1) Consider the following linearly separable training set:

$$\left\{\mathbf{x}^1 = \begin{pmatrix}0\\0\end{pmatrix}, \mathbf{x}^2 = \begin{pmatrix}0\\2\end{pmatrix}, \mathbf{x}^3 = \begin{pmatrix}1\\1\end{pmatrix}, \mathbf{x}^4 = \begin{pmatrix}1\\-1\end{pmatrix}\right\}$$

$$\{t_1 = -1, t_2 = +1, t_3 = +1, t_4 = -1\}$$

a) Initialize all weights to one (including the bias). Use a learning rate of one for simplicity. Apply the perceptron learning algorithm until convergence.

Most default, common, normal, NPC-type exercise.

According to the question, we start with $\eta = 1$ and $\mathbf{w} = (1\,1\,1)^T$.

Now, we take the first data point $\mathbf{x}^1$ and augment it with a bias dedicated dimension:

$$(x_0^1\, x_1^1\, x_2^1)^T \rightarrow (1\,0\,0)^T$$

Add bias to all data points.

and compute the perceptron output for it:

target was (-1)

$$o^1 = sign\,(\mathbf{w} \cdot \mathbf{x}) = sign\,(1 \cdot 1 + 1 \cdot 0 + 1 \cdot 0) = +1$$

Apply perceptron formula.

Since the perceptron made a mistake, we use the output to compute an update:

$$\mathbf{w} = (1\,1\,1)^T + 1\,(-1 - 1)\,(1\,0\,0)^T = (-1\,1\,1)^T$$

$w + \eta(t - o^1)\, x^1 \quad w_{new}$

Use this weight vector from now on. (Until next mistake.)

$o^2 = sign(-1 \times 1 + 0 \times 1 + 1 \times 2) = +1$ ✓

$o^3 = sign(-1 + 1 + 1) = +1$ ✓

$o^4 = sign(-1 + 1 - 1) = -1$ ✓

No mistake was done again, which concludes our first sweep through the data (i.e. the first epoch). If we do another epoch, we see that the weights don't change. Thus, we have convergence.

Solution:

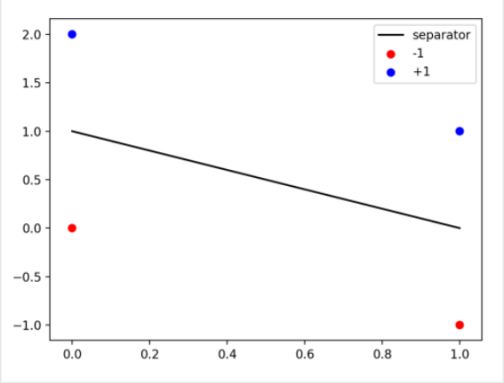
Since we are working in two dimensions the weights define a separation line between the two classes. We get the equation for this line by checking the critical point where the decision changes from +1 to -1:

$$w_0 x_0 + w_1 x_1 + w_2 x_2 = 0$$

Useful for visualization.

$$w_0 x_0 + w_1 x_1 + w_2 x_2 = 0$$
$$(-1)1 + x_1 + x_2 = 0$$
$$x_2 = -x_1 + 1$$

Having the line's equation, we can draw it in a plot with our data points to see the separation:



(From another exercise)

c) What is the perceptron output for the query point $(0\ 0\ 1)^T$?

Solution:
We start by augmenting the query with a bias dimension:

$$(x_0\ x_1\ x_2\ x_3)^T \rightarrow (1\ 0\ 0\ 1)^T$$

After reaching convergence, we might be asked the outputs of specific query points.

We take the converged weights:

$$\mathbf{w} = (-1\ 1\ 1\ 1)^T$$

And compute the perceptron output for the point:

$$o = sign\,(\mathbf{w} \cdot \mathbf{x}) = sign\,(-1 \cdot 1 + 1 \cdot 0 + 1 \cdot 0 + 1 \cdot 1) = +1$$

We see that the point is on the boundary. So, because of the way we defined the sign function, the perceptron outputs $+1$, thus recognizing the point as a member of the learned class.

3) What happens if we replace the sign function by the step function?

$$\Theta\,(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}$$

Specifically, how would you change the learning rate to get the same results?

Great question!

Solution:
Recall the lerning rule in question:

$$\mathbf{w} = \mathbf{w} + \eta\,(t - o)\,\mathbf{x}$$

The only term that depends on the perceptron output is the error $(y - o)$. With the sign function, the error is:

$$\delta_{sign} = t - o_{sign} = t - sign(\mathbf{w} \cdot \mathbf{x})$$

Which can be unfolded as:

$$\delta_{sign} = \begin{cases} +2 & t = +1, \mathbf{w} \cdot \mathbf{x} < 0 \\ -2 & t = -1, \mathbf{w} \cdot \mathbf{x} \geq 0 \end{cases}$$

Remember, we only need to update the weights when the perceptron makes a mistake.

↳ What about 0, $t = sign(w \cdot x)$?

Thus, we don't need to consider this case

Moving on:

Whereas with the step function, the error is:

$$\delta_{step} = t - o_{step} = t - \Theta(\mathbf{w} \cdot \mathbf{x})$$

Which can be unfolded as:

$$\delta_{step} = \begin{cases} +1 & t = +1, \mathbf{w} \cdot \mathbf{x} < 0 \\ -1 & t = -1, \mathbf{w} \cdot \mathbf{x} \geq 0 \end{cases}$$

Having said that, we notice that there is a relation between the two error terms $\delta_{sign} = 2\delta_{step}$.

So, since we had the update:

$$\mathbf{w} = \mathbf{w} + \eta_{sign}\delta_{sign}\mathbf{x}$$

If we replace the δ_{sign} by the error for the step function we get:

$$\mathbf{w} = \mathbf{w} + 2\eta_{sign}\delta_{step}\mathbf{x}$$

To get exactly the same update, we need to define a learning rate which is twice the one we used $\eta_{step} = 2\eta_{sign}$ and get:

$$\mathbf{w} = \mathbf{w} + \eta_{step}\delta_{step}\mathbf{x}$$

4) The perceptron can learn a relatively large number of functions. In this exercise, we focus on simple logical functions.

a) Show graphically that a perceptron can learn the logical NOT function. Give an example with specific weights.

In this kind of exercise, we only need to show there is a hyperplane that separates what belongs to either class +1 or class -1. (Hyperplane is a fancy word for point, line or plane, depending on the number of dimensions.)

Solution:

The NOT function receives as input a logical value $x \in \{-1, +1\}$ and outputs its logical negation $t \in \{-1, +1\}$. We can enumerate all possible inputs and their inputs:

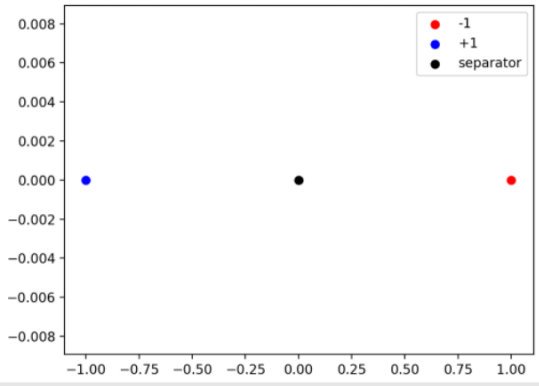
- For $x = -1$ the output is $t = +1$
- For $x = +1$ the output is $t = -1$

To show that a perceptron can learn a given function we just need to show that all points that require a positive output $(+1)$ can be separated from all points that require a negative output by an hyperplane.

Since we are working with 1 dimensional inputs, an hyperplane is, in this case, a point.

So, is there a point that accurately separates the points? Yes, any point between -1 and $+1$ will achieve this.

An example comes from setting the perceptron weights to zero:



2 Gradient descent learning

Gradient descent itself is not hard, but it takes very, very long time to perform even a single iteration. Calculating the derivatives of the error and output functions usually takes centuries as well. Thus, exercises that ask us to determine the learning rule or compute iterations for this algorithm take huge, unpractical amounts of time. In any case, here goes.

3) Consider the following training data:

$$\left\{\mathbf{x}^{(1)}=\begin{pmatrix}1\\1\end{pmatrix},\mathbf{x}^{(2)}=\begin{pmatrix}2\\1\end{pmatrix},\mathbf{x}^{(3)}=\begin{pmatrix}1\\3\end{pmatrix},\mathbf{x}^{(4)}=\begin{pmatrix}3\\3\end{pmatrix}\right\}$$

$$\left\{t^{(1)}=1,t^{(2)}=1,t^{(3)}=0,t^{(4)}=0\right\}$$

In this exercise, we will work with a unit that computes the following function:

$$output\left(\mathbf{x};\mathbf{w}\right)=\exp\left(\left(\mathbf{w}\cdot\mathbf{x}\right)^2\right)$$

And we will use the half sum of squared errors as our error (loss) function:

$$E\left(\mathbf{w}\right)=\frac{1}{2}\sum_{k=1}^{N}\left(t^{(k)}-output\left(\mathbf{x}^{(k)};\mathbf{w}\right)\right)$$

a) Determine the gradient descent learning rule for this unit.

This is a simple example, with "short" calculations.

Solution:

To apply gradient descent, we want an update rule that moves a step of size η towards the opposite direction from the gradient of the error function with respect to the weights:

$$\mathbf{w}=\mathbf{w}-\eta\frac{\partial E\left(\mathbf{w}\right)}{\partial\mathbf{w}}$$

To find the learning rule, we must compute the gradient of the error function with respect to the parameter vector.

$$\begin{aligned}\frac{\partial E\left(\mathbf{w}\right)}{\partial\mathbf{w}}&=\frac{\partial\frac{1}{2}\sum_{k=1}^{N}\left(t^{(k)}-output\left(\mathbf{x}^{(k)};\mathbf{w}\right)\right)^2}{\partial\mathbf{w}}\\&=\frac{\partial\frac{1}{2}\sum_{k=1}^{N}\left(t^{(k)}-\exp\left(\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)^2\right)\right)^2}{\partial\mathbf{w}}\\&=\frac{1}{2}\frac{\partial\sum_{k=1}^{N}\left(t^{(k)}-\exp\left(\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)^2\right)\right)^2}{\partial\mathbf{w}}\\&=\frac{1}{2}\sum_{k=1}^{N}\frac{\partial\left(t^{(k)}-\exp\left(\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)^2\right)\right)^2}{\partial\mathbf{w}}\end{aligned}$$

$$f(x)=\left(a-\exp\left(g(x)^2\right)\right)^2$$

$$f'(x)=2\left(a-\exp\left(g(x)^2\right)\right)\left(-\exp\left(g(x)^2\right)\right)2g(x)g'(x)$$

$$=\frac{1}{2}\sum_{k=1}^{N}\left(2\left(t^{(k)}-\exp\left(\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)^2\right)\right)(-1)\exp\left(\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)^2\right)2\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)\mathbf{x}^{(k)}\right)$$

$$=-2\sum_{k=1}^{N}\left(\left(t^{(k)}-\exp\left(\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)^2\right)\right)\exp\left(\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)^2\right)\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)\mathbf{x}^{(k)}\right)$$

So, we can write our update rule as follows.

$$\begin{aligned}\mathbf{w}&=\mathbf{w}-\eta\frac{\partial E\left(\mathbf{w}\right)}{\partial\mathbf{w}}\\&=\mathbf{w}-\eta\left(-2\sum_{k=1}^{N}\left(\left(t^{(k)}-\exp\left(\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)^2\right)\right)\exp\left(\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)^2\right)\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)\mathbf{x}^{(k)}\right)\right)\\&=\mathbf{w}+2\eta\sum_{k=1}^{N}\left(\left(t^{(k)}-\exp\left(\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)^2\right)\right)\exp\left(\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)^2\right)\left(\mathbf{w}\cdot\mathbf{x}^{(k)}\right)\mathbf{x}^{(k)}\right)\end{aligned}$$

b) Compute the stochastic gradient descent update for input $\mathbf{x} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, t = 0$ initialized with $\mathbf{w} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$ and learning rate $\eta = 2$.

Solution:

In stochastic gradient descent we make one update for each training example. So, instead of summing accross all data points we adapt the learning rule for one example only:

$$\mathbf{w} = \mathbf{w} + 2\eta \left(\left(t - \exp\left((\mathbf{w} \cdot \mathbf{x})^2 \right) \right) \exp\left((\mathbf{w} \cdot \mathbf{x})^2 \right) (\mathbf{w} \cdot \mathbf{x}) \mathbf{x} \right)$$

We can now do the updates. Let us start with the first example:

$$\mathbf{w} = \mathbf{w} + 2\eta \left(\left(t - \exp\left((\mathbf{w} \cdot \mathbf{x})^2 \right) \right) \exp\left((\mathbf{w} \cdot \mathbf{x})^2 \right) (\mathbf{w} \cdot \mathbf{x}) \mathbf{x} \right)$$

$$= \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + 2\,(2) \left(0 - \exp\left(\left(\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \right)^2 \right) \right)$$

$$\exp\left(\left(\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \right)^2 \right) \left(\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \right) \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + 4 \left((0 - \exp(1)) \exp(1) (1) \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \right)$$

$$= \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} - 4 \begin{pmatrix} e^2 \\ e^2 \\ e^2 \end{pmatrix}$$

$$= \begin{pmatrix} -4e^2 \\ 1 - 4e^2 \\ -4e^2 \end{pmatrix}$$

A non stochastic approach, from another exercise, not including the calculation of the gradient of the error function:

$$
\begin{aligned}
\mathbf{w} &= \mathbf{w} + 2\eta \sum_{k=1}^{4} \left(\left(t^{(k)} - \sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(k)}\right)\right) \left(\sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(k)}\right) \left(1 - \sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(k)}\right)\right)\right) \mathbf{x}^{(k)} \right) \\
&= \mathbf{w} + \sum_{k=1}^{4} 2\eta \left(\left(t^{(k)} - \sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(k)}\right)\right) \left(\sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(k)}\right) \left(1 - \sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(k)}\right)\right)\right) \mathbf{x}^{(k)} \right) \\
&= \mathbf{w} + 2\eta \left(\left(t^{(1)} - \sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(1)}\right)\right) \left(\sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(1)}\right) \left(1 - \sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(1)}\right)\right)\right) \mathbf{x}^{(1)} \right) + \\
&+ 2\eta \left(\left(t^{(2)} - \sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(2)}\right)\right) \left(\sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(2)}\right) \left(1 - \sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(2)}\right)\right)\right) \mathbf{x}^{(2)} \right) + \\
&+ 2\eta \left(\left(t^{(3)} - \sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(3)}\right)\right) \left(\sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(3)}\right) \left(1 - \sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(3)}\right)\right)\right) \mathbf{x}^{(3)} \right) + \\
&+ 2\eta \left(\left(t^{(4)} - \sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(4)}\right)\right) \left(\sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(4)}\right) \left(1 - \sigma\left(2\mathbf{w}\cdot\mathbf{x}^{(4)}\right)\right)\right) \mathbf{x}^{(4)} \right) = \\
&= \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \\
&+ 2 \left(1 - \sigma\left(2 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}\right)\right) \sigma\left(2 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}\right) \left(1 - \sigma\left(2 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}\right)\right) \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \\
&+ 2 \left(1 - \sigma\left(2 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}\right)\right) \sigma\left(2 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}\right) \left(1 - \sigma\left(2 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}\right)\right) \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} + \\
&+ 2 \left(0 - \sigma\left(2 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 3 \end{pmatrix}\right)\right) \sigma\left(2 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 3 \end{pmatrix}\right) \left(1 - \sigma\left(2 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 3 \end{pmatrix}\right)\right) \begin{pmatrix} 1 \\ 1 \\ 3 \end{pmatrix} + \\
&+ 2 \left(0 - \sigma\left(2 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 3 \\ 3 \end{pmatrix}\right)\right) \sigma\left(2 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 3 \\ 3 \end{pmatrix}\right) \left(1 - \sigma\left(2 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 3 \\ 3 \end{pmatrix}\right)\right) \begin{pmatrix} 1 \\ 3 \\ 3 \end{pmatrix} = \\
&= \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + 2\left(1 - \sigma(6)\right)\left(\sigma(6)\left(1 - \sigma(6)\right)\right) \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + 2\left(1 - \sigma(8)\right)\left(\sigma(8)\left(1 - \sigma(8)\right)\right) \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} + \\
&+ 2\left(0 - \sigma(10)\right)\left(\sigma(10)\left(1 - \sigma(10)\right)\right) \begin{pmatrix} 1 \\ 1 \\ 3 \end{pmatrix} + 2\left(0 - \sigma(14)\right)\left(\sigma(14)\left(1 - \sigma(14)\right)\right) \begin{pmatrix} 1 \\ 3 \\ 3 \end{pmatrix} = \\
&= \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \begin{pmatrix} 1.2197 \times 10^{-5} \\ 1.2197 \times 10^{-5} \\ 1.2197 \times 10^{-5} \end{pmatrix} + \begin{pmatrix} 2.2484 \times 10^{-7} \\ 4.4969 \times 10^{-7} \\ 2.2484 \times 10^{-7} \end{pmatrix} + \\
&+ \begin{pmatrix} -9.0787 \times 10^{-5} \\ -9.0787 \times 10^{-5} \\ -2.7236 \times 10^{-4} \end{pmatrix} + \begin{pmatrix} -1.6631 \times 10^{-6} \\ -4.9892 \times 10^{-6} \\ -4.9892 \times 10^{-6} \end{pmatrix} = \\
&= \begin{pmatrix} 0.99991997 \\ 0.99991687 \\ 0.99973507 \end{pmatrix}
\end{aligned}
$$

(I can only hope that the professor is sane enough not to actually put something like this in the exam. Good luck to all of us.)