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

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

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

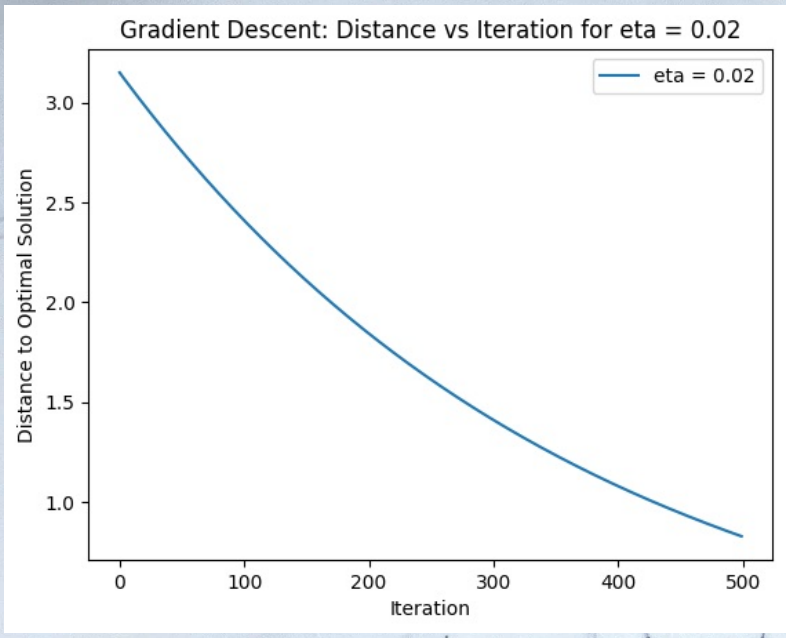
$$=\frac{15-2}{13}=\frac{13}{13}=1$$

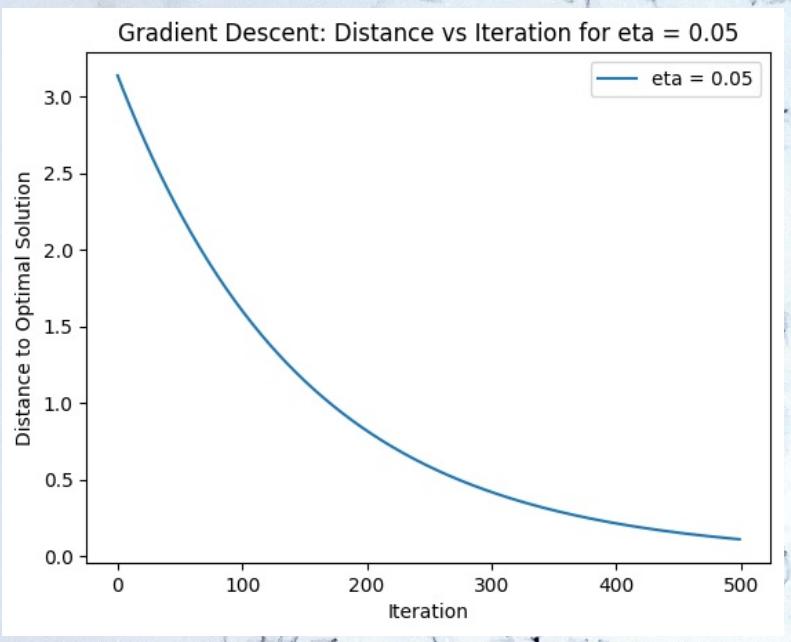
$$\boxed{w_1=1}$$

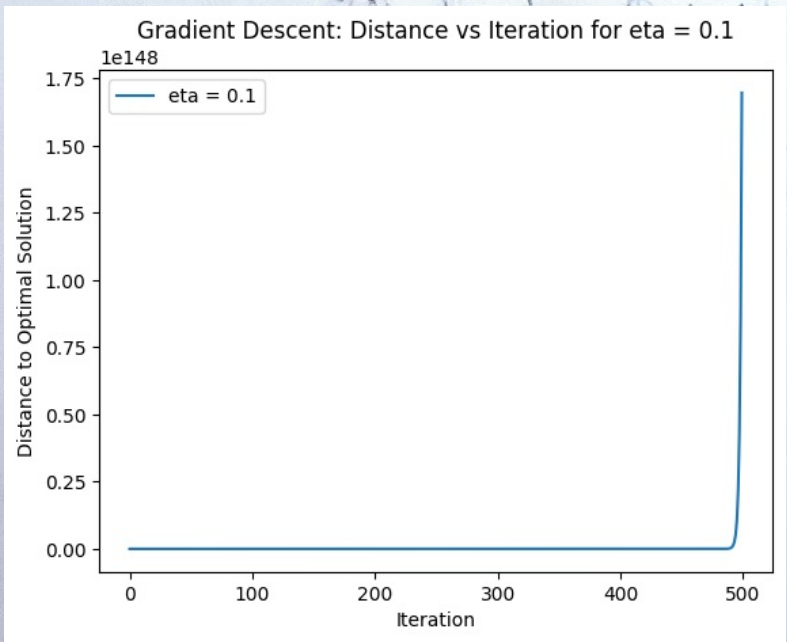
Thus, the optimal solution is $w_1=1$

$w_2=3$

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 $\phi(x)$

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

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

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

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

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

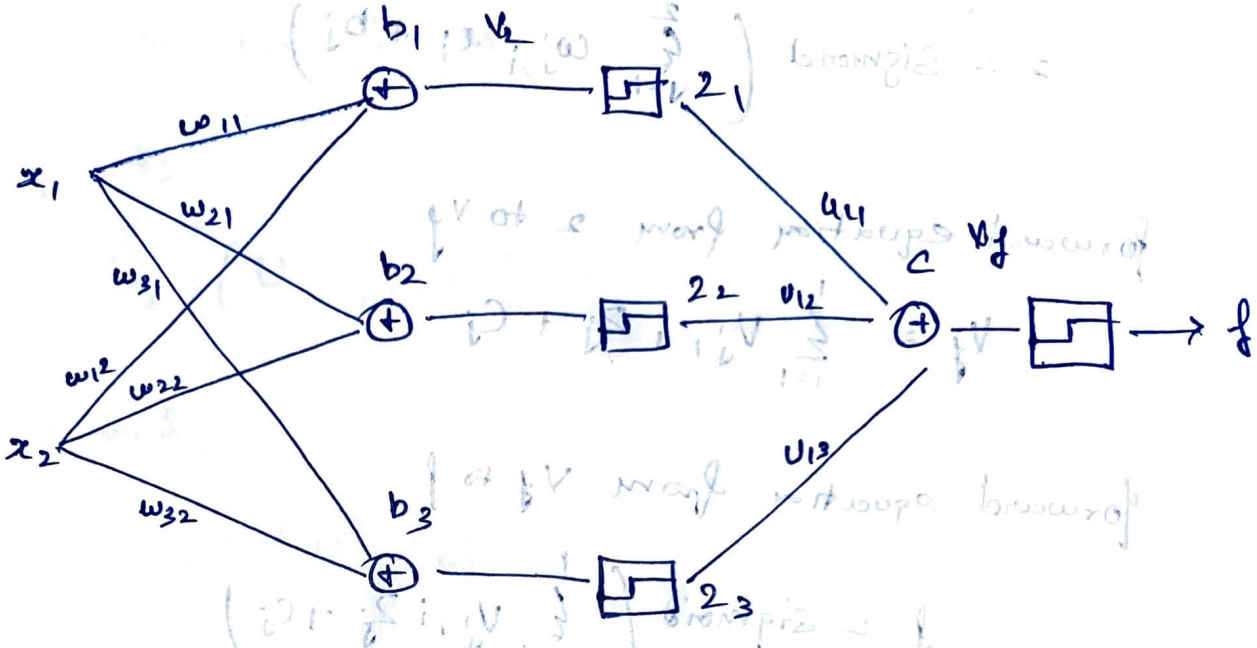
here $\frac{e^{-ax}}{1+e^{-ax}} = 1 - \phi(x)$

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

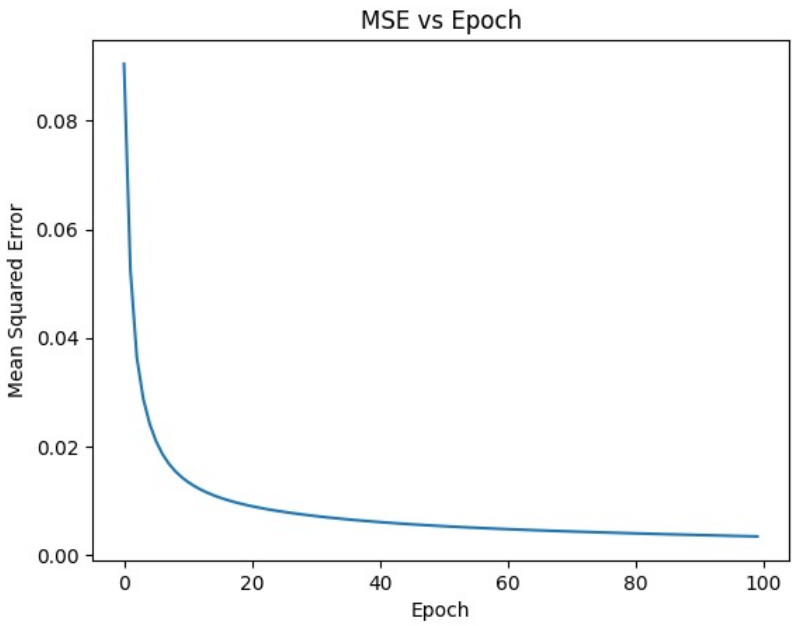
$$\nabla_U l = 2 \cdot \delta_f$$

$$\nabla_c l = \delta_f$$

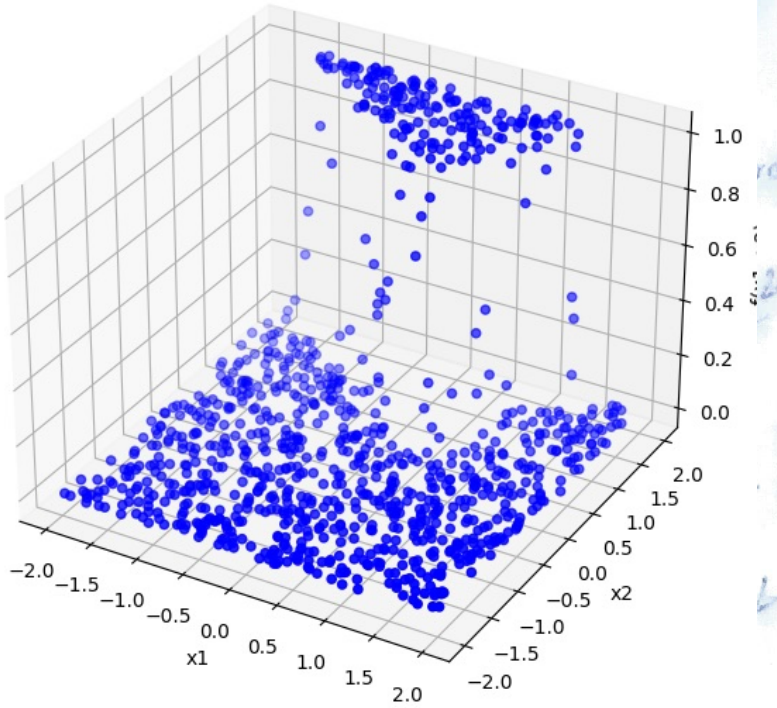
$$\nabla_w l = x \cdot \delta_2$$

$$\nabla_b l = \delta_2$$

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.