Instituto Politécnico Nacional

Escuela Superior de Cómputo

Asignatura:

Neural Networks

Grupo: 3CM2

Práctica .4 **Red ADALINE**

Alumno:

Garcia Garcia Rafael

Profesor: Moreno Armendariz Marco Antonio

Introducción:

En la práctica pasada se explicó del nacimiento, funcionamiento y comportamiento de la red perceptrón y como puede ser aplicada a problemas sencillos. Una red muy similar es la Adaptative Linear Network o mejor conocida como red ADALINE. A finales de los años 50 Bernard Widrow empezaba a trabajar en redes neuronales, en el tiempo en el que Frank Rosenblatt desarrolló la regla de aprendizaje del perceptrón, años después, en 1960 se presenta la red ADALINE creada por Widrow y su asesorado, asimismo presentaron una regla de aprendizaje la cual denominaron algoritmo LMS (Least Mean Square).

La red ADALINE al tener similitudes al perceptrón y al ser su única diferencia que usa una función de transferencia linear en vez de hard limit, presenta las mismas

problemáticas: Solo pueden resolver problemas linealmente separables.

Marco teórico.

Explicación de la arquitectura,

ADALINE fue desarrollada por el profesor Bernard Widrow y su alumno Ted Hoff en la Universidad de Stanford en 1960. Como se observa en la figura 1 (Hagan), tiene la misma arquitectura que el perceptrón a diferencia de la función de transferencia purelin().

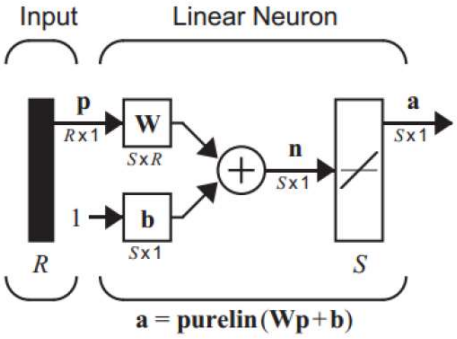
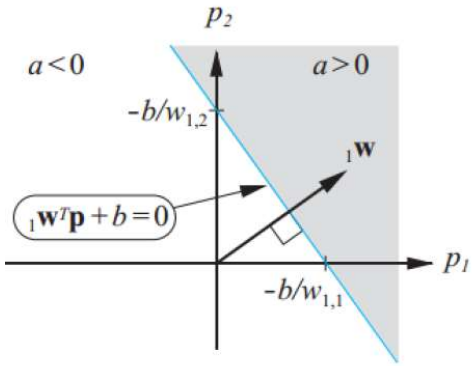


Figura 1



La salida de la neurona es mayor que 0 en el área gris. En el área blanca la salida es menor que cero. Ahora, ¿qué implica esto sobre el ADALINE?

Dice que ADALINE se puede usar para clasificar objetos en dos categorías. Sin embargo, solo puede hacerlo si los objetos son linealmente separables. Por lo tanto, a este respecto, el ADALINE tiene la misma limitación que el perceptrón.

Esto nos lleva a poder hablar más de la regla de aprendizaje de la red ADALINE: El algoritmo LMS, Dado que el objetivo del ADALINE es poder estimar de la manera más exacta la salida, se busca minimizar la desviación de la red para todos los patrones de entrada, eligiendo una medida del error global. Normalmente se utiliza el error cuadrático medio.

$$E = \frac{1}{2}\sum_{p=1}^{m}(d^p - y^p)^2$$

La manera de reducir este error global es ir modificando los valores de los pesos al procesar cada entrada, de forma iterativa, mediante la regla del descenso del gradiente. Suponiendo que tenemos una constante de aprendizaje alfa.

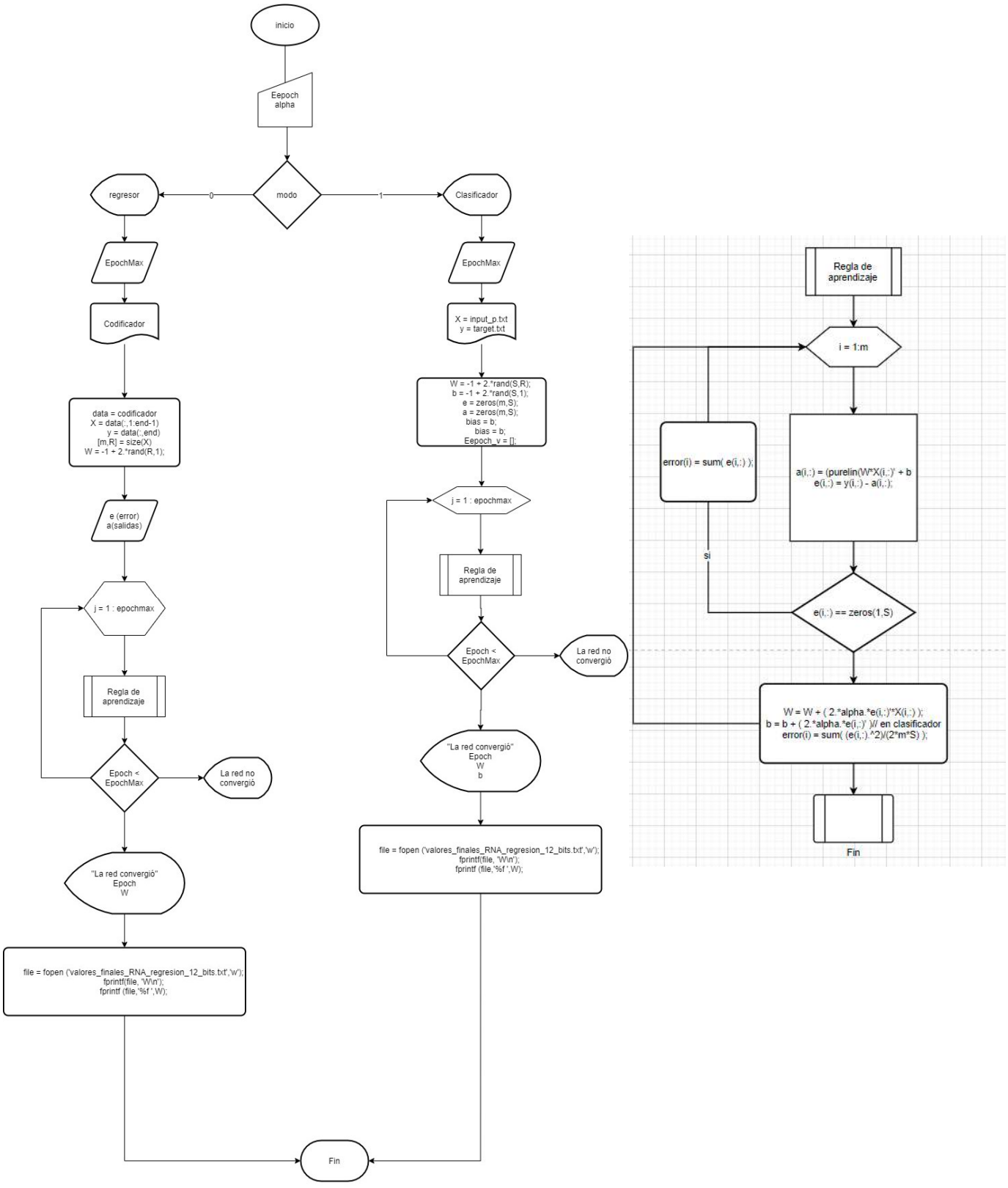
$$\Delta_p w_j = -\alpha \frac{\partial E^p}{\partial w_j}$$

Si operamos con la derivada, queda:

$$\Delta_p w_j = \alpha(d^p - y^p) \cdot x_j$$

Que será la expresión que utilizaremos por cada entrada para modificar los pesos. Con respecto al perceptrón, la red ADALINE posee la ventaja de que su gráfica de error es un hiperparaboloide que posee o bien un único mínimo global, o bien una recta de infinitos mínimos, todos ellos globales. Esto evita la gran cantidad de problemas que da el perceptrón a la hora del entrenamiento debido a que su función de error (también llamada de coste) posee numerosos mínimos locales.

Diagrama de Flujo:



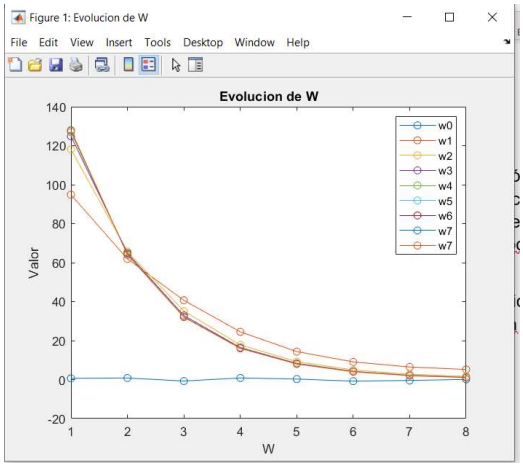
Experimentos.

Modo regresor:

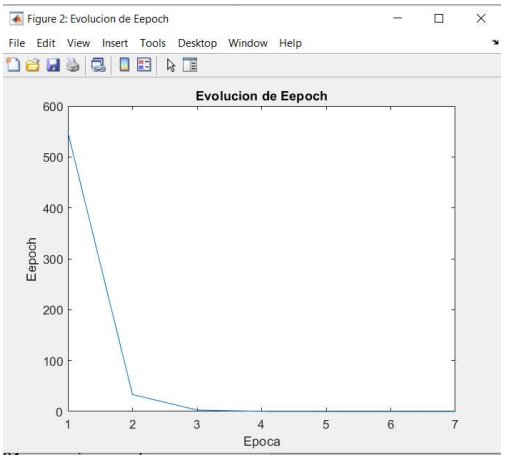
Experimento 1:

Para este experimento se tuvo que desarrollar una función el cuál pudiera crear un codificador de 8 bits, este codificador se mete como entrada para el entrenamiento de nuestra red neuronal ADALINE. Para este ejemplo decidimos poner un valor de EEPOCH de 0.001, un número de EpochMax = 10 y Alpha de 0.04 para ver si nuestra red converge.

Se realiza el experimento y como se observa en las gráficas la red converge de manera muy rápida, además de que el valor de Eepoch desciende de manera significativa.



Gráfica 1. Evolución pesos Codificador 8 bits



Gráfica 2. Evolución Eepoch Codificador 8 bits

Como salida de nuestra terminal se muestran los valores de W, y de Eepoch, además de que va contando el número de época en el que nos encontramos.

Hola, Bienvenido a la red Adaline\n Seleccione 0 Modo REGRESOR 1 modo CLASIFICADOR otro SALIR 0 Valor de Alpha: 0.0400 Valor de Eepoch 1.0000e-03 Selecciono la red en modo REGRESOR Introduzca el numero de epochMax 10	La red convergió Valores finales Epoch 3.6168e-04 W 127.9645 64.0179 32.0067 16.0037 8.0029 4.0020 2.0016 1.0014

epochMax = 10 Generando matriz de pesos... Epoca número: 1 Epoca número: 2 Epoca número: 3 Epoca número: 4 Epoca número: 5 Epoca número: 6 Epoca número: 7	Fin del programa espero verte printo :D >>

En conclusión el experimento funcionó bastante bien para la entrada y se puede verificar esto con la salida que tuvimos en nuestro vector **a** con nuestro últimos valores de salida.

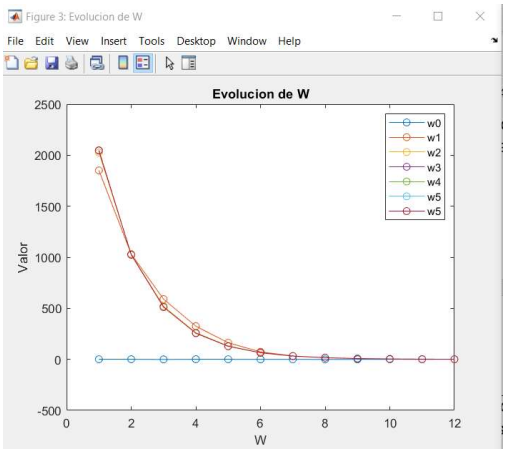
0	10.0098	19.0064
1.0043	11.0092	20.0097
2.0049	12.0096	21.0060
3.0085	13.0082	22.0057
4.0064	14.0075	23.0017
5.0091	15.0053	24.0103
6.0090	16.0116	25.0058
7.0101	17.0098	26.0056
8.0090	18.0097	27.0010
9.0100		

Tabla.1 salida vector **a**

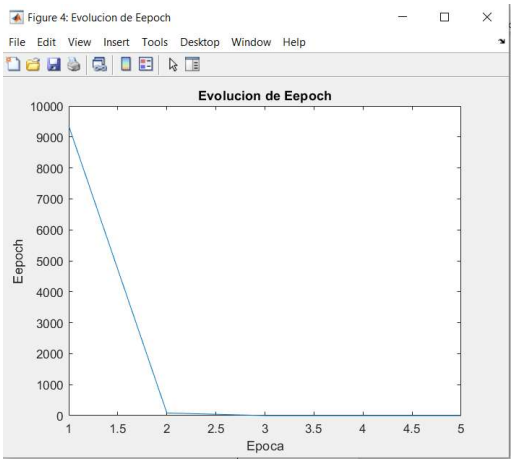
Experimento 2:

Ahora se empleó un codificador de 12 bits, este codificador se mete como entrada para el entrenamiento de nuestra red neuronal ADALINE. Para este ejemplo decidimos poner un valor de EEPOCH de 0.001, un número de **EpochMax** = 10 y **Alpha** de 0.04 para ver si nuestra red converge.

De igual manera como en el ejercicio anterior convergió muy rápido, además de que los resultados a la hora de abrir la salida son bastante parecidos al dataset con el que se entrenó.



Gráfica 3. Evolución pesos Codificador 12 bits



Gráfica 4. Evolución Eepoch Codificador 12 bits

Como salida de nuestra terminal se muestran los valores de W, y de Eepoch, además de que va contando el número de época en el que nos encontramos.

```
>> Adaline_p4
Hola, Bienvenido a la red Adaline\n
Seleccione 0 Modo REGRESOR 1 modo CLASIFICADOR otro SALIR
0
Valor de Alpha:
    0.0400

Valor de Eepoch
   1.0000e-03

Selecciono la red en modo REGRESOR
Introduzca el numero de epochMax
10
Generando matriz de pesos...
Epoca número: 1
Epoca número: 2
Epoca número: 3
Epoca número: 4
Epoca número: 5
La red convergió
Valores finales
Epoch
   2.0475e-04
```

```
W
   1.0e+03 *

    2.0480
    1.0240
    0.5120
    0.2560
    0.1280
    0.0640
    0.0320
    0.0160
    0.0080
    0.0040
    0.0020
    0.0010

Fin del programa espero verte printo :D
>>
```

En conclusión el experimento funcionó bastante bien para la entrada y se puede verificar esto con la salida que tuvimos en nuestro vector **a** con nuestro últimos valores de salida.

	1
1	0
2	0.9999
3	1.9998
4	2.9997
5	3.9998
6	4.9997
7	5.9998
8	6.9997
9	7.9998
10	8.9997
11	9.9998

10.9997
11.9998
12.9998
13.9998
14.9999
15.9996
16.9997
17.9997
18.9997
19.9997

20.9998
21.9998
22.9999
23.9997
24.9998
25.9998
27.0000
27.9999
29.0000

Tabla.2 salida vector **a**

Modo clasificador:

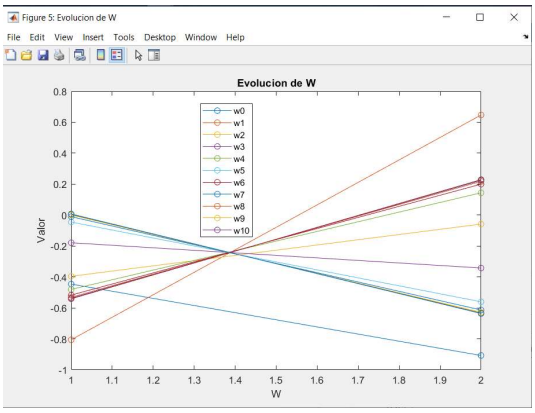
Experimento 1:

Para el segundo ejemplo se usa un data set compuesto de 4 clases como el que se muestra en la figura 3 (Hagan).

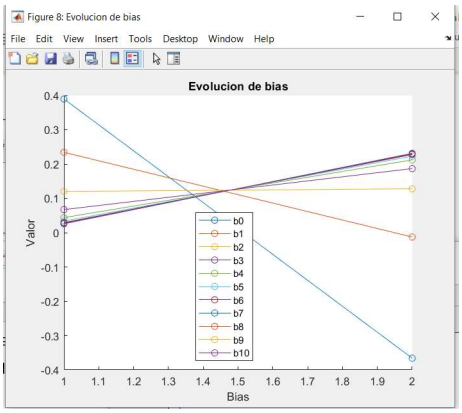
$$\left\{\mathbf{p}_1 = \begin{bmatrix}1\\1\end{bmatrix}, \mathbf{t}_1 = \begin{bmatrix}-1\\-1\end{bmatrix}\right\}\left\{\mathbf{p}_2 = \begin{bmatrix}1\\2\end{bmatrix}, \mathbf{t}_2 = \begin{bmatrix}-1\\-1\end{bmatrix}\right\}\left\{\mathbf{p}_3 = \begin{bmatrix}2\\-1\end{bmatrix}, \mathbf{t}_3 = \begin{bmatrix}-1\\1\end{bmatrix}\right\}$$
$$\left\{\mathbf{p}_4 = \begin{bmatrix}2\\0\end{bmatrix}, \mathbf{t}_4 = \begin{bmatrix}-1\\1\end{bmatrix}\right\}\left\{\mathbf{p}_5 = \begin{bmatrix}-1\\2\end{bmatrix}, \mathbf{t}_5 = \begin{bmatrix}1\\-1\end{bmatrix}\right\}\left\{\mathbf{p}_6 = \begin{bmatrix}-2\\1\end{bmatrix}, \mathbf{t}_6 = \begin{bmatrix}1\\-1\end{bmatrix}\right\}$$
$$\left\{\mathbf{p}_7 = \begin{bmatrix}-1\\-1\end{bmatrix}, \mathbf{t}_7 = \begin{bmatrix}1\\1\end{bmatrix}\right\}\left\{\mathbf{p}_8 = \begin{bmatrix}-2\\-2\end{bmatrix}, \mathbf{t}_8 = \begin{bmatrix}1\\1\end{bmatrix}\right\}$$

Figura 2

Para este ejercicio se requirieron 2 neuronas debido al número de clases, estas neuronas tienen un vector de pesos los cuales definirán la frontera de decisión de nuestro ejercicio las cuales estarán ilustradas en las siguientes gráficas junto a los valores de EEPOCH y de pesos y bias.



Gráfica 5. Evolución pesos Clasificador 4 clases



Gráfica 6. Evolución Bias clasificar 4 clases

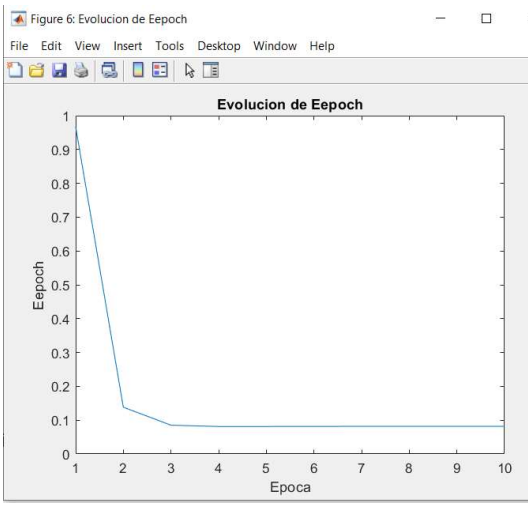
En la gráfica 5 vemos como al tener 2 neuronas tenemos una convergencia hacia unos valores de pesos en específico, esto quiere decir que en la teoría debe clasificar bien las entradas que le introduzcamos. De igual manera con el bias, pero como este no como no tiene solo tiene 1 vector columna solo tiene una línea que converge a cierto valor como se ve en la gráfica 6.

Como salida de nuestra terminal se muestran los valores de W, y de Eepoch, además de que va contando el número de época en el que nos encontramos.

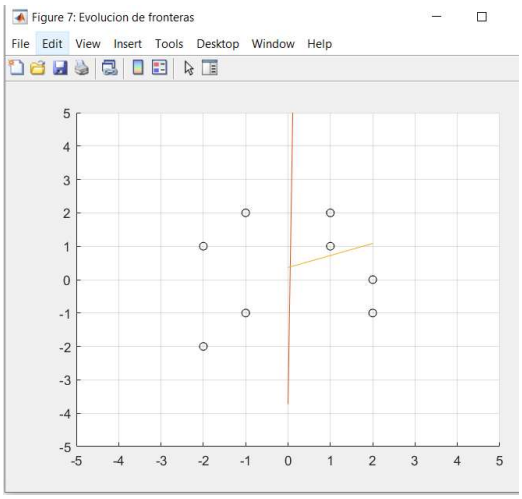
```
>> Adaline_p4
Hola, Bienvenido a la red Adaline\n
Seleccione 0 Modo REGRESOR 1 modo CLASIFICADOR otro SALIR
1
Valor de Alpha:
   0.0400

Valor de Eepoch
  1.0000e-03

Selecciono la red en modo CLASIFICADOR
El numero de rasgos es: 2
El numero de clases ingresadas es: 4
El numero de neuronas necesarias es: 2
Ingrese el numero de MAX_EPOCH:10
Epoca número: 1
Epoca número: 2
Epoca número: 3
Epoca número: 4
Epoca número: 5
Epoca número: 6
Epoca número: 7
Epoca número: 8
Epoca número: 9
Epoca número: 10
Epoch
   0.0818
```

```
W
  -0.5412   0.0069
   0.2289  -0.6363

Bias
   0.0259
   0.2313
```

Veamos que la salida de la terminal no converge a un valor tan diminuto en Eepoch pero el error sigue manteniéndose relativamente pequeño, por lo cuál se mostrará que a pesar de esto logra hacer su objetivo sin problema



Gráfica 7. Evolución Eepoch en clasificador 4 clases



Gráfica 8. Fronteras de decisión 4 clases

En conclusión el experimento fue todo un éxito por lo cuál podemos decir que clasifica de manera correcta

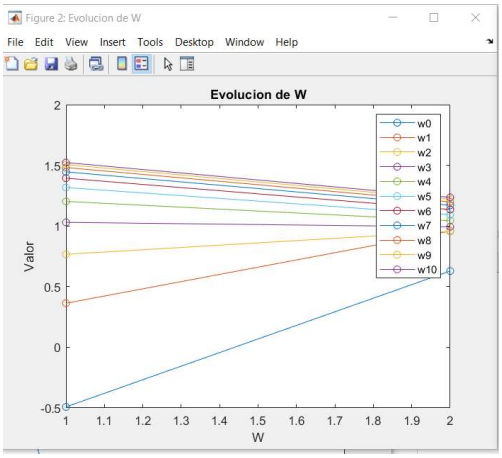
Experimento 2:

Este primer experimento se realizó con el Dataset de la compuerta AND por lo cuál el data set de entrada es el de la figura 2 (Hagan) donde sustituiremos los 0's por -1 ya que por la función de transferencia de Adaline tenemos valores de -1 a 1:

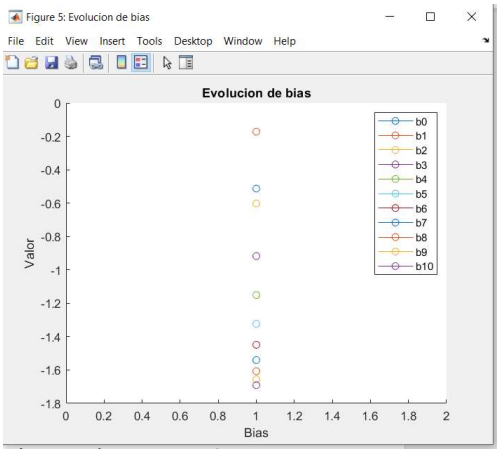
$$\left\{\mathbf{p}_1 = \begin{bmatrix}0\\0\end{bmatrix}, t_1 = 0\right\}\left\{\mathbf{p}_2 = \begin{bmatrix}0\\1\end{bmatrix}, t_2 = 0\right\}\left\{\mathbf{p}_3 = \begin{bmatrix}1\\0\end{bmatrix}, t_3 = 0\right\}\left\{\mathbf{p}_4 = \begin{bmatrix}1\\1\end{bmatrix}, t_4 = 1\right\}.$$

Figura 3

Para este ejercicio se requirió 1 neurona debido al número de clases, esta neurona tienen un vector de pesos el cual definirá la frontera de decisión de nuestro ejercicio las cuales estarán ilustradas en las siguientes gráficas junto a los valores de EEPOCH y de pesos y bias.



Gráfica 9. Evolución pesos Clasificador 2 clases



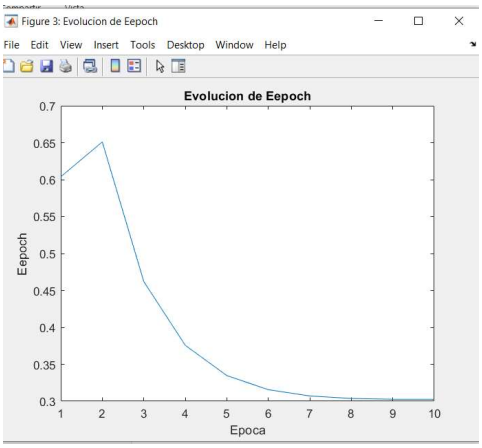
Gráfica 10. Evolución Bias clasificar 2 clases

Observemos como tanto en la gráfica 9 y en la 10 los valores de pesos y bias van convergiendo a un punto por lo cuál el error ya no va a tener mucha diferencia en las últimas épocas.

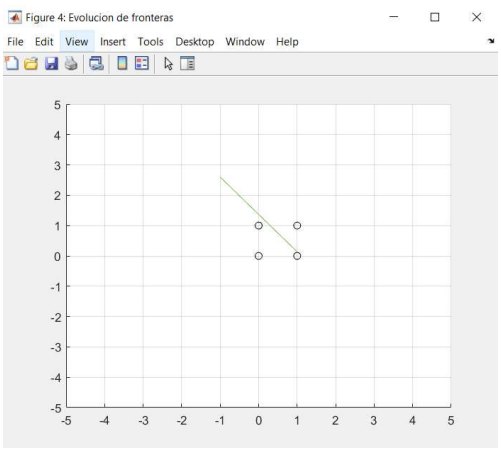
Como salida de nuestra terminal se muestran los valores de W, y de Eepoch, además de que va contando el número de época en el que nos encontramos.

>> Adaline_p4 Hola, Bienvenido a la red Adaline\n Seleccione 0 Modo REGRESOR 1 modo CLASIFICADOR otro SALIR 1 Valor de Alpha: 0.1800 Valor de Eepoch 1.0000e-03 Selecciono la red en modo CLASIFICADOR El numero de rasgos es: 2 El numero de clases ingresadas es: 2 El numero de neuronas necesarias es: 1 Ingrese el numero de MAX_EPOCH:10	Epoca número: 1 Epoca número: 2 Epoca número: 3 Epoca número: 4 Epoca número: 5 Epoca número: 6 Epoca número: 7 Epoca número: 8 Epoca número: 9 Epoca número: 10 Epoch 0.3026 W 1.5225 1.2358 Bias -1.6907

Veamos que la salida de la terminal no converge a un valor tan diminuto en Eepoch pero el error sigue manteniéndose relativamente pequeño, por lo cuál se mostrará que a pesar de esto logra hacer su objetivo sin problema



Gráfica 11. Evolución Eepoch en clasificador 2 clases



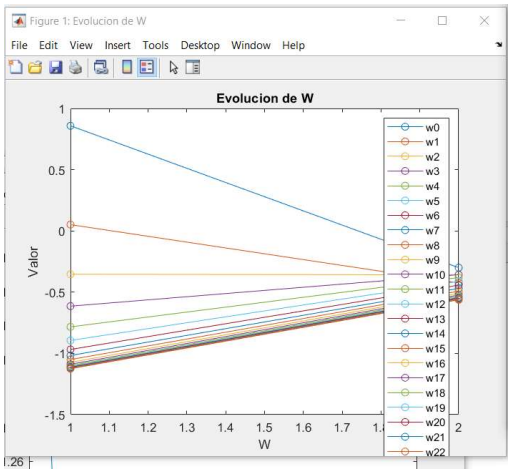
Gráfica 12. Fronteras de decisión 2 clases

En conclusión el experimento fue todo un éxito por lo cuál podemos decir que clasifica de manera correcta la compuerta AND, se puede comprobar la convergencia tanto en la gráfica 11 y en la 12.

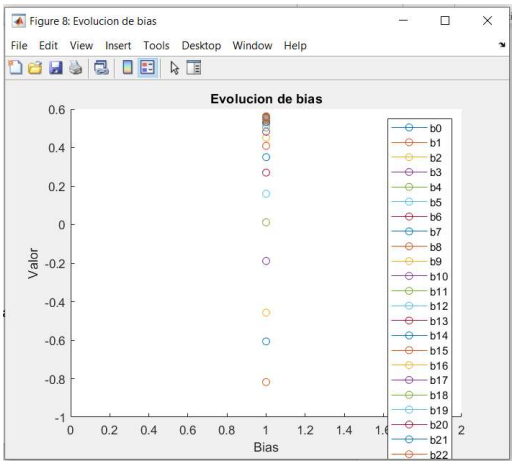
Experimento 3:

Este es el último experimento de nuestra donde tenemos un dataset que no es linealmente separable, se puede obviar que nuestra res ADALINE no va a converger correctamente debido a esta limitante por lo cuál se verá reflejado en nuestro entrenamiento.

Si observamos los valores de pesos y bias en este ejercicio observaremos que los convergen pero al comparar con el nivel de error que tiene se puede decir que no convergió la red.



Gráfica 13. Evolución pesos Clasificador XOR

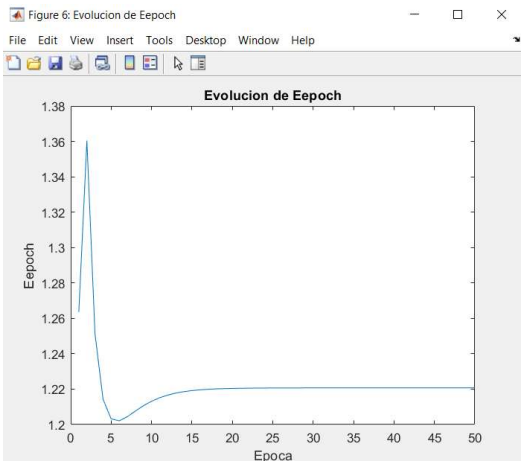


Gráfica 14. Evolución Bias clasificar XOR

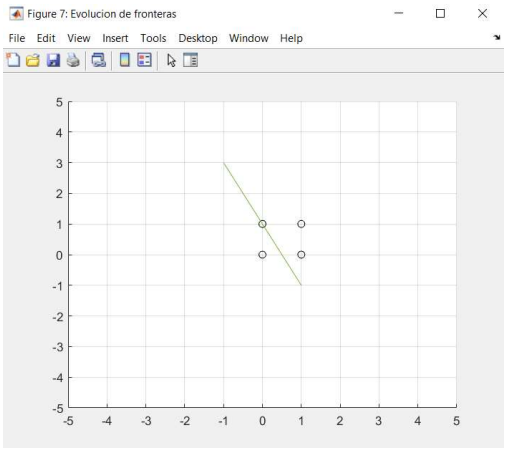
Como salida de nuestra terminal se muestran los valores de W, y de Eepoch, además de que va contando el número de época en el que nos encontramos.

```
>> Adaline_p4
Hola, Bienvenido a la red Adaline\n
Seleccione 0 Modo REGRESOR 1 modo CLASIFICADOR otro SALIR
1
Valor de Alpha:
   0.1800

Valor de Eepoch
  1.0000e-03

Selecciono la red en modo CLASIFICADOR
El numero de rasgos es: 2
El numero de clases ingresadas es: 2
El numero de neuronas necesarias es: 1
Ingrese el numero de MAX_EPOCH:50
Epoca número: 1
Epoca número: 2
Epoca número: 3
Epoca número: 4
Epoca número: 5
Epoca número: 6
Epoca número: 7
Epoca número: 8
Epoca número: 9
Epoca número: 10
Epoca número: 11
Epoca número: 12
Epoca número: 13
Epoca número: 14
Epoca número: 15
Epoca número: 16
Epoca número: 17
Epoca número: 18
Epoca número: 19
Epoca número: 20
Epoca número: 21
Epoca número: 22
Epoca número: 23
Epoca número: 24
Epoca número: 25
Epoca número: 26
Epoca número: 27
Epoca número: 28
Epoca número: 29
Epoca número: 30
Epoca número: 31
Epoca número: 32
Epoca número: 33
Epoca número: 34
Epoca número: 35
Epoca número: 36
Epoca número: 37
Epoca número: 38
Epoca número: 39
Epoca número: 40
Epoca número: 41
Epoca número: 42
Epoca número: 43
Epoca número: 44
Epoca número: 45
Epoca número: 46
Epoca número: 47
Epoca número: 48
Epoca número: 49
Epoca número: 50
Epoch
   1.2207

W
  -1.1250   -0.5625

Bias
   0.5625

La red no convergió vuelva a intentarlo
```

Veamos que la salida de la terminal no converge a un valor tan diminuto en Eepoch pero el error sigue manteniéndose relativamente pequeño, por lo cuál se mostrará que a pesar de esto logra hacer su objetivo sin problema



Gráfica 15. Evolución Eepoch en clasificador XOR



Gráfica 16. Fronteras de decisión XOR

Tanto el error y las fronteras de decisión no dan un resultado coherente debido a que el problema no es linealmente separable, la red hace lo que puede para poder clasificarlos de manera adecuada pero falla en su intento de hacerlo por lo mismo. Este experimento fracaso por lo mismo pero es por las capacidades de la red ADALINE

Discusión:

Como pudimos observar en nuestros experimentos la parte teórica vista en nuestras clases fue de gran utilidad para poderlas realizar además de que tuvimos una manera más visual de observar como una red como ADALINE que es para problemas lineales no puede ayudarnos mucho en problemas no lineales ya que los valores de pesos y bias oscilan. Esto para otro tipo de problemas nos será útil porque podremos identificar cuando nuestra red neuronal tiene un problema de bias(diseño) o varianza.

Ahora si retomamos lo visto la red Adaline comparada con el perceptrón tiene muchas más implementaciones y su regla de aprendizaje nos dio mejores resultados que el perceptrón de una sola capa, esto es importante porque también sabremos cuando usar una ante la otra.

Conclusiones:

Es notoria la diferencia con la red perceptrón a pesar de tener un gran parecido lo que nos lleva al pensamiento del peso de las funciones de transición ya que es lo único que cambian, sin embargo, eso hace que cambie totalmente la manera de resolver los problemas y la forma de programar los mismos. Es necesario entender el primero para tener una mayor facilidad de programar una red ADALINE ya que sirven como un gran complemento y de igual forma, muchas funciones que ya se habían utilizado en la red perceptrón son útiles también en esta práctica.

Bibliografía

Hagan, M. T. (s.f.). Neural Network Design. En M. T. Hagan, *Neural Network Design* (pág. 1010).

Pereira, U. T. (2000). *http://www.medicinaycomplejidad.org/pdf/redes/Competitivas.pdf*. Obtenido de medicinaycomplejidad.