INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO

Neural Networks

"Mcculloch-Pitts"

Abstact

The following work use the Mcculloch-Pitts Neural Network to implement the logic gates AND, OR (2 entries) and NOT using program MATLAB.

By:

Ricardo Vargas Sagrero

Professor:
Dr. Marco Antonio Moreno Armendariz

August 2018

Contents

No table of contents entries found.

Introducción

La neurona de Mcculloch-Pitts es conocida como el primer modelo neural moderno, fue propuesto por en 1943. Es un modelo binario con estados de apagado (0) y encendido (1). Su modelo matemático hace uso de un umbral θ que determina la salida de la neurona.

Este tipo de neuronas tiene un aprendizaje a prueba y error, también, este tipo de célula puede resolver cualquier problema aritmético y lógico excepto aquellos que no son linealmente separables, como lo es la compuerta XOR.

Arquitectura y modelo matemático

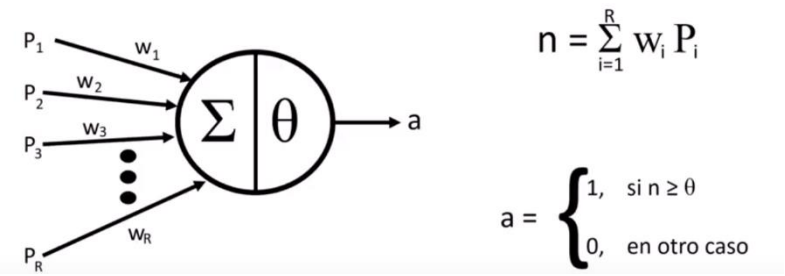


Figura 1.Modelo matemático de celula McCulloch-Pitts

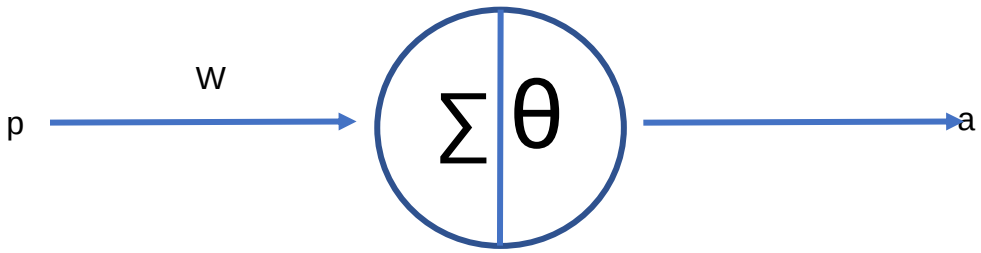
Su modelo matemático se divide en acumulación de energía (izquierda) y comparación con el umbral (derecha), si la suma de los productos es mayor o igual tema la salida es 1, en otro caso es 0.

Diagrama de flujo del programa

Este programa tiene la finalidad de implementar una neurona de Mcculloch-Pitts para las compuertas AND, OR (2 entradas) y NOT.

Compuerta NOT

La primera compuerta por implementar será la NOT con la siguiente arquitectura:



Se tiene el siguiente conjunto de entrenamiento

	P	**Target**
D1	0	1
D2	1	0

El entrenamiento de esta neurona es aprueba y error por lo cual debemos de asignar valor a W y a θ, para nuestro primer intento asignaremos los valores w = 1, θ = 1:

Propagamos D1:

$$n = w\,p = (1)(0) = 0$$

Ahora preguntamos si 0 >= 1, la respuesta es no, entonces a = 0.

Comparamos a con t1 (target 1):

$$a == t1$$

No

Propagamos D2:

Se proponen los valores w = 1, θ = -1:

$$n = w\,p = (1)(0) = 0$$

Ahora preguntamos si 0 >= 0, la respuesta es si, entonces a = 1.

Comparamos a con t1 (target 1):

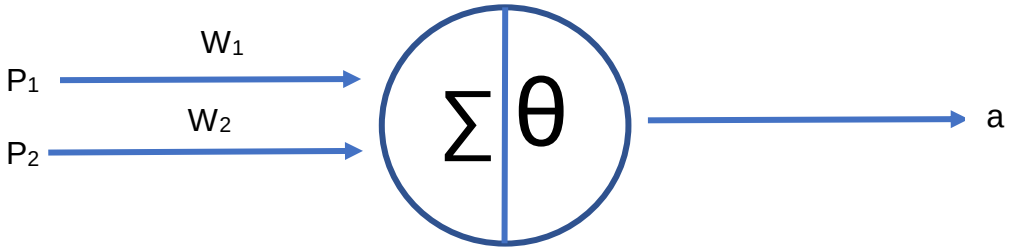
$$a = t1 => 1 = 1$$

SI

Como vemos en la segunda propagación la neurona a lanzado los resultados esperados, por lo tanto, el aprendizaje es exitoso.

Compuerta AND

La siguiente compuerta será la compuerta AND con la siguiente arquitectura a implementar:



	P_1	P_2	Target
D1	0	0	0
D2	0	1	0
D3	1	0	0
D4	1	1	1

Figure 2. Tabla de verdad compuerta AND

Se procede con el entrenamiento de la neurona ingresando valores para W y θ, para nuestro primer intento W1 y W2 = 2, θ = 4:

Propagamos D1:

$$n = w\,p_{1,1} = (2)(0) = 0$$

$$n = w\, p_{1,2} = (2)(0) = 0$$

Ahora preguntamos si 0 >= 1, la respuesta es NO, entonces a = 0.

Comparamos a con t1 (target 1):

$$a = t1 => 0 = 0$$ SI

Se procede con los siguientes valores :

$$n = w\, p_{2,1} = (2)(0) = \ 0$$

Comparación con umbral 0 >= 1, la respuesta es NO, entonces a = 0

Comparamos con t2 (target 2):

$$a = t2 => 0 = 0$$ SI

$$n = w\, p_{2,2} = (2)(1) = \ 1$$

Comparación con umbral 1 >= 1, la respuesta es NO, entonces a = 0

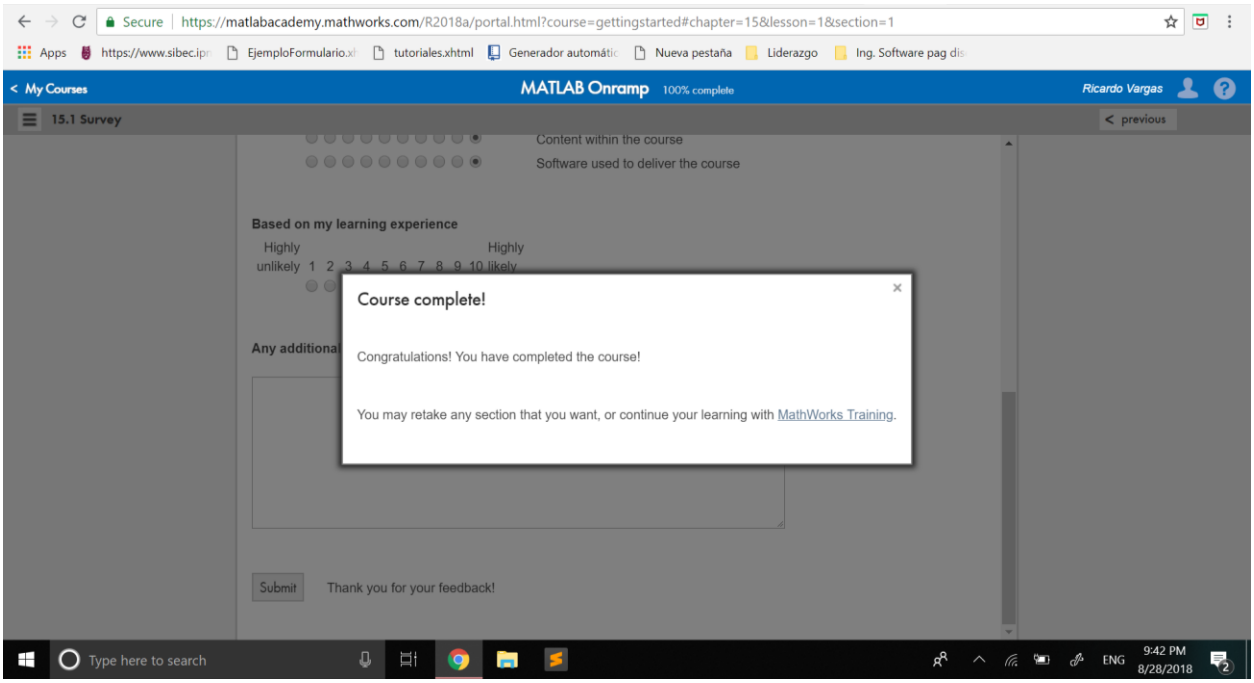
Comparamos con t3 (target 3):

$$a = t2 => 0 = 0$$ SI

Conclusiones

Por fallas técnicas no se pudieron terminar todos los ejercicios, solo la compuerta NOT fue implementada exitosamente. De la cual podemos concluir que con este tipo de neuronas al no poder tener un aprendizaje automático se tiene que probar distintos valores de pesos sinápticos y valores de umbral para llegar a un aprendizaje exitoso. Lo cual seria lo mismo para todos los problemas lógicos y aritméticos que esta neurona puede resolver.

Captura de pantalla del curso MATLAB Onramp



Apéndice

```
%% Compuertas NOT, AND, OR con neurona Mcculluch-
pitts
% Primero se tiene la tabla de verdad de la
compuerta

tVerdad = [0,1;1,0];
%%
% Como el entrenamiento es a prueba y error
entonces asignamos los valores
% de w y teta

w = 1;
teta = -1;
%%
% Ahora probramos el modelo matematico
```

```
%
% n = p * w

n = tVerdad(1,1)* w;
%%
% Entramos a la parte de la comparacion con el
umbral n > teta

for i = 1:2
    n = tVerdad(i,i)* w;
    if ( n > teta)
        a = 0;
        if(a == tVerdad(i,i))
            resultadoNOT(1,i) = 1;
        else
            resultadoNOT(1,i) = 0;
        end
    end
end
%%
% Solo se obtuvieron los resultados esperados con
teta = 0;
%% *Compuerta AND de 2 entradas*
% Tabla de verdad de la compuerta AND de 2
entradas

tand = [0,0,0;0,1,0;1,0,0;1,1,1];
%%
% **
```