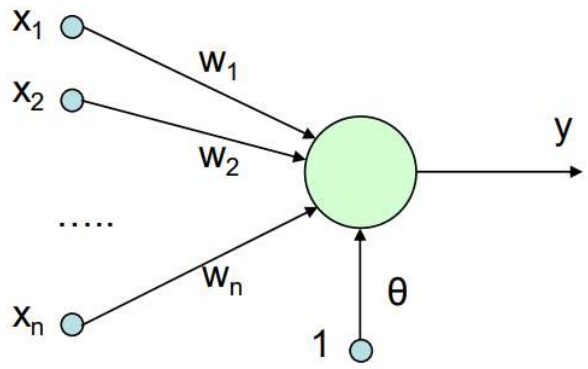
Estudiante: Vinicio Veletanga

RED NEURONAL ADALINE



Adaline se caracteristisa por que utiliza directamente la salida de la red teniendo en cuenta en minimizar el error.Donde tiene como objetivo una red tal que la salida sea igual a la salida obtenida.

REGLA DELTA O REGLA APRENDIZAJE

Esta regla busca realizar un conjunto de pesos que nos permita minimizar la funcion de error.

$$\Delta w_i = \alpha \sum_{\forall p} (d_p - y_p) x_i$$

Realizando procesos interactivos donde se van modificando los pesos. Donde cada cambio en cada peso proporcional a la derivada del error. El entrenamiento se realiza presentando repetidamente una serie de parejas de entradasy salidas.

Regla de Widrow-Hoff

$$\Delta w_i = \eta \, a_i \, (t * x)$$

siendo η la constante de aprendizaje, ai la salida de la unidad i, t la salida deseada y por último x la salida de la unidad Adaline. No obstante la variante de esta regla más utilizada considera el valor de la suma ponderada S en vez del valor de la salida de la unidad Adaline.

Parámetros iniciales de la red neuronal artificial

$$\vec{W} = [0.84, 0.394, 0.783]$$

$$\vec{X} = \begin{bmatrix} \mathbf{x1} & \mathbf{x2} & \mathbf{x3} & \delta \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 1 & 1 & 3 \\ 1 & 0 & 0 & 4 \\ 1 & 0 & 0 & 5 \\ 1 & 1 & 0 & 6 \\ 1 & 1 & 1 & 7 \end{bmatrix}$$

$$\alpha = 0.3$$

Función de Widrow-Hoff

In [44]:

```
1 import matplotlib.pyplot as plt
2 import numpy as np
```

In [37]:

```
# PESOS
w=np.array([0.84,0.394,0.783])
alpha=0.3
X=np.array([[0, 0, 1], [0, 1, 0], [0, 1, 1], [1, 0, 0], [1, 0, 1], [1, 1, 0],[1, 1, 1]]
r=[1, 2, 3, 4, 5, 6, 7]
lon=len(r)
res=0
con=True
cont=1
errores=[]
xp1=[0, 0, 1]
xp2=[0, 1, 0]
xp3=[0, 1, 1]
xp4=[1, 0, 0]

xp5=[1, 0, 1]
xp6=[1, 1, 0]
xp7=[1, 1, 1]
print("PASO: ",cont)
for i in range(lon):
        print("\n")
        print("Patrón ",(i+1) , " = (", X[i][0],",",X[i][1],",",X[i][2],")")

        val=X[i][0]*w[0]+X[i][1]*w[1]+X[i][2]*w[2]
        print("W1:",w[0],", x1:",X[i][0],", W2:",w[1],",X2:",X[i][1],", W3:",w[2],",X3:
        print("Esperado: ",r[i])
        print("Obtenido: ",val)
        ER= r[i]-val
        print("\033[4;35m","Valor de Error : ",ER,'\033[0;m')
        print("SE CALCULA LOS NUEVOS PESOS")

        w[0] = w[0]+(alpha*ER)*X[i][0]
        w[1] = w[1]+(alpha*ER)*X[i][1]
        w[2] = w[2]+(alpha*ER)*X[i][2]
        print("Peso W1: ",w[0],"Peso W2: ",w[1],"Peso W3: ",w[2])


```

```
PASO:  1


Patrón  1  = ( 0 , 0 , 1 )
W1: 0.84 , x1: 0 , W2: 0.394 ,X2: 0 , W3: 0.783 ,X3: 1
Esperado:  1
Obtenido:  0.783
 Valor de Error :  0.21699999999999997
SE CALCULA LOS NUEVOS PESOS
Peso W1:  0.84 Peso W2:  0.394 Peso W3:  0.8481000000000001


Patrón  2  = ( 0 , 1 , 0 )
W1: 0.84 , x1: 0 , W2: 0.394 ,X2: 1 , W3: 0.8481000000000001 ,X3: 0
Esperado:  2
Obtenido:  0.394
 Valor de Error :  1.6059999999999999
SE CALCULA LOS NUEVOS PESOS
Peso W1:  0.84 Peso W2:  0.8757999999999999 Peso W3:  0.8481000000000001
```

```
Patrón  3  = ( 0 , 1 , 1 )
W1: 0.84 , x1: 0 , W2: 0.8757999999999999 ,X2: 1 , W3: 0.8481000000000001 ,X
3: 1
Esperado:  3
Obtenido:  1.7239
 Valor de Error :  1.2761
SE CALCULA LOS NUEVOS PESOS
Peso W1:  0.84 Peso W2:  1.25863 Peso W3:  1.23093


Patrón  4  = ( 1 , 0 , 0 )
W1: 0.84 , x1: 1 , W2: 1.25863 ,X2: 0 , W3: 1.23093 ,X3: 0
Esperado:  4
Obtenido:  0.84
 Valor de Error :  3.16
SE CALCULA LOS NUEVOS PESOS
Peso W1:  1.7879999999999998 Peso W2:  1.25863 Peso W3:  1.23093


Patrón  5  = ( 1 , 0 , 1 )
W1: 1.7879999999999998 , x1: 1 , W2: 1.25863 ,X2: 0 , W3: 1.23093 ,X3: 1
Esperado:  5
Obtenido:  3.01893
 Valor de Error :  1.9810699999999999
SE CALCULA LOS NUEVOS PESOS
Peso W1:  2.3823209999999997 Peso W2:  1.25863 Peso W3:  1.8252510000000002


Patrón  6  = ( 1 , 1 , 0 )
W1: 2.3823209999999997 , x1: 1 , W2: 1.25863 ,X2: 1 , W3: 1.8252510000000002
,X3: 0
Esperado:  6
Obtenido:  3.6409509999999994
 Valor de Error :  2.3590490000000006
SE CALCULA LOS NUEVOS PESOS
Peso W1:  3.0900356999999996 Peso W2:  1.9663447 Peso W3:  1.825251000000000
2


Patrón  7  = ( 1 , 1 , 1 )
W1: 3.0900356999999996 , x1: 1 , W2: 1.9663447 ,X2: 1 , W3: 1.82525100000000
02 ,X3: 1
Esperado:  7
Obtenido:  6.8816314
 Valor de Error :  0.11836860000000016
SE CALCULA LOS NUEVOS PESOS
Peso W1:  3.1255462799999996 Peso W2:  2.00185528 Peso W3:  1.86076158000000
01
```

REFERENCIAS:

Xabier Basogain Olabe,2019.Redes Neuronales Artificiales y sus aplicaciones.Escuela Superior de Ingenieria de Bilbao.

https://fdocuments.ec/document/redes-neuronales-artificiales-y-sus-aplicaciones-computacion-tradicional-y.html (https://fdocuments.ec/document/redes-neuronales-artificiales-y-sus-aplicaciones-computacion-tradicional-y.html)

In [ ]:

```
1
```