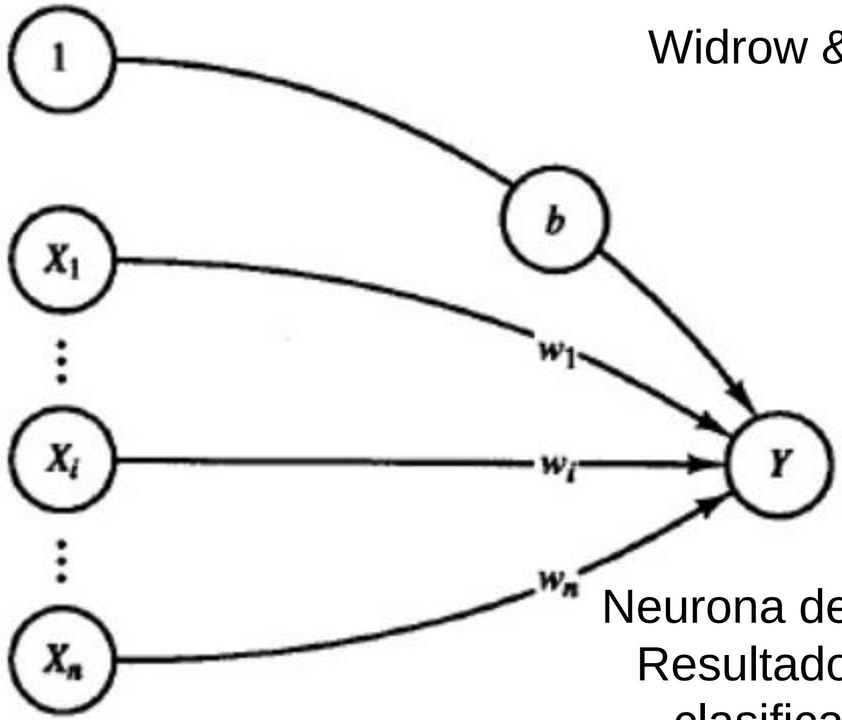
ADALINE
(ADaptive LInear NEuron)

Widrow & Hoff, 1960



$$y\_in = b + \sum_i x_i w_i$$

Durante el entrenamiento:

$$f(y\_in) = y\_in$$

Neurona de salida: Resultado de la clasificación

Capa de entrada: parámetros que se utilizan para la clasificación

La regla de aprendizaje se llama la regla delta o de Widrow-Hoff o de mínimo error cuadrático medio:

$$w_i(\text{nuevo}) = w_i(\text{anterior}) + \alpha(t - y\_in)x_i$$

t es el valor objetivo y α es la tasa de aprendizaje

ADALINE

- Después del entrenamiento se aplica una función de transferencia con umbral que se establece a 1 si la entrada neta es mayor o igual a cero y -1 en caso contrario.

- Cualquier problema separable linealmente (salidas +1 y -1) puede resolverse con un ADALINE.

- La regla delta minimiza el error cuadrático medio entre la activación y el valor objetivo.

Algoritmo de aprendizaje del ADALINE

Paso 0: Inicializar todos los pesos y sesgos (valores aleatorios pequeños)
Establecer la tasa de aprendizaje α.

Paso 1: Mientras que la condición de parada sea falsa, ejecutar pasos 2-6

Paso 2: Para cada par de entrenamiento (***s***:*t*) bipolar, ejecutar los pasos 3-5:

Paso 3: Establecer las activaciones a las neuronas de entrada

$$x_i = s_i \quad (i=1\ldots n)$$

Paso 4: Calcular la respuesta de la neurona de salida:

$$y\_in = b + \sum_i x_i w_i$$

Paso 5: Ajustar los pesos y el sesgo:

$$w_i(\text{nuevo}) = w_i(\text{anterior}) + \alpha(t - y\_in)x_i$$

$$b(\text{nuevo}) = b(\text{anterior}) + \alpha(t - y\_in)$$

Paso 6: Comprobar la condición de parada: si el cambio de peso más grande en el paso 2 es menor que una tolerancia especificada: parar; en caso contrario, continuar.

Tasa de aprendizaje en el ADALINE

$$w_i(\text{nuevo}) = w_i(\text{anterior}) + \alpha(t\text{-}y\_in)x_i$$

$$b(\text{nuevo}) = b(\text{anterior}) + \alpha(t\text{-}y\_in)$$

- Se suele tomar un valor pequeño para α (por ejemplo α=0.1).

- Si se toma un valor grande, el aprendizaje no converge, si se toma un valor muy pequeño el aprendizaje es lento.

- Para una única neurona de salida se suele tomar $0.1 \leq n\alpha \leq 1$ con n el número de neuronas de entrada.

Uso del ADALINE (fase de explotación)

Paso 0: Aplicar la regla delta de aprendizaje para establecer el valor de los pesos de las conexiones.

Paso 1: Para cada vector de entrada $\mathbf{x}$ a clasificar, ejecutar pasos 2-3

Paso 2: Establecer las activaciones a las neuronas de entrada $x_i = s_i \quad (i=1\ldots n)$

Paso 3: Calcular la respuesta de la neurona de salida:

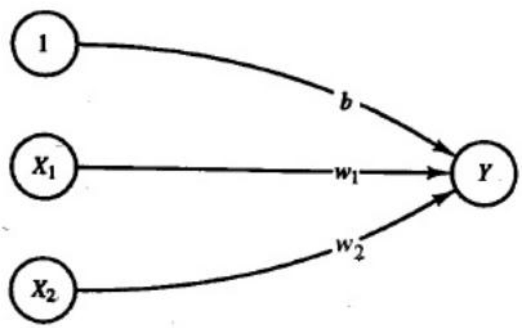
$$y\_in = b + \sum_i x_i w_i$$

$$y = f(y\_in) = \begin{cases} 1 & \text{si } y\_in \geq 0 \\ -1 & \text{si } y\_in < 0 \end{cases}$$

También se puede utilizar una función de transferencia binaria en vez de bipolar

Ejemplos de uso del ADALINE

ADALINE para la función AND con entradas binarias y objetivos bipolares



w_i(nuevo) = w_i(anterior) + $\alpha(t\text{-}y\_in)x_i$

b(nuevo) = b(anterior) + $\alpha(t\text{-}y\_in)$

$$y\_in = b + \sum_i x_i w_i$$

w_0=b

La regla delta encuentra los pesos que minimizan el error cuadrático medio:

$$E = \sum_{p=1}^{4} [x_1(p)w_1 + x_2(p)w_2 + w_0 - t(p)]^2$$

donde $x_1(p)w_1 + x_2(p)w_2 + w_0$

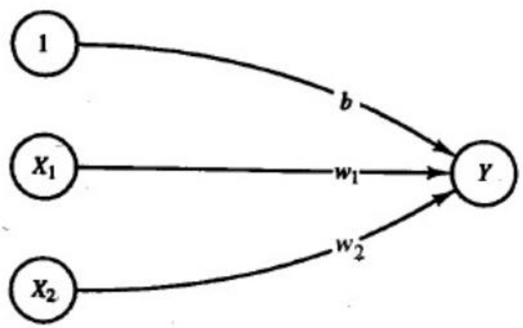
es la entrada neta a la neurona de salida para el patrón p y $t(p)$ es su salida objetivo para el patrón p

x_1	x_2	t
1	1	1
1	0	-1
0	1	-1
0	0	-1

Los pesos que minimizan el error son: $w_1=1$, $w_2=1$, $w_0=-3/2$, y por tanto la línea separadora es:

$$x_2 = -x_1 + \frac{3}{2}$$

ADALINE para la función AND con entradas y objetivos bipolares



w_i(nuevo) = w_i(anterior) + $\alpha(t\text{-}y\_in)x_i$

b(nuevo) = b(anterior) + $\alpha(t\text{-}y\_in)$

$$y\_in = b + \sum_i x_i w_i$$

w_0=b

La regla delta encuentra los pesos que minimizan el error cuadrático medio:

$$E = \sum_{p=1}^{4} [x_1(p)w_1 + x_2(p)w_2 + w_0 - t(p)]^2$$

x_1	x_2	t
1	1	1
1	-1	-1
-1	1	-1
-1	-1	-1

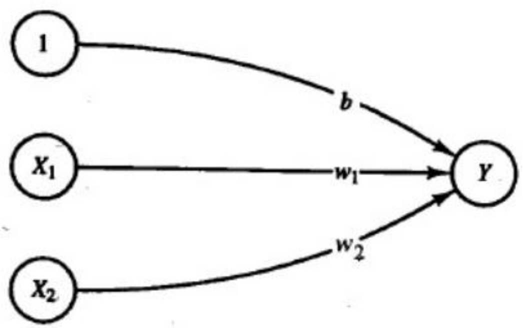
donde $x_1(p)w_1 + x_2(p)w_2 + w_0$

es la entrada neta a la neurona de salida para el patrón p y $t(p)$ es su salida objetivo para el patrón p

Los pesos que minimizan el error son: w_1=1/2, w_2=1/2, w_0=-1/2, y por tanto la línea separadora es:

$$x_2 = -x_1 + 1$$

ADALINE para la función AND NOT con entradas y objetivos bipolares



$w_i(\text{nuevo}) = w_i(\text{anterior}) + \alpha(t\text{-}y\_in)x_i$

$b(\text{nuevo}) = b(\text{anterior}) + \alpha(t\text{-}y\_in)$

$$y\_in = b + \sum_i x_i w_i$$

w_0=b

x_1	x_2	t
1	1	-1
1	-1	1
-1	1	-1
-1	-1	-1

La regla delta encuentra los pesos que minimizan el error cuadrático medio:

$$E = \sum_{p=1}^{4} [x_1(p)w_1 + x_2(p)w_2 + w_0 - t(p)]^2$$

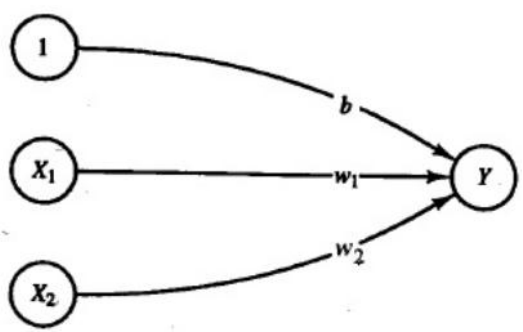
donde $x_1(p)w_1 + x_2(p)w_2 + w_0$

es la entrada neta a la neurona de salida para el patrón p y $t(p)$ es su salida objetivo para el patrón p

Los pesos que minimizan el error son: $w_1=1/2$, $w_2=-1/2$, $w_0=-1/2$, y por tanto la línea separadora es:

$$x_2 = x_1 - 1$$

ADALINE para la función OR con entradas y objetivos bipolares



$$w_i(\text{nuevo}) = w_i(\text{anterior}) + \alpha(t\text{-}y\_in)x_i$$

$$b(\text{nuevo}) = b(\text{anterior}) + \alpha(t\text{-}y\_in)$$

$$y\_in = b + \sum_i x_i w_i$$

w_0=b

x_1	x_2	t
1	1	1
1	-1	1
-1	1	1
-1	-1	-1

La regla delta encuentra los pesos que minimizan el error cuadrático medio:

$$E = \sum_{p=1}^{4} [x_1(p)w_1 + x_2(p)w_2 + w_0 - t(p)]^2$$

donde $x_1(p)w_1 + x_2(p)w_2 + w_0$

es la entrada neta a la neurona de salida para el patrón p y $t(p)$ es su salida objetivo para el patrón p

Los pesos que minimizan el error son: w_1=1/2, w_2=1/2, w_0=1/2, y por tanto la línea separadora es:

$$x_2 = -x_1 - 1$$

Regla Delta para una única neurona de salida

- La regla delta cambia los pesos para minimizar la diferencia de la entrada a la neurona de salida, y_in, y el objetivo t.

- El objetivo de este proceso es minimizar el error sobre todos los patrones de entrenamiento. Esto se hace para cada patrón.

- Para distinguir entre el índice del peso que se actualiza y el índice del sumatorio de las entradas, llamamos I al índice del peso que ese está actualizando. La regla delta especifica que el cambio de este peso es:

$$\Delta w_I = \alpha(t - y\_in)x_I$$

Regla Delta para una única neurona de salida

- El error cuadrático para un patrón particular es:

$$E = (t - y\_in)^2.$$

donde E es una función de todos los pesos w_i, puesto que $y\_in = \sum_i x_i w_i$

- El gradiente de E (un vector que consiste de las derivadas parciales de E con respecto a cada peso) proporciona la dirección de más rápido crecimiento de este error,

$$-\frac{\partial E}{\partial w_I} = 2(t - y\_in)\frac{\partial y\_in}{\partial w_I} = 2(t - y\_in)x_I$$

- Por tanto, el error se reduce más rápido ajustando los pesos mediante la regla delta:

$$\Delta w_I = \alpha(t - y\_in)x_I$$

Regla Delta para varias neuronas de salida

- El error cuadrático para un patrón particular es: $E = \sum_{j=1}^{m} (t_j - y\_in_j)^2$

donde E es una función de todos los pesos w_i, puesto que $y\_in_j = \sum_i x_i w_{ij}$

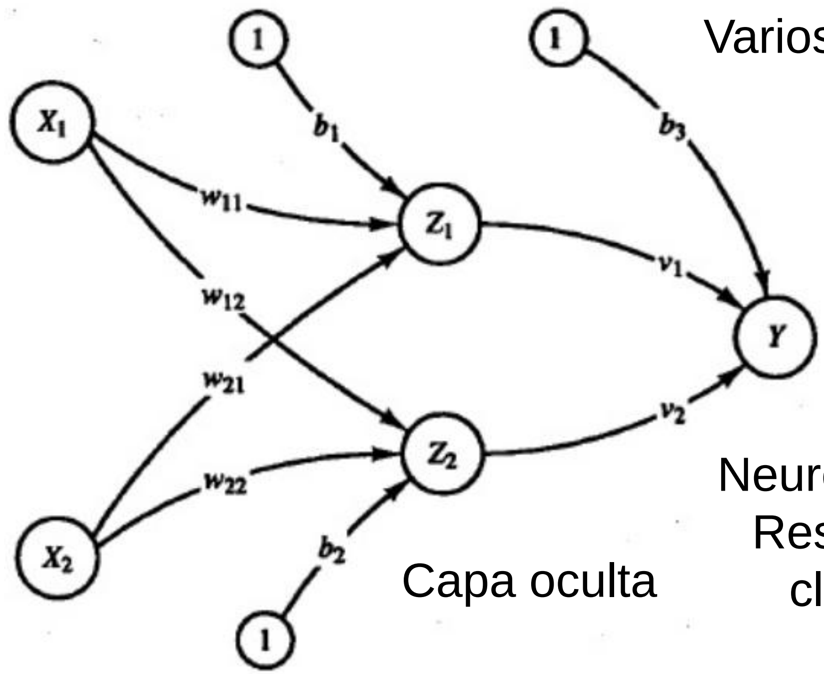
- El gradiente de E (un vector que consiste de las derivadas parciales de E con respecto a cada peso) proporciona la dirección de más rápido crecimiento de este error,

$$-\frac{\partial E}{\partial w_{IJ}} = 2(t_J - y\_in_J)\frac{\partial y\_in_J}{\partial w_{IJ}} = 2(t_J - y\_in_J)x_I$$

- Por tanto, el error se reduce más rápido ajustando los pesos mediante la regla delta:

$$\Delta w_{IJ} = \alpha(t_J - y\_in_J)x_I$$

MADALINE
(Many ADaptive LInear NEurons)



Capa oculta

Varios ADALINEs dispuestos en una red multicapa.

$$z\_in_j = b_j + \sum_i x_i w_{ij}$$

$$z_j = f(z\_in_j) = \begin{cases} 1 & \text{si } z\_in_j \geq 0 \\ -1 & \text{si } z\_in_j < 0 \end{cases}$$

Neurona de salida: Resultado de la clasificación

$$y\_in = b + \sum_i z_i w_i$$

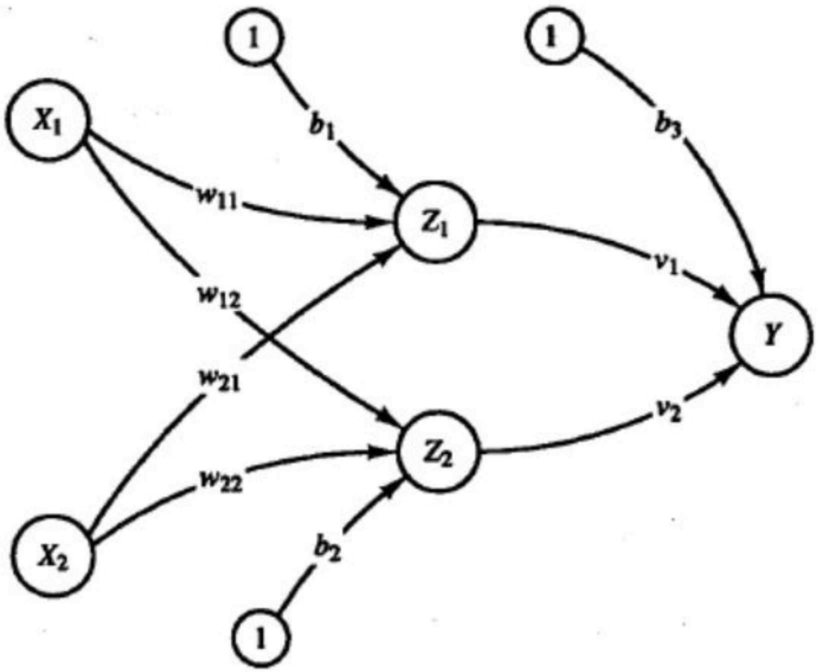
$$y = f(y\_in) = \begin{cases} 1 & \text{si } y\_in \geq 0 \\ -1 & \text{si } y\_in < 0 \end{cases}$$

Capa de entrada: parámetros que se utilizan para la clasificación

- La salida y es una función no lineal del vector de entrada (x_1, x_2).

- El uso de la capa oculta, proporciona propiedades computacionales que no tienen las redes de una sola capa.

MADALINE
Regla de aprendizaje MRI



- MRI = MADALINE Rule I

- Los pesos v_1 y v_2 y el bias b_3 a la neurona Y se determinan de forma que la respuesta de Y sea 1 si Z_1 o Z_2 (o las dos) envían un 1, y -1 si Z_1 y Z_2 envían la señal -1. Es decir, La neurona Y hace un OR de las señales enviadas por Z_1 y Z_2.

- Los pesos de las conexiones a Y son por tanto: $v_1=v_2=b_3=1/2$

Regla de aprendizaje MRI (1/2)

Paso 0: Inicializar todos los pesos y sesgos (valores aleatorios pequeños), salvo para v_1=v_2=b_3=1/2 (en este ejemplo)
Establecer la tasa de aprendizaje α (un valor pequeño).

Paso 1: Mientras que la condición de parada sea falsa, ejecutar pasos 2-8

Paso 2: Para cada par de entrenamiento (***s***:***t***) bipolar, ejecutar los pasos 3-7:

Paso 3: Establecer las activaciones a las neuronas de entrada

$x_i = s_i \quad (i=1\ldots n)$

Paso 4: Calcular la respuesta de la neurona de salida:

$$z\_in_j = b_j + \sum_i x_i w_{ij}$$

Paso 5: Determinar la salida para cada neurona de la capa oculta

$$z_j = f(z\_in_j)$$

Paso 6: Determinar la salida de la red

$$y\_in = b_3 + \sum_i z_i w_i$$

$$y = f(y\_in)$$

Regla de aprendizaje MRI (2/2)

Paso 7: Determinar el error y actualizar los pesos:

- Si $t=y$ no se actualizan los pesos
- Si $t=1$, se actualizan los pesos en Z_J (la neurona que su entrada es más próxima a 0):

$$w_{iJ}(\text{nuevo}) = w_{iJ}(\text{anterior}) + \alpha(1\text{-}z\_in_J)x_i$$
$$b_J(\text{nuevo}) = b_J(\text{anterior}) + \alpha(1\text{-}z\_in_J)$$

- Si $t=$-1, se actualizan los pesos en todas las unidades Z_k que tienen una entrada positiva:

$$w_{ik}(\text{nuevo}) = w_{ik}(\text{anterior}) + \alpha(-1\text{-}z\_in_k)x_i$$
$$b_k(\text{nuevo}) = b_k(\text{anterior}) + \alpha(-1\text{-}z\_in_k)$$

Paso 8: Comprobar la condición de parada. Si los pesos han parado de cambiar o alcanzado un nivel aceptable, o si se a alcanzado un número de iteraciones máximas, entonces se para.

NOTA: el paso 7 tiene como objetivo actualizar los pesos si se produce un error y además de forma que sea más probable que la red produzca la respuesta deseada.

Observaciones sobre MRI

- Se pueden construir MADALINEs en los que la neurona de salida realiza otra función lógica (por ejemplo AND). La regla de actualización de pesos se modificaría entonces para producir este tipo de salida.

- Para permitir entrenamiento en los pesos de todas las capas se propuso una versión de MRI que se denominó MRII.

Regla de aprendizaje MRII

Paso 0: Inicializar todos los pesos
Establecer la tasa de aprendizaje α

Paso 1: Mientras que la condición de parada sea falsa, ejecutar pasos 2-8

Paso 2: Para cada par de entrenamiento (***s***:***t***) bipolar, ejecutar los pasos 3-7:
Pasos 3-6 Igual que MRI

Paso 7: Determinar el error y actualizar los pesos si es necesario:
Si $t \neq y$ ejecutar pasos 7a-b para cada neurona de la capa oculta cuya entrada sea lo suficientemente cercana a 0 (por ejemplo entre -0.25 y 0.25). Empezar con la neurona cuya entrada es más cercana a 0, luego con la siguiente, etc.

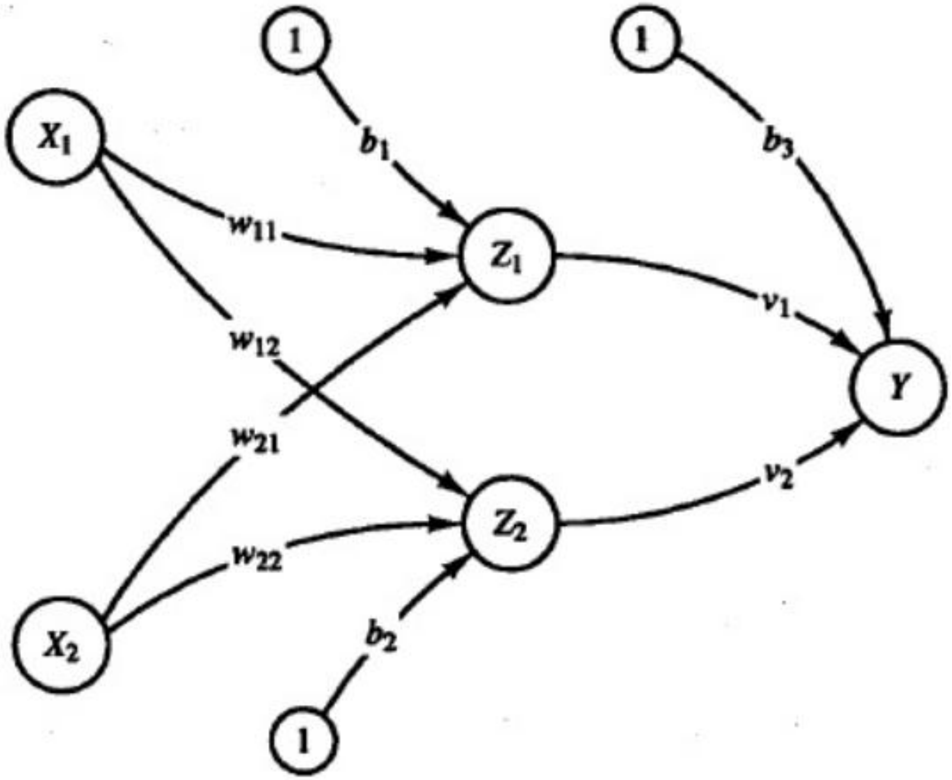
Paso 7a: cambiar la salida de la unidad: de +1 a -1 o al revés.

Paso 7b: recalcular la respuesta de la red. Si el error se reduce, ajustar los pesos de esta neurona (usar su nueva salida como objetivo y aplicar la regla Delta).

Paso 8: Comprobar la condición de parada. Si los pesos han parado de cambiar o alcanzado un nivel aceptable, o si se a alcanzado un número de iteraciones máximas, entonces se para.

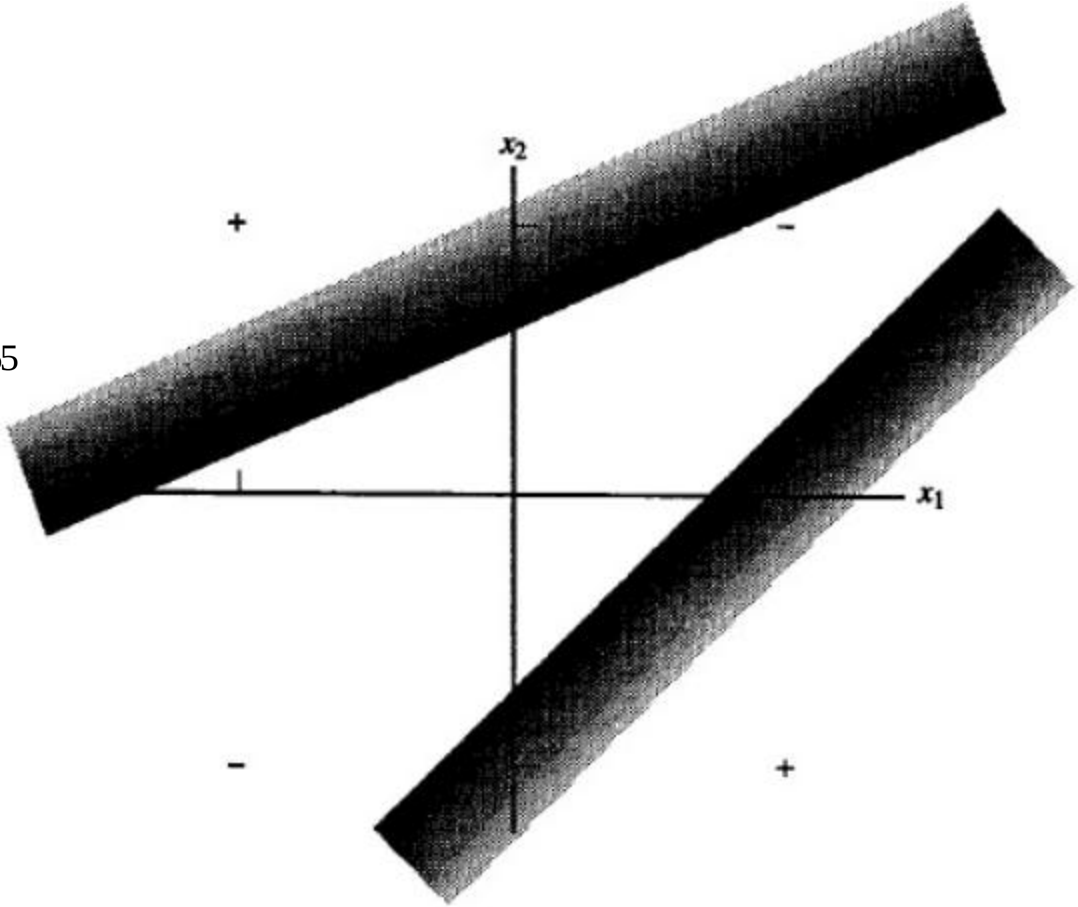
Ejemplo: MADALINE para XOR con MRI

x_1	x_2	t
1	1	-1
1	-1	1
-1	1	1
-1	-1	-1



Pesos iniciales a Z_1: w_{11}=0.5, w_{21}=0.2, b_1=0.3
Pesos iniciales a Z_2: w_{12}=0.1, w_{22}=0.2, b_2=0.15
Pesos iniciales a Y: v_1=0.5, v_2=0.5, b_3=0.5

Ejemplo: MADALINE para XOR con MRI



$$x_2 = 0.48\,x_1 + 0.65$$

$$x_2 = 0.96\,x_1 - 0.82$$

OBSERVACIONES

- El número de capas ocultas se puede estimar en casos en los que los patrones se encuentran en regiones que se pueden acotar por un número finito de líneas, planos o hiperplanos.

- Una red con una capa oculta con p neuronas puede aprender una respuesta acotada por p líneas rectas.

- Si las respuestas a una misma categoría ocurren en más de una región disjunta del espacio de entradas, se puede utilizar una capa oculta adicional para combinar estas regiones y que el entrenamiento sea más fácil.