1. Show that the single-Perceptron learning rule with hard limit activation converges (for linearly-separable problems). Is this an example of Supervised or Unsupervised Learning?

Let

$$x = \begin{bmatrix} w \\ b \end{bmatrix}$$

and

$$z_q = \begin{bmatrix} p_q \\ 1 \end{bmatrix}$$

Then we have that $w^T p + b = x^T z$. The learning rule can be represented as $\Delta x = ez$. Since x is only effected when e is 1 or -1, we can ignore the cases when $e = 0$. Following this observation, the learning rule becomes

$$x(k) = x(k-1) + z'(k-1)$$

where $z'(k-1) \in \{ -z(k-1), z(k-1)\}$

Assume that there exists a weight vector that correctly categorizes all input vectors. Then we have a solution vector x^* such that

$$(x^*)^T z_q > \delta > 0 \text{ if } t_q = 1 \quad \forall q \in \mathbb{N}$$

and

$$(x^*)^T z_q < -\delta < 0 \text{ if } t_q = 0 \quad \forall q \in \mathbb{N}.$$

We would like to find an upper and lower bound for the weight vector at each step k in the algorithm. Without loss of generality, assume that the weight vector is initialized as the zero-vector. Then we have

$$x(k) = \sum_{i=0}^{k-1} z'(i)$$

$$\implies (x^*)^T x(k) = (x^*)^T \sum_{i=0}^{k-1} z'(i)$$

Since $(x^*)^T z'(i) > \delta$ we know that

$$(x^*)^T x(k) > k\delta \tag{1}$$

Using the Cauchy-Schwartz inequality, we have

$$\left((x^*)^T x(k)\right)^2 \leq ||x^*||^2 \cdot ||x(k)||^2. \tag{2}$$

Combining (1) an (2), we get

$$||x(k)||^2 > \frac{(k\delta)^2}{||x^*||^2} \tag{3}$$

So we have a lower bound for the weight vector.

Next, we would like to find an upper bound.

$$\begin{aligned}||x(k)||^2 &= x^T(k)x(k) \\ &= [x(k-1) + z'(k-1)]^T [x(k-1) + z'(k-1)] \\ &= x^T(k-1)x(k-1) + 2x^T(k-1)z'(k-1) + z'^T(k-1)z'(k-1) \qquad (4)\end{aligned}$$

Since the weights are not updated unless the previous input vector has misclassified the inputs, we know that $x^T(k-1)z'(k-1) \leq 0$. So we can rewrite the (4) as

$$\begin{aligned}||x(k)||^2 &\leq x^T(k-1)x(k-1) + z'^T(k-1)z'(k-1) \\ &= ||x(k-1)||^2 + ||z'(k-1)||^2\end{aligned}$$

repeating this process, we get

$$||x(k)||^2 \leq \sum_{i=0}^{k-1} ||z'(i)||^2$$

Let $M = \max\{||z'(i)||\}, \;\; i = 0 : (k-1)$. Then

$$||x(k)||^2 \leq kM \qquad (5)$$

So we have an upper bound for the weight vector at any step.

Putting together the (3) and (5), we have

$$\begin{aligned}\frac{(k\delta)^2}{||x^*||^2} &< ||x(k)||^2 \leq kM \\ \implies k &< \frac{M||x^*||^2}{\delta^2}.\end{aligned}$$

Since k has an upper bound, the weights will converge in a finite number of steps. Therefore, the perceptron learning rule will converge in a finite number of steps.

2. Show x XOR y = (x OR y) AND [NOT(x AND y)] (truth tables are good enough).

x	y	x OR y	NOT (x AND y)	(x OR y) AND [NOT (x AND y)]	x XOR y
1	1	1	0	0	0
1	0	1	1	1	1
0	1	1	1	1	1
0	0	0	1	0	0

3. Why are the outputs of the OR and the NOT-AND Perceptrons guaranteed to be linearly separable? Does this hold in higher dimensions? Explain.

Because there always exists a straight line sepeartes the two output classes.

This hold in higher dimensions.

Prove above conclusion in a formal mathematical way:

Let $f(x_1, x_2, ..., x_n)$ be OR Perceptron and let $g(x_1, x_2, ..., x_n)$ be NOT-AND Perceptron. Let $X = (x_1, x_2, ..., x_n)$, $x_i \in \{0, 1\}$be the input vector. Then, we should have output as following:

$$f(X) = \begin{cases} 0, & x_1 = x_2 = ... = x_n = 0 \\ 1, & otherwise \end{cases}$$

$$g(X) = \begin{cases} 0, & x_1 = x_2 = ... = x_n = 1 \\ 1, & otherwise \end{cases}$$

To prove that they both hold the learly separable feature is equivalent to prove that we can always build a hyperplane.

For $f(X)$, we set up a hyperplane as follow:

$$\mathcal{L}(X) = \mathbf{1} \cdot X - \frac{1}{2} = \mathbf{0}$$

In this case:

(1) If $X = \mathbf{0}$, $\mathcal{L}(X) = \mathbf{0} - \frac{1}{2} = -\frac{1}{2} < 0$.
(2) If $X \neq \mathbf{0}$, that is, $\exists i \in 1:n$, s.t. $X_i = 1$, then $\mathcal{L}(X) \geq X_i - \frac{1}{2} = 1 - \frac{1}{2} = \frac{1}{2} > 0$
(3) Appy the hard-limit activation here, would give us
   (a) $hardlim(\mathbf{1} \cdot X - b) = \mathbf{0}$, when $X = \mathbf{0}$.
   (b) $hardlim(\mathbf{1} \cdot X - b) = \mathbf{1}$, otherwise.

That is, this separation hyperplane gives the same result as the OR Perceptron, and thus proved that the OR Perceptron is guaranteed to be linearly separable.

The same thought hold for NOT-AND Perceptron, while we use the following separation hyperplane:

$$\tilde{\mathcal{L}}(X) = -\mathbf{1} \cdot X + (n - \frac{1}{2}) = \mathbf{0},$$

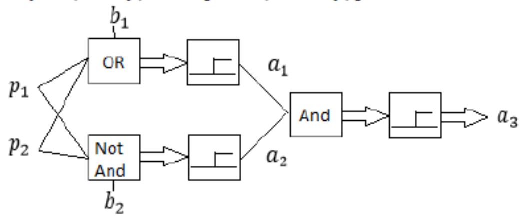
In this case:

(1) If $X = \mathbf{1}$, $\tilde{\mathcal{L}}(X) = -n + (n - \frac{1}{2}) = -\frac{1}{2} < 0$.
(2) If $X \neq \mathbf{1}$, that is, $\exists i \in 1:n$, s.t. $X_i = 0$, then $\tilde{\mathcal{L}}(X) = -\sum_{j \neq i}^{n} X_j - X_i + n - \frac{1}{2} \geq -(n-1) - 0 + n - \frac{1}{2} = 1 - \frac{1}{2} = \frac{1}{2} > 0$
(3) Appy the hard-limit activation here, would give us
   (a) $hardlim(\mathbf{1} \cdot X - b) = hardlim(-\frac{1}{2}) = \mathbf{0}$, when $X = \mathbf{1}$.
   (b) $hardlim(\mathbf{1} \cdot X - b) = \mathbf{1}$, otherwise.

In this way we showed that both OR and the NOT-AND Perceptrons guaranteed to be linearly separable, which hold for any dimenstions equal or larger than 2.

4. Can you think of a Perceptron *architecture* that would solve the XOR problem using an MLP with Hard-Limit activations? Discuss.

The Multi-Layer Perceptron solution to XOR is to include a previously-trained OR single Perceptron (a hidden node) and a previously-trained NOT-AND single Perceptron (another hidden node). And then combined them with a previously-trained single AND Perceptron (the output node). The combination should allow the MLP to implement the XOR function as follows: x XOR y = (x OR y) AND [NOT (x AND y)].



5. Can you think of a Perceptron *learning rule* that would converge in an *automated* fashion for the XOR problem using an MLP with Hard-Limit activations? Discuss.

You cannot solve the XOR problem in an automated fashion using MLP with Hard-Limit activations. As discussed in the lecture notes, the SATALINE can solve the XOR problem in an automated fashion using MLP with Hard-Limit activations.

6. Derive a MADALINE learning rule for a simple MADALINE with two inputs, two linear-activation nodes in the hidden layer and one linear-activation output node by back-propagating the error from the output node to the input weights using the chain rule of differentiation. The goal is to minimize the squared error: $e = ½(t - y)^2$ using gradient descent.

Notation: inputs: $\boldsymbol{p}$, first layer matrix is $\boldsymbol{W}$, hidden activation is $\boldsymbol{a}$, output layer matrix is U, output is y, objective function is $e = \frac{1}{2}(t-y)^2$.

$$\begin{aligned}
\boldsymbol{W} &= \begin{bmatrix} W_{11} & W_{12} & b_1 \\ W_{21} & W_{22} & b_2 \end{bmatrix} \\
\boldsymbol{p} &= \begin{pmatrix} p_1 \\ p_2 \\ 1 \end{pmatrix} \\
\boldsymbol{a} &= \boldsymbol{W}\boldsymbol{p} \\
\boldsymbol{a} &= \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} \\
Ux &= y \\
U &= \begin{bmatrix} U_1 & U_2 & c_1 \end{bmatrix} \\
u &= \begin{bmatrix} U_1 & U_2 \end{bmatrix} \\
x &= \begin{bmatrix} a_1 \\ a_2 \\ 1 \end{bmatrix}
\end{aligned}$$

Apply Delta learning rule, in k th iteration

$$\begin{aligned}
\Delta U_k &= -\eta \nabla_U (e_k) \\
&= -\eta \,(t-y_k)\, \boldsymbol{x}_k^T \\
\Delta W_k &= -\xi \nabla_{\boldsymbol{a}_k}(e_k)\, \nabla_{\boldsymbol{W}_k}(\boldsymbol{a}_k) \\
&= -\xi \,(t-y_k)\, \boldsymbol{u}_k^T \boldsymbol{p}_k^T
\end{aligned}$$

7. Show that MADALINEs with one Hidden Layer cannot solve Linearly-Separable two-class problems by demonstrating that the separation hyperplane is given by

$$\widetilde{\mathbf{W}}\mathbf{p} + \tilde{\mathbf{b}} = \mathbf{0},$$

where $\widetilde{\mathbf{W}}$ and $\tilde{\mathbf{b}}$ are functions of the hidden-layer and output-layer weights and biases. Show that this result holds for an arbitrary number of hidden layers.

Let W and b be, respectively, the weights and the biases for the hidden layer and $\widehat{W}$ and $\hat{b}$ be, respectively , the weights and the biases for the output layer. The hidden layer produces $Wp + b = a$. The separation hyperplane is given by

$$\widehat{W}a + \hat{b} = 0$$
$$\widehat{W}(Wp + b) + \hat{b} = 0$$
$$\widehat{W}Wp + \widehat{W}b + \hat{b} = 0$$

Let $\widehat{W}W = \widetilde{W}$ and $\widehat{W}b + \hat{b} = \tilde{b}$. So we have $\widetilde{W}p + \tilde{b} = 0$ and clearly, a single line (i.e. one hidden layer MADLINE) cannot separate the two non-linearly separable classes.
For an arbitrary number (n) of hidden layers, we have $W_1, W_2, \dots W_n$ as the weight for the hidden layers and $b_1, b_2, \dots, b_n$ as the biases for the hidden layers. The separation hyperplane is given by

$$\left(\widehat{W}W_1W_2 \dots W_n\right)p + \left(\widehat{W}b_1 + \widehat{W}b_2 + \cdots + \widehat{W}b_n + \hat{b}\right) = 0 = \widetilde{W}p + \tilde{b}$$

Similar to the one hidden layer case, the multiple hidden layers case also cannot solve two non-linearly separable classes.

8. What are the two criteria required for ANNs to be universal mapping approximators? What condition must be met for automated learning on a multi-layer architecture?

automated learning on a multi-layer architecture?
The two criteria required for ANNs to be universal mapping approximates are hidden layers (as MLP) and non-linear activation. Automated learning on a multi-layer architecture requires smooth (differentiable) activation.