1. (20 MARKS) Consider a binary linear classification problem where the data points are two dimensional, i.e., $\mathbf{x} = (x_1, x_2) \in \mathbb{R}^2$ and the labels $y \in \{-1, 1\}$. Throughout this problem consider the data-set with following three points:

$$\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3)\}$$

where the input data-vectors are given by:

$$\mathbf{x}_1 = (1,0)^T, \quad \mathbf{x}_2 = (0,1)^T, \quad \mathbf{x}_3 = (-1,0)^T.$$

and the associated labels are given by

$$y_1 = +1, \quad y_2 = +1, \quad y_3 = -1.$$

Our aim is to find a linear classification rule: $w_0 + w_1 x_1 + w_2 x_2$ with weight vector $\mathbf{w} = (w_0, w_1, w_2)^T$ that classifies this dataset.

10 marks

(a) Suppose we implement the perceptron learning algorithm as discussed in the class with the initial weight vector $\mathbf{w} = (0,0,0)^T$ and the standard update rule for mis-classified points. Assume that each point that falls on the boundary is treated as a mis-classified point and the algorithm visits the points in the following order:

$$\mathbf{x}_1 \to \mathbf{x}_2 \to \mathbf{x}_3 \to \mathbf{x}_1 \to \mathbf{x}_2 \cdots$$

until it terminates. Show the output of the perceptron algorithm in each step and sketch the final decision boundary when the algorithm terminates. What is the distance between the decision boundary to the closest data vector in $\mathcal{D}$?
[**Important**: When applying the perceptron update, recall that you have to transform the data vectors to include the constant term i.e., $\mathbf{x}_1 = (1,0)^T$ must be transformed to $\tilde{\mathbf{x}}_1 = (1,1,0)^T$ etc.]

$w^{(0)} = [0 \;\; 0 \;\; 0]^T$

step 1 : $\hat{y}_1 = \text{Sign}(w^{(0)T} \tilde{x}_1) = \text{Sign}([0\;0\;0]\begin{bmatrix}1\\1\\0\end{bmatrix}) = 0$ misclassif.

$w^{(1)} \leftarrow w^{(0)} + y_1 \tilde{x}_1 \quad \Rightarrow w^{(1)} = \begin{bmatrix}0\\0\\0\end{bmatrix} + \begin{bmatrix}1\\1\\0\end{bmatrix} = \begin{bmatrix}1\\1\\0\end{bmatrix}$

Step 2 : $\hat{y}_2 = \text{Sign}(w^{(1)T} \tilde{x}_2) = \text{Sign}([1\;1\;0]\begin{bmatrix}0\\1\\1\end{bmatrix}) = +1$ Correctly Classified

Step 3 : $\hat{y}_3 = \text{Sign}(w^{(1)T} \tilde{x}_3) = \text{Sign}([1\;1\;0]\begin{bmatrix}-1\\0\\0\end{bmatrix}) = 0$ misclassified

$w^{(2)} \leftarrow w^{(1)} + y_3 \tilde{x}_3 \quad \Rightarrow w^{(2)} = \begin{bmatrix}1\\1\\0\end{bmatrix} + (-1)\begin{bmatrix}1\\-1\\0\end{bmatrix} = \begin{bmatrix}0\\2\\0\end{bmatrix}$

Step 4 : $\hat{y}_1 = \text{Sign}(w^{(2)T} \tilde{x}_1) = \text{Sign}([0\;2\;0]\begin{bmatrix}1\\1\\0\end{bmatrix}) = +1$ correctly classified.

Step 5: $\hat{y}_2 = \text{Sign}\left(w^{(2)T}\tilde{x}_2\right) = \text{Sign}\left([0\ 2\ 0]\begin{bmatrix}1\\0\\1\end{bmatrix}\right) = 0$ misclassified.

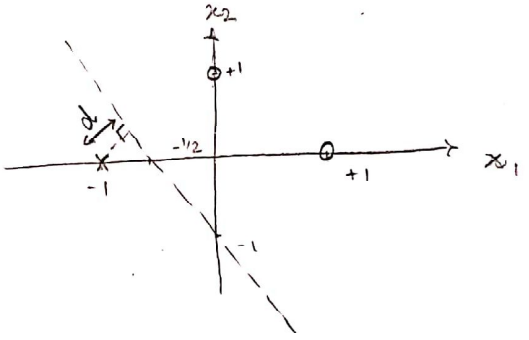
$$w^{(3)} \leftarrow w^{(2)} + y_2\tilde{x}_2 \Longrightarrow w^{(3)} = \begin{bmatrix}0\\2\\0\end{bmatrix} + \begin{bmatrix}1\\0\\1\end{bmatrix} = \begin{bmatrix}1\\2\\1\end{bmatrix}$$

Step 6: $\hat{y}_3 = \text{Sign}\left(w^{(3)T}\tilde{x}_3\right) = \text{Sign}\left([1\ 2\ 1]\begin{bmatrix}1\\-1\\0\end{bmatrix}\right) = -1$ Correctly classif.

Step 7: $\hat{y}_1 = \text{Sign}\left(w^{(3)T}\tilde{x}_1\right) = \text{Sign}\left([1\ 2\ 1]\begin{bmatrix}1\\0\\0\end{bmatrix}\right) = +1$ Correctly classif.

Step 8: $\hat{y}_2 = \text{Sign}\left(w^{(3)T}\tilde{x}_2\right) = \text{Sign}\left([1\ 2\ 1]\begin{bmatrix}1\\0\\1\end{bmatrix}\right) = +1$ Correctly classified.

$w^{(3)} = \begin{bmatrix}1\\2\\1\end{bmatrix}$ boundary: $1 + 2x_1 + x_2 = 0$



$$d = \frac{|1 + 2(-1) + 0|}{\sqrt{2^2 + 1^2}} = \frac{1}{\sqrt{5}}$$

10 marks

(b) Suppose we modify the perceptron algorithm as follows: At iteration t suppose that $\mathbf{w}^t = (w_0^t, w_1^t, w_2^t)$ is the present value of the weight vector and $(\mathbf{x}^t, y^t)$ is the training sample from $\mathcal{D}$ selected. Let us express $\mathbf{x}^t = (x_1^t, x_2^t)$. We perform the standard update to $\mathbf{w}^t$ if any of the following two conditions are satisfied:

- The training point $(\mathbf{x}^t, y^t)$ is mis-classified with respect to $\mathbf{w}^t$ (or lies on the decision boundary)
- The weight vector $\mathbf{w}^t$ and the training point $(\mathbf{x}^t, y^t)$ are such that we have:

$$\frac{y^t(w_0^t + w_1^t x_1^t + w_2^t x_2^t)}{\sqrt{(w_1^t)^2 + (w_2^t)^2}} \leq \frac{1}{2}.$$

Assume that we initialize $\mathbf{w}^0 = (0,0,0)$ and the algorithm visits the points in $\mathcal{D}$ in the following order:

$$\mathbf{x}_1 \to \mathbf{x}_2 \to \mathbf{x}_3 \to \mathbf{x}_1 \to \mathbf{x}_2 \cdots$$

until it terminates. Show the output of this algorithm in each step and sketch the final decision boundary when the algorithm terminates. What is the distance between the decision boundary to the closest data vector in $\mathcal{D}$?

$w^{(0)} = [0, 0, 0]^T$

step 1 : $\hat{y}_1 = \text{sign}(w^{(0)T} \tilde{x}_1) = \text{sign}([0,0,0]\begin{bmatrix}1\\1\\0\end{bmatrix}) = 0$ misclassified

$w^{(1)} \leftarrow w^{(0)} + y_1 \tilde{x}_1 \Rightarrow w^{(1)} = \begin{bmatrix}1\\1\\0\end{bmatrix}$

step 2 : $\hat{y}_2 = \text{sign}(w^{(1)T} \tilde{x}_2) = \text{sign}([1,1,0]\begin{bmatrix}1\\0\\1\end{bmatrix}) = +1$

$$\frac{y_2(w_0^1 + w_1^1 x_1^2 + w_2^1 x_2^2)}{\sqrt{(w_1^1)^2 + (w_2^1)^2}} = \frac{+1(1+0+0)}{\sqrt{1+0}} = 1 > 1/2$$

Step 3 : $\hat{y}_3 = \text{sign}(w^{(1)T} \tilde{x}_3) = \text{sign}([1,1,0]\begin{bmatrix}-1\\0\\1\end{bmatrix}) = 0$ misclassified

$w^{(2)} \leftarrow w^{(1)} + y_3 \tilde{x}_3 \rightarrow w^{(2)} = \begin{bmatrix}1\\1\\0\end{bmatrix} + (-1)\begin{bmatrix}-1\\0\\1\end{bmatrix} = \begin{bmatrix}0\\2\\0\end{bmatrix}$

total/10

[continue part (b) here]

Step 4: $\hat{y}_1 = \text{sign}(w^{(2)T}\tilde{x}_1) = \text{sign}([0,2,0]\begin{bmatrix}1\\1\\0\end{bmatrix}) = +1$ correctly classified

$$\frac{y_1(w_0^2 + w_1^2 x_1^1 + w_2^2 x_2^1)}{\sqrt{(w_1^2)^2 + (w_2^2)^2}} = \frac{1(0+2+0)}{\sqrt{2^2+0^2}} = 1 > 1/2$$

Step 5: $\hat{y}_2 = \text{sign}(w^{(2)T}\tilde{x}_2) = \text{sign}([0,2,0]\begin{bmatrix}1\\0\\1\end{bmatrix}) = 0$ misclassified

$$w^{(3)} = w^{(2)} + y_2\tilde{x}_2 = \begin{bmatrix}0\\2\\0\end{bmatrix} + (1)\begin{bmatrix}1\\0\\1\end{bmatrix} = \begin{bmatrix}1\\2\\1\end{bmatrix}$$

Step 6: $\hat{y}_3 = \text{sign}(w^{(3)T}\tilde{x}_3) = \text{sign}([1\ 2\ 1]\begin{bmatrix}-1\\0\\1\end{bmatrix}) = -1$ correct $\quad \frac{y_3(w_0^3 + w_1^3 x_1^3 + w_2^3 x_2^3)}{\sqrt{(w_1^3)^2 + (w_2^3)^2}} = \frac{1}{\sqrt{5}} < 1/2$

$$w^{(4)} = w^{(3)} + y_3\tilde{x}_3 = \begin{bmatrix}1\\2\\1\end{bmatrix} + (-1)\begin{bmatrix}-1\\0\\1\end{bmatrix} = \begin{bmatrix}0\\3\\1\end{bmatrix}$$

Step 7: $\hat{y}_1 = \text{sign}(w^{(4)T}\tilde{x}_1) = \text{sign}([0,3,1]\begin{bmatrix}1\\1\\0\end{bmatrix}) = +1$ ✓ $\quad \frac{y_1(w_0^4 + w_1^4 x_1^1 + w_2^4 x_2^1)}{\sqrt{(w_1^4)^2 + (w_2^4)^2}} = \frac{3}{\sqrt{10}} > 1/2$ ✓

Step 8: $\hat{y}_2 = \text{sign}(w^{(4)T}\tilde{x}_2) = \text{sign}([0,3,1]\begin{bmatrix}1\\0\\1\end{bmatrix}) = +1$ ✓ $\quad \frac{y_2(w_0^4 + w_1^4 x_1^2 + w_2^4 x_2^2)}{\sqrt{(w_1^4)^2 + (w_2^4)^2}} = \frac{1}{\sqrt{10}} < 1/2$

$$w^{(5)} = w^{(4)} + y_2\tilde{x}_2 = \begin{bmatrix}1\\3\\2\end{bmatrix}$$

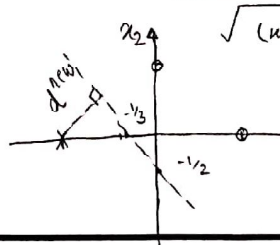
Step 9: $\hat{y}_3 = \text{sign}(w^{(5)T}\tilde{x}_3) = \text{sign}([1,3,2]\begin{bmatrix}-1\\0\\1\end{bmatrix}) = -1$ ✓ $\quad \frac{y_3(w_0^5 + w_1^5 x_1^3 + w_2^5 x_2^3)}{\sqrt{(w_1^5)^2 + (w_2^5)^2}} = \frac{2}{\sqrt{13}} > 1/2$ ✓

Step 10: $\hat{y}_1 = \text{sign}(w^{(5)T}\tilde{x}_1) = \text{sign}([1,3,2]\begin{bmatrix}1\\1\\0\end{bmatrix}) = +1$ ✓ $\quad \frac{y_1(w_0^5 + w_1^5 x_1^1 + w_2^5 x_2^1)}{\sqrt{(w_1^5)^2 + (w_2^5)^2}} = \frac{4}{\sqrt{13}} > 1/2$ ✓

Step 11: $\hat{y}_2 = \text{sign}(w^{(5)T}\tilde{x}_2) = \text{sign}([1\ 3\ 2]\begin{bmatrix}1\\0\\1\end{bmatrix}) = +1$ ✓ $\quad \frac{y_2(w_0^5 + w_1^5 x_1^2 + w_2^5 x_2^2)}{\sqrt{(w_1^5)^2 + (w_2^5)^2}} = \frac{3}{\sqrt{13}} > 1/2$ ✓

$w^{(5)} = \begin{bmatrix}1\\3\\2\end{bmatrix}$: boundary $1 + 3x_1 + 2x_2$

$$d^{new} = \frac{|1 + 3(-1) + 2(0)|}{\sqrt{3^2+2^2}} = \frac{2}{\sqrt{14}}$$



$d^{new} > d$ margin improved

□

2. (20 MARKS) Suppose we use a multi-class softmax regression model to classify input data vectors $\mathbf{x} \in \mathbb{R}^{d+1}$ (including bias) with two possible class labels $y \in \{1, 2\}$. Let $\mathbf{w}(1)$ and $\mathbf{w}(2)$ be the weight vectors for classes 1 and 2, respectively. For any input $\mathbf{x}$, we hypothesize that the probability of $\mathbf{x}$ belonging to class i is

$$\hat{P}^{\mathrm{SM}}(i|\mathbf{x}) \triangleq \frac{e^{\mathbf{w}(i)^T\mathbf{x}}}{e^{\mathbf{w}(1)^T\mathbf{x}} + e^{\mathbf{w}(2)^T\mathbf{x}}}, \quad \text{for } i \in \{1, 2\}.$$

Furthermore, for any given training example $(\mathbf{x}_n, y_n)$, we define the loss function as

$$e_n^{\mathrm{SM}}(\mathbf{w}(1), \mathbf{w}(2)) = -\log \hat{P}^{\mathrm{SM}}(y_n|\mathbf{x}_n) .$$

10 marks

(a) Find the gradients of $e_n^{\mathrm{SM}}(\mathbf{w}(1), \mathbf{w}(2))$ with respect to $\mathbf{w}(1)$ and $\mathbf{w}(2)$. (Note that you should always consider the two possible values of y_n.)

if $y_n = +1$: $e_n(w^{(1)}, w^{(2)}) = -\log \hat{P}^{SM}(y_n|x_n)$

$$\nabla_{w(1)} e_n(w^{(1)}, w^{(2)}) = \frac{-1}{\hat{P}^{SM}(y_n|x_n)} \cdot \nabla_{w(1)} \left[\frac{e^{w^{(1)T}x_n}}{e^{w^{(1)T}x_n} + e^{w^{(2)T}x_n}} \right] \quad (+1)$$

$$= -\frac{\left(e^{w^{(1)T}x_n} + e^{w^{(2)T}x_n}\right)}{e^{w^{(1)T}x_n}} \cdot \frac{x_n e^{w^{(1)T}x_n}\left(e^{w^{(1)T}x_n} + e^{w^{(2)T}x_n}\right) - x_n e^{w^{(1)T}x_n} \cdot e^{w^{(1)T}x_n}}{\left(e^{w^{(1)T}x_n} + e^{w^{(2)T}x_n}\right)^2} \quad (+1)$$

$$= \frac{-x_n e^{w^{(2)T}x_n}}{e^{w^{(1)T}x_n} + e^{w^{(2)T}x_n}} \quad (+1)$$

$$\Rightarrow \nabla_{w(1)} e_n(w(1), w(2)) = \frac{-x_n e^{w^{(2)T}x_n}}{e^{w^{(1)T}x_n} + e^{w^{(2)T}x_n}}$$

$$\nabla_{w(2)} e_n(w^{(1)}, w^{(2)}) = \frac{-1}{\hat{P}^{SM}(y_n|x_n)} \cdot \nabla_{w(2)} \left[\frac{e^{w^{(1)T}x_n}}{e^{w^{(1)T}x_n} + e^{w^{(2)T}x_n}} \right] = \quad (+1)$$

$$= -\frac{\left(e^{w^{(1)T}x_n} + e^{w^{(2)T}x_n}\right)}{e^{w^{(1)T}x_n}} \cdot \left[\frac{-x_n e^{w^{(2)T}x_n} \cdot e^{w^{(1)T}x_n}}{\left(e^{w^{(1)T}x_n} + e^{w^{(2)T}x_n}\right)^2} \right] \quad (+1) = \frac{+x_n e^{w^{(2)T}x_n}}{e^{w^{(1)T}x_n} + e^{w^{(2)T}x_n}} \quad (+1)$$

if $y_n = +2$: similarly (+2)

$$\nabla_{w(2)} e_n(w(1), w(2)) = \frac{-x_n e^{w(1)^T x_n}}{e^{w(1)^T x_n} + e^{w(2)^T x_n}} \qquad \nabla_{w(1)} e_n(w(1), w(2)) = \frac{+x_n e^{w(1)^T x_n}}{e^{w(1)^T x_n} + e^{w(2)^T x_n}} \quad (+2)$$

4 marks

(b) Suppose instead of the above softmax regression model, we use binary logistic regression to learn whether or not input $\mathbf{x}$ should be labelled class 1. We hypothesize that $\mathbf{x}$ belongs to class 1 with probability

$$\hat{P}^{\mathrm{LR}}(1|\mathbf{x}) \triangleq \frac{e^{\mathbf{w}^T\mathbf{x}}}{1+e^{\mathbf{w}^T\mathbf{x}}},$$

and $\mathbf{x}$ belongs to class 2 with probability $\hat{P}^{\mathrm{LR}}(2|\mathbf{x}) = 1-\hat{P}^{\mathrm{LR}}(1|\mathbf{x})$. For any given training example $(\mathbf{x}_n, y_n)$, we define the loss function as

$$e_n^{\mathrm{LR}}(\mathbf{w}) = -\log \hat{P}^{\mathrm{LR}}(y_n|\mathbf{x}_n).$$

Find a relationship between $(\mathbf{w}(1), \mathbf{w}(2))$ and $\mathbf{w}$, so that we have

$$\begin{cases} \hat{P}^{\mathrm{SM}}(1|\mathbf{x}) = \hat{P}^{\mathrm{LR}}(1|\mathbf{x}), \\ \hat{P}^{\mathrm{SM}}(2|\mathbf{x}) = 1-\hat{P}^{\mathrm{LR}}(1|\mathbf{x}). \end{cases}$$

$\hat{P}^{SM}(1|X) = \hat{P}^{LR}(1|X) \Rightarrow \dfrac{e^{w(1)^T x}}{e^{w(1)^T x} + e^{w(2)^T x}} = \dfrac{e^{(w(1)-w(2))^T x}}{1 + e^{(w(1)-w(2))^T x}} = \dfrac{e^{w^T x}}{1+e^{w^T x}}$ (+1) (+2) $\Rightarrow w = w(1) - w(2)$

$\hat{P}^{SM}(2|X) = \dfrac{e^{w(2)^T x}}{e^{w(2)^T x} + e^{w(1)^T x}} = 1 - \hat{P}^{SM}(1|X) = 1 - \hat{P}^{LR}(1|X)$ (+1)

if the first equality holds, the second one also holds.

6 marks

(c) Given $(\mathbf{w}(1), \mathbf{w}(2))$ and $\mathbf{w}$ as described Part (b), we apply SGD to separately train the above softmax regression model and binary logistic regression model, with constant learning rates ϵ^{SM} and ϵ^{LR}, respectively. For both models, all weights are initialized to zero, and we use the same random seed so that in each iteration of SGD the same random training example is selected. Find a relationship between ϵ^{SM} and ϵ^{LR}, so that $e_n^{\mathrm{SM}}(\mathbf{w}(1), \mathbf{w}(2))$ and $e_n^{\mathrm{LR}}(\mathbf{w})$ are identical in all iterations of SGD.

if $y_n = +1$

$$\nabla_w e_n^{LR}(w) = \nabla_w \left[-\log \hat{P}^{LR}(1|x_n)\right] = -\frac{1+e^{w^T x_n}}{e^{w^T x_n}} \cdot \nabla_w \left[\frac{e^{w^T x_n}}{1+e^{w^T x_n}}\right]$$

$$= -\frac{1+e^{w^T x_n}}{e^{w^T x_n}} \left[\frac{x_n e^{w^T x_n}(1+e^{w^T x_n}) - x_n e^{w^T x_n} \cdot e^{w^T x_n}}{(1+e^{w^T x_n})^2}\right] = -\frac{x_n}{1+e^{w^T x_n}}$$ (+1)

if $y_n = +2$:

$$\nabla_w e_n^{LR}(w) = \nabla_w \left[-\log \hat{P}^{LR}(2|x_n)\right] = -\frac{1+e^{w^T x_n}}{1} \nabla_w \left[\frac{1}{1+e^{w^T x_n}}\right] =$$

$$= -(1+e^{w^T x_n}) \cdot \left[-\frac{x_n e^{w^T x_n}}{(1+e^{w^T x_n})^2}\right] = \frac{x_n e^{w^T x_n}}{1+e^{w^T x_n}}$$ (+1)

if $y_n = +1$

$$w^{old} = w_{(1)}^{old} - w_{(2)}^{old}$$

$$w_{(1)}^{new} = w_{(1)}^{old} + \epsilon^{SM} \cdot \left(\frac{-x_n \, e^{w_{(2)}^{old^T} x_n}}{e^{w_{(1)}^{old^T} x_n} + e^{w_{(2)}^{old^T} x_n}} \right) = w_{(1)}^{old} - \epsilon^{SM} \frac{x_n}{e^{w^{old^T} x_n} + 1}$$

$$w_{(2)}^{new} = w_{(2)}^{old} + \epsilon^{SM} \left(\frac{+x_n \, e^{w_{(2)}^{old^T} x_n}}{e^{w_{(1)}^{old^T} x_n} + e^{w_{(2)}^{old^T} x_n}} \right) = w_{(2)}^{old} + \epsilon^{SM} \frac{x_n}{e^{w^{old^T} x_n} + 1}$$

$$w^{new} = w^{old} + \epsilon^{LR} \left(\frac{-x_n}{1 + e^{w_{old}^T x_n}} \right)$$ (+1) for updates

$$w^{new} = w_{(1)}^{new} - w_{(2)}^{new} \Rightarrow w^{old} - \epsilon^{LR} \frac{x_n}{1 + e^{w_{old}^T x_n}} = \underbrace{(w_{(1)}^{old} - w_{(2)}^{old})}_{w^{old}} - 2\epsilon^{SM} \frac{x_n}{1 + e^{w^{old^T} x_n}}$$

$$\Rightarrow \epsilon^{LR} = 2\epsilon^{SM}$$ (+1)

Alternatively, if $y_n = +2$:

$$w_{(1)}^{new} = w_{(1)}^{old} + \epsilon^{SM} \left(\frac{+x_n \, e^{w_{(1)}^{old^T} x_n}}{e^{w_{(1)}^{old^T} x_n} + e^{w_{(2)}^T x_n}} \right) = w_{(1)}^{old} + \epsilon^{SM} \frac{x_n \cdot e^{w^{old^T} x_n}}{e^{w^{old^T} x_n} + 1}$$

$$w_{(2)}^{new} = w_{(2)}^{old} + \epsilon^{SM} \left(\frac{-x_n \, e^{w_{(1)}^{old^T} x_n}}{e^{w_{(1)}^{old^T} x_n} + e^{w_{(2)}^{old^T} x_n}} \right) = w_{(2)}^{old} - \epsilon^{SM} \frac{x_n \cdot e^{w^{old^T} x_n}}{e^{w^{old^T} x_n} + 1}$$ (+1) for updates

$$w^{new} = w^{old} + \epsilon^{LR} \left(\frac{x_n \, e^{w^{old^T} x_n}}{1 + e^{w^{old^T} x_n}} \right)$$

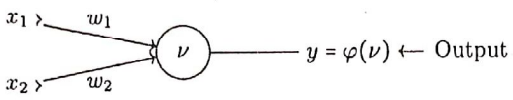
$$w^{new} = w_{(1)}^{new} - w_{(2)}^{new} \Rightarrow w^{old} + \epsilon^{LR} \frac{x_n \cdot e^{w^{old^T} x_n}}{1 + e^{w^{old^T} x_n}} = (w_{(1)}^{old} - w_{(2)}^{old}) + 2\epsilon^{SM} \frac{x_n e^{w^{old^T} x_n}}{1 + e^{w_{old}^T x_n}}$$

$$\Rightarrow \epsilon^{LR} = 2\epsilon^{SM}$$ (+1)

3. (10 MARKS) Assume two logical inputs (they can be either 0 or 1) as the following:

x_1	0	1	0	1
x_2	0	0	1	1

They are input to our single-layer model shown below:



where our weights and activation function are defined as following:

$$weights = \begin{cases} w_1 = 1 \\ w_2 = 1 \end{cases} \qquad \varphi(\nu) = \begin{cases} 1 & \text{if } \nu \geq 2 \\ 0 & \text{otherwise} \end{cases}$$

4 marks

(a) Given the 4 different set of inputs that x_1 and x_2 can have, calculate the output of the unit and mention what function can be represented by this unit?

x_1	x_2	$\nu = w_1x_1 + w_2x_2$	$y = \varphi(\nu)$
0	0	0	0
1	0	1	0
0	1	1	0
1	1	2	1

Output is 1 only when both inputs are active. This is logical AND operator.

3 marks

(b) Suggest on how to change the threshold levels (ν) of the activation function to implement the following function (we will use the same weights as before):

x_1	0	1	0	1
x_2	0	0	1	1
$g(x_1, x_2)$	0	1	1	1

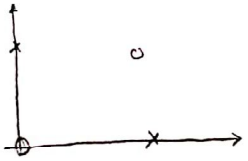
$g(x_1, x_2)$ represents the logical OR operator, since it outputs 1 if either of the inputs are active.

$$\varphi(\nu) = \begin{cases} 1 & \text{if } \nu \geq 1 \\ 0 & 0.w \end{cases}$$

3 marks

(c) Can the following function shown below be implemented by a *single* unit (one set of inputs and an activation function)? Explain why?

x_1	0	1	0	1
x_2	0	0	1	1
$z(x_1, x_2)$	0	1	1	0



Case 1: Linear classifier: No, there is no single linear classifier that can correctly classifies all data points.

Case 2: If we are not limited to linear classifiers one can implement $z(x_1, x_2)$ by the following:

$$z(x_1, x_2) = \begin{cases} 1 & \text{if } \frac{1}{2} < x_1 + x_2 < 3/2 \\ 0 & 0.w \end{cases}$$

total/10