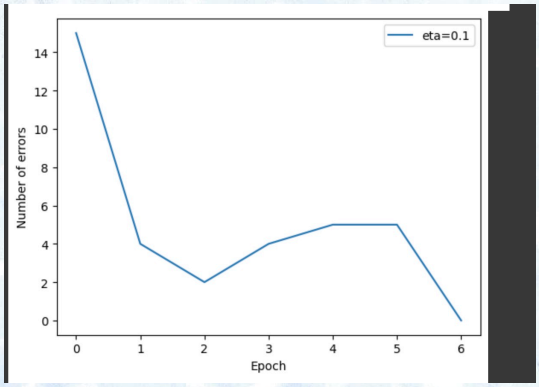
Thus proved.

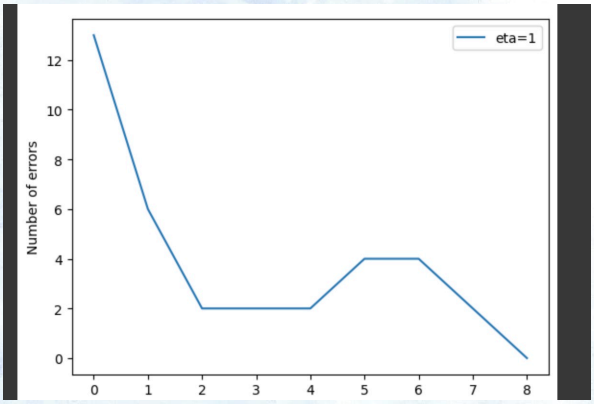
2) a)

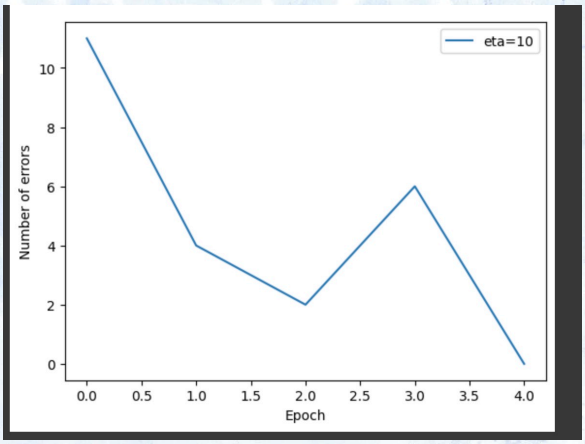


Due to randomness, the weights may not be the same.

b)



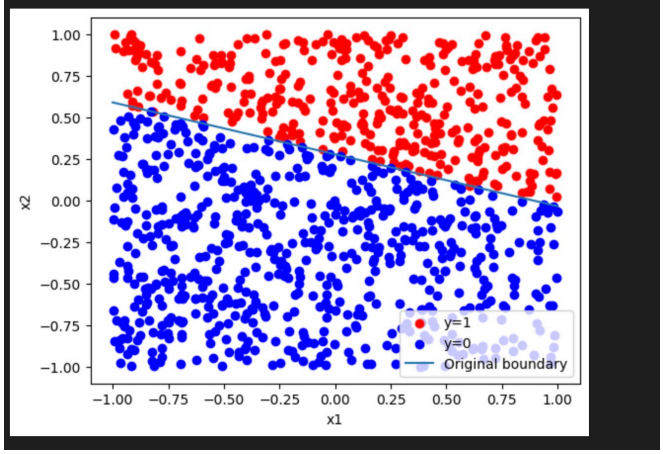




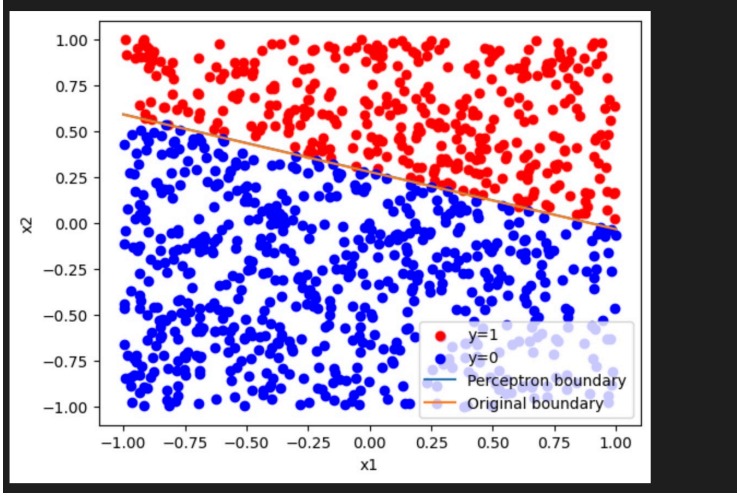
Smaller eta had more control over parameter than the larger as the steps are bigger.

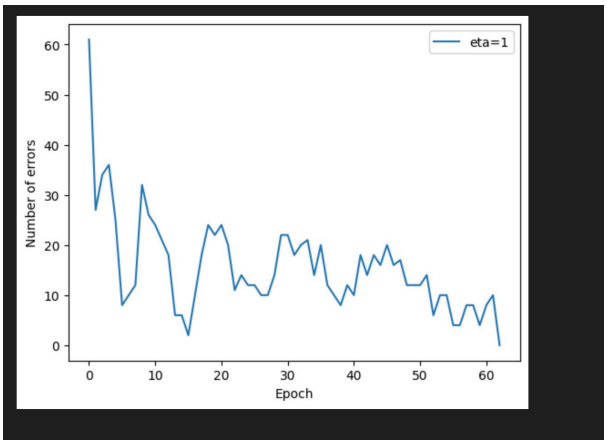
9

```
Initial weights: [-0.20804209774248367, 0.15008323417476732, -0.372239200383498]
```




```
Final weights: [-7.2080421   8.05960245 25.83207236]
Errors: [61, 27, 34, 36, 25, 8, 10, 12, 32, 26, 24, 21, 18, 6, 6, 2, 10, 18, 24, 22, 24, 20,
```

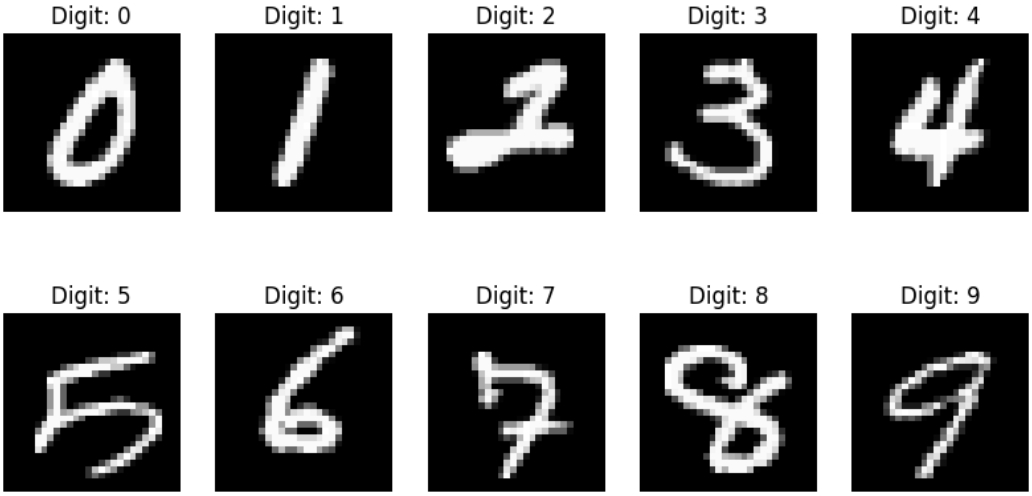




Gokhulnath Thirumaran
675086474

Neural Networks - Homework4

1.a)OUTPUT



1.c)

```
d = 10: MSE = 0.076562941097590O7, Mistakes = 38587
d = 50: MSE = 0.053354880047606 26, Mistakes = 15594
d = 100: MSE = 0.046409823616435 28, Mistakes = 12430
d = 200: MSE = 0.042253536353747 69, Mistakes = 11000
d = 500: MSE = 0.039377841635735815, Mistakes = 10397
```

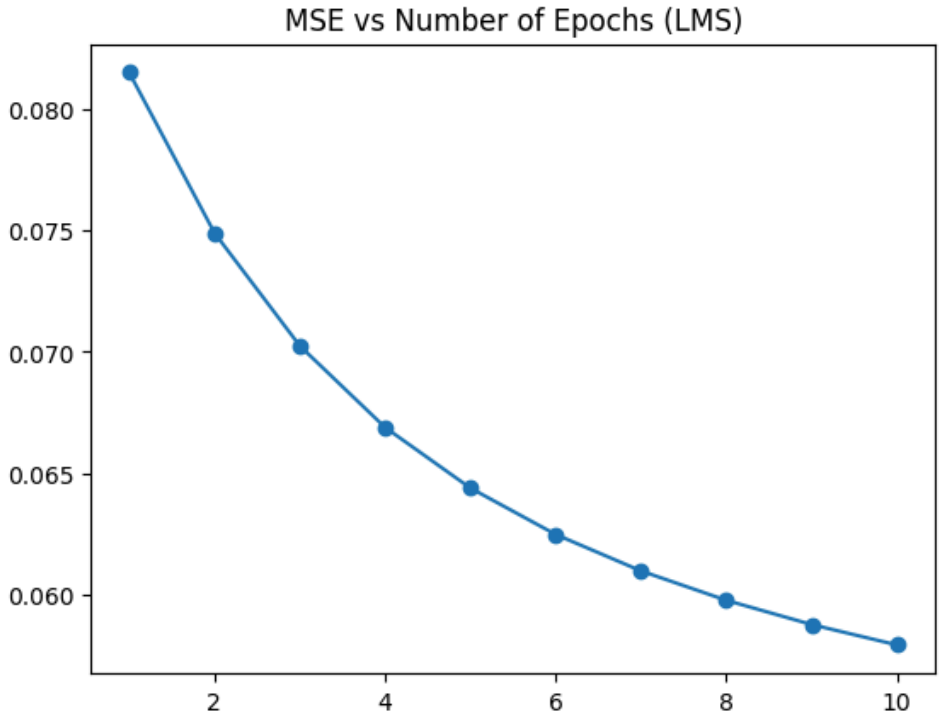
If we were to randomly guess a digit from 0 to 9 for each image in the dataset, the probability of guessing correctly is 1/10, while the probability of being wrong is 9/10.
Given the dataset's 70,000 images, we can estimate the expected number of mistakes from random guessing as follows:

Expected Mistakes = 9/10 * 70,000 = 63,000

Comparing the number of mistakes in the table to the expected number of mistakes, we can see that the model is performing significantly better than random guessing. For example, when d=10, the model makes 38587 mistakes, which is much lower than the expected number of mistakes.

Based on the table, it appears that a good choice for d is around 200-500. At these values, the model achieves a relatively low MSE and a relatively low number of mistakes.

1.d)



1) a) Gradient Descent

Given:

$$R(w) = 13w_1^2 - 10w_1w_2 + 4w_1 + 2w_2^2 - 2w_2 + 1$$

Partial Derivatives of R with respect to w_1 & w_2

$$\frac{\partial R}{\partial w_1} = 26w_1 - 10w_2 + 4 \quad - (1)$$

$$\frac{\partial R}{\partial w_2} = -10w_1 + 4w_2 - 2 \quad - (2)$$

To find the optimal solution, equate (1) & (2) to zero

$$26w_1 - 10w_2 + 4 = 0 \quad - (3)$$

$$-10w_1 + 4w_2 - 2 = 0 \quad - (4)$$

From (3),

$$26w_1 - 10w_2 + 4 = 0$$

$$26w_1 = 10w_2 - 4$$

$$w_1 = \frac{10w_2 - 4}{26}$$

$$w_1 = \frac{5w_2 - 2}{13} \rightarrow (5)$$

Substitute ⑤ in ④

$$-10\left(\frac{5w_2-2}{13}\right)+4w_2-2=0$$

$$\frac{-50w_2+20}{13}+4w_2-2=0$$

$$-50w_2+20+52w_2-26=0$$

$$-50w_2+52w_2+20-26=0$$

$$2w_2-6=0$$

$$w_2-3=0$$

$$\boxed{w_2=3} \rightarrow ⑥$$

Sub ⑥ in ⑤

$$w_1=\frac{5w_2-2}{13}$$

$$=\frac{5(3)-2}{13}$$

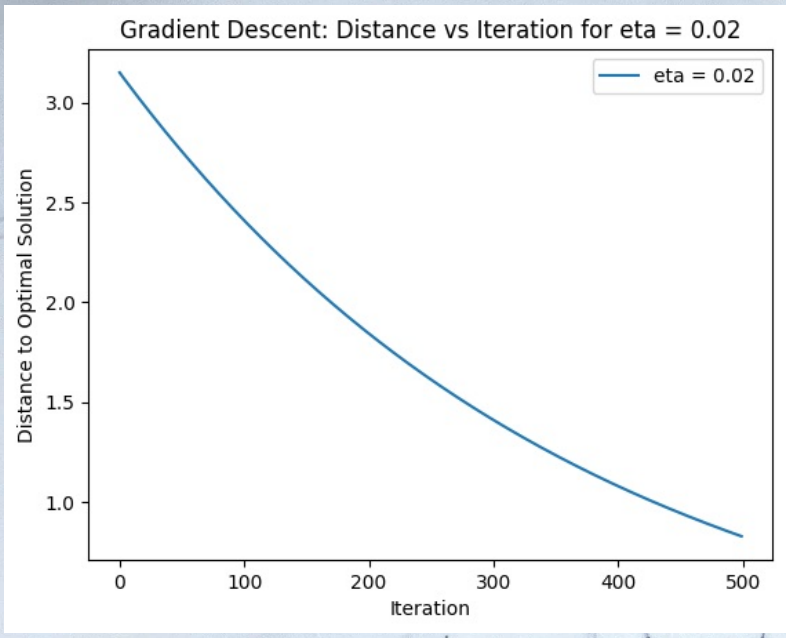
$$=\frac{15-2}{13}=\frac{13}{13}=1$$

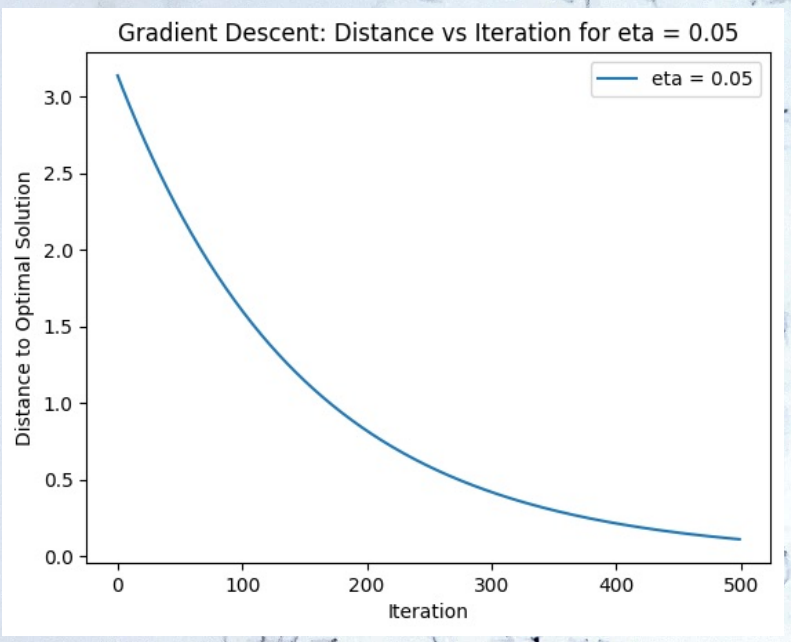
$$\boxed{w_1=1}$$

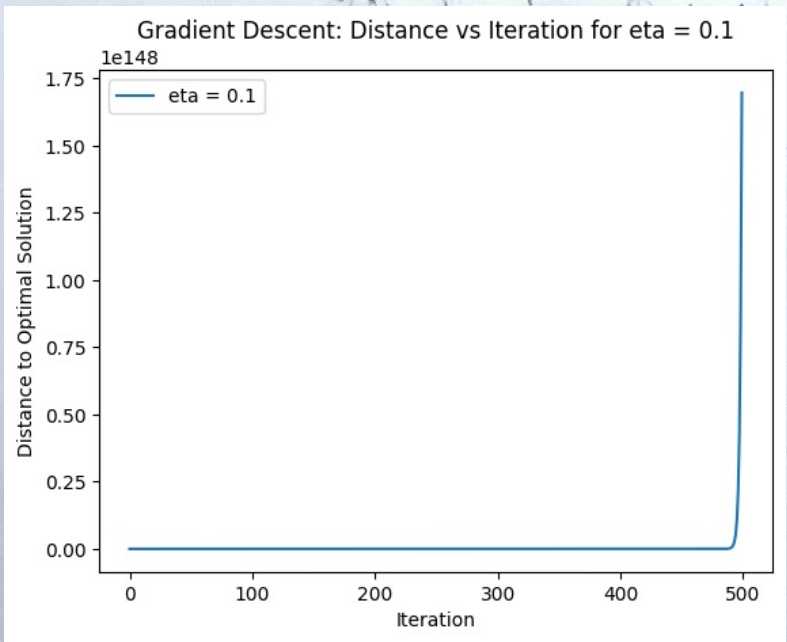
Thus, the optimal solution is $w_1=1$

$w_2=3$

b)


Gradient Descent: Distance vs Iteration for eta = 0.02
eta = 0.02
Distance to Optimal Solution
Iteration
3.0
2.5
2.0
1.5
1.0
0
100
200
300
400
500


Gradient Descent: Distance vs Iteration for eta = 0.05
eta = 0.05
Distance to Optimal Solution
Iteration
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
100
200
300
400
500


Gradient Descent: Distance vs Iteration for eta = 0.1
1e148
eta = 0.1
Distance to Optimal Solution
Iteration
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
100
200
300
400
500

c) If η is large, the algorithm may overshoot the optimal point, which might cause divergence.

If η is small, the convergence may be slow.

Thus choosing a optimal η is important.

2) a) Back propagation.

The Sigmoid function ϕ for x is

$$\phi(x) = \frac{1}{1+e^{-5x}}$$

Derivative of $\phi(x)$

$$\phi(x) = (1+e^{-ax})^{-1} \quad \text{where } a = +5$$

$$\phi'(x) = (-1)(1+e^{-ax})^{-2} \cdot (e^{-ax} \cdot -a)$$

$$= (1+e^{-ax})^{-2} (ae^{-ax})$$

$$= \frac{ae^{-ax}}{(1+e^{-ax})^2}$$

$$= a \cdot \frac{1}{(1+e^{-ax})} \cdot \frac{e^{-ax}}{(1+e^{-ax})}$$

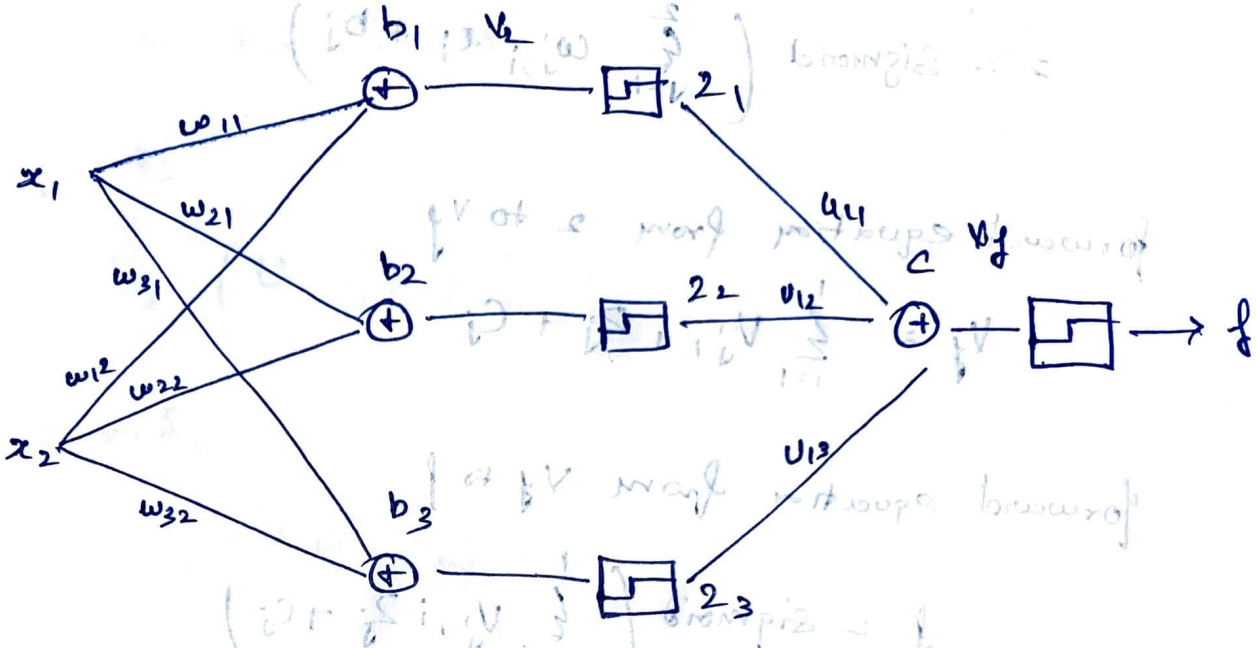
here $\frac{e^{-ax}}{1+e^{-ax}} = 1 - \phi(x)$

Thus

$$\phi'(x) = a \cdot \phi(x)(1-\phi(x))$$

$$\phi'(x) = 5 \cdot \phi(x)(1-\phi x))$$

2) b)



input vector $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_{2\times1}$ weight matrix $w = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix}_{3\times2}$

Bias vector $b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}_{3\times1}$ Hidden layer o/p vector $z = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix}_{3\times1}$

$V = [u_{11}, u_{12}, u_{13}]_{1\times3}$ $C = [C]_{1\times1}$ $f = [f]_{1\times1}$

weight matrix Bias output.

forward equation from x to V_2

$$V_2 = \sum_{i=1}^{2} w_{j,i} x_i + b_j$$

forward equation from V_2 to z

$$z = \text{Sigmoid}\left(\sum_{i=1}^{2} w_{j,i} x_i + b_j \right)$$

forward equation from z to V_f

$$V_f = \sum_{i=1}^{1} V_{j,i} z_j + c_j$$

forward equation from V_f to f

$$f = \text{Sigmoid}\left(\sum_{i=1}^{1} V_{j,i} z_j + c_j \right).$$

2) c) Given,

Squared loss function,

$$L(y, f) = (f - y)^2$$

also we know that

Sigmoid function

$$\phi(v) = \frac{1}{1 + e^{-v}}$$

derivation sigmoid fn

$$\phi'(v) = \phi(v)(1 - \phi(v))$$

$$\nabla_f l = \frac{\partial l}{\partial f} = 2(f - y)$$

$$\delta_f = \nabla_f l \cdot \phi(v_f) = 2(f - y)\,\phi'(v_f)$$

backward equation from

i) δ_f to δ_2

$$\delta_2 = (U \cdot \delta_f) \cdot \phi'(v_2)$$

ii) δ_2 to δ_x

$$\delta_x = (w^T \cdot \delta_2)$$

Gradients for

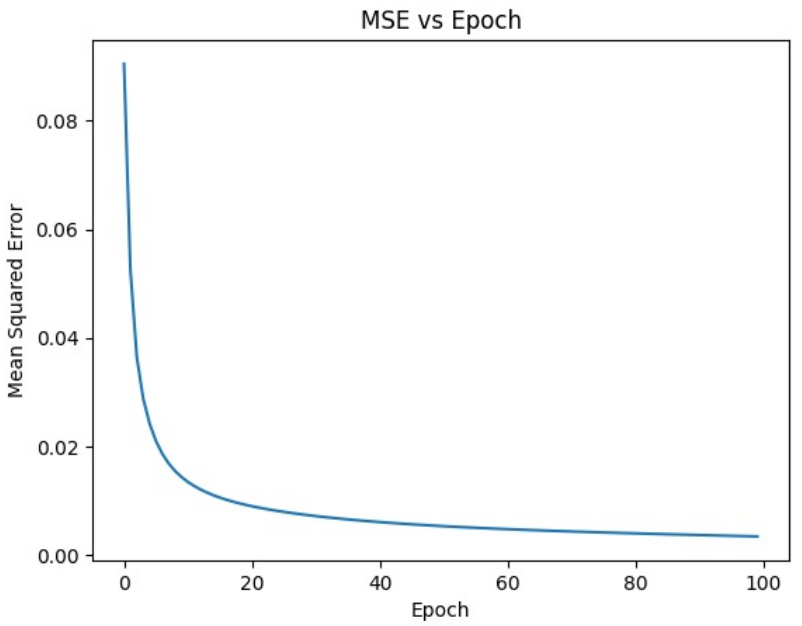
$$\nabla_U l = 2 \cdot \delta_f$$

$$\nabla_c l = \delta_f$$

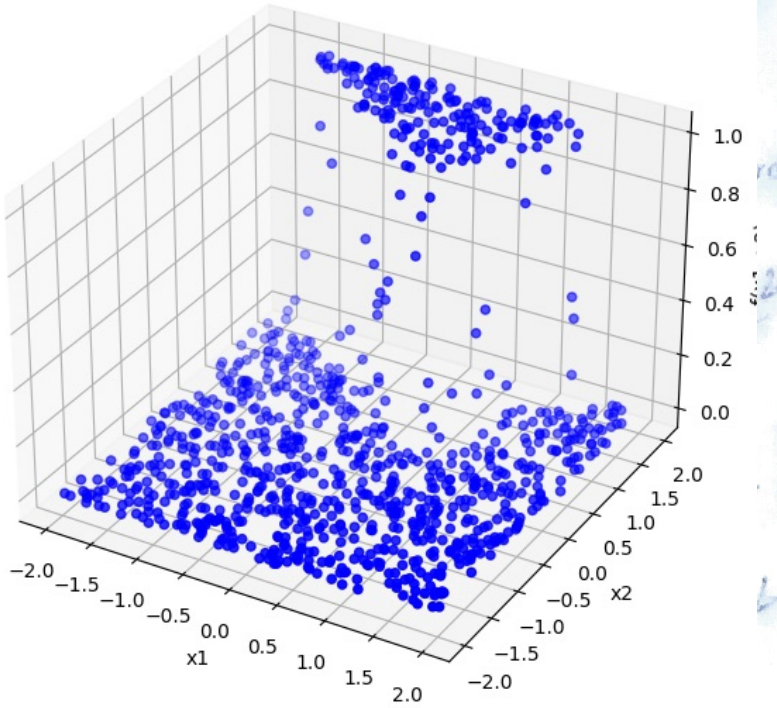
$$\nabla_w l = x \cdot \delta_2$$

$$\nabla_b l = \delta_2$$

d)


MSE vs Epoch
Mean Squared Error
Epoch
0.00
0.02
0.04
0.06
0.08
0
20
40
60
80
100

e)


x1
x2
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
0.0
0.2
0.4
0.6
0.8
1.0

d) we have choosen

Learning rate change

Mini batch gradient Descent

Start y high but multiply by 0.9 if MSE ↑

Results

Starting with a high learning rate can accelerate convergence early in the process. But as we approach a minimum, smaller learning rate stabilize it. Mini Batches can help reduce variance in the gradient, thus leading to smoother convergence.