REDES NEURONALES

2021

Clase 16 Parte 2

Facultad de Matemática, Astronomía, Física y Computación
Universidad Nacional de Córdoba

Jueves 7 de octubre 2021

http://www.famaf.unc.edu.ar/~ftamarit/redes2021

https://www.famaf.unc.edu.ar/course/view.php?id=798

Si existe una o infinitas rectas que brindan esta deseada separación diremos que el problema es

LINEALMENTE SEPARABLE.

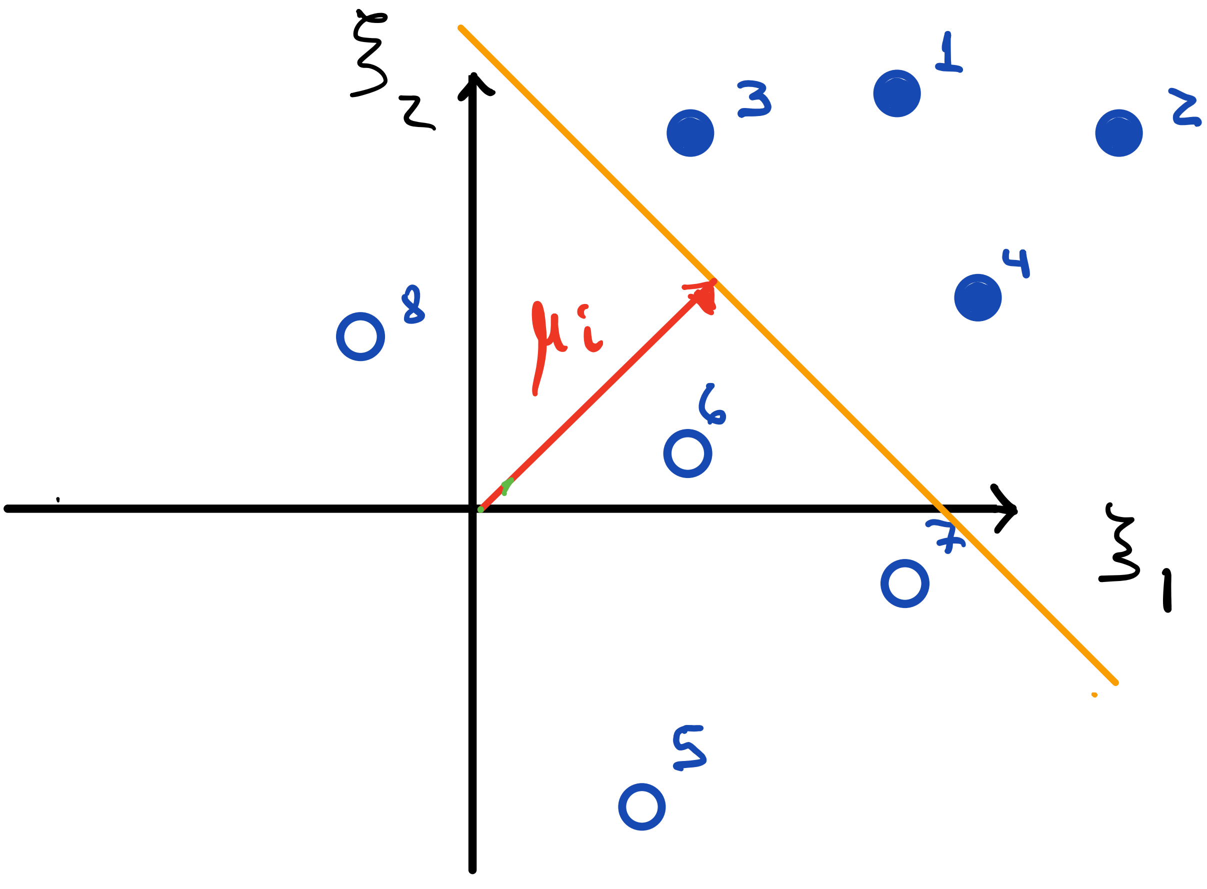
En el caso más general, **separabilidad lineal** significa que en $\mathbb{R}^N$ debe existir un híper plano $\mathbb{R}^{N-1}$ que separe a los vectores del conjunto de entrenamiento.

¿Qué sucede si ahora incluimos el umbral de la neurona de salida?

$$O = \text{signo}\left(\sum_{k=1} w_k \xi_k - w_0\right)$$

$$O = \text{signo}\left(\vec{w} \cdot \vec{\xi} - w_0\right)$$

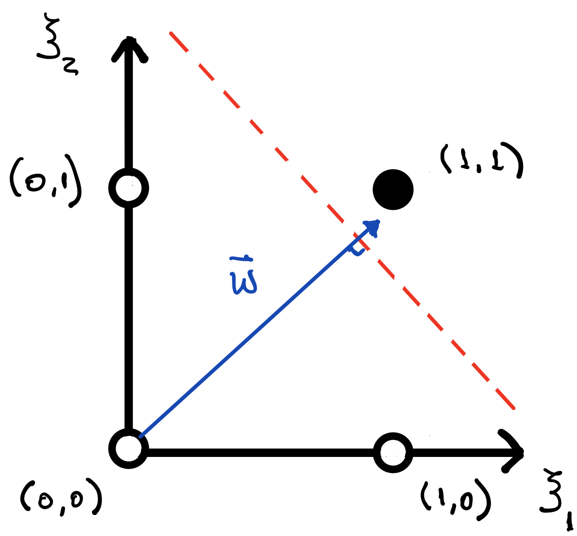
Ahora el híper-plano de dimensión *N-1* debe pasar a una distancia *W* del origen

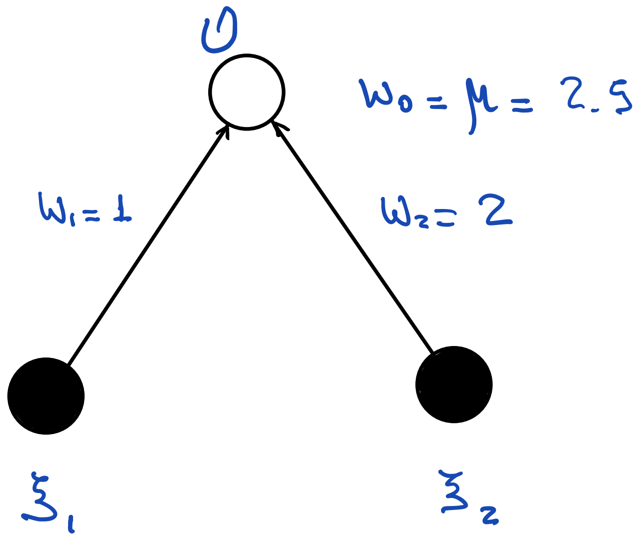


Miremos la función de la lógica booleana **AND**. Como tenemos dos variables de entrada (features) binarias (O para indicar el valor “falso” y 1 para indicar el valor “verdadero” T), tenemos solo cuatro entradas posibles.

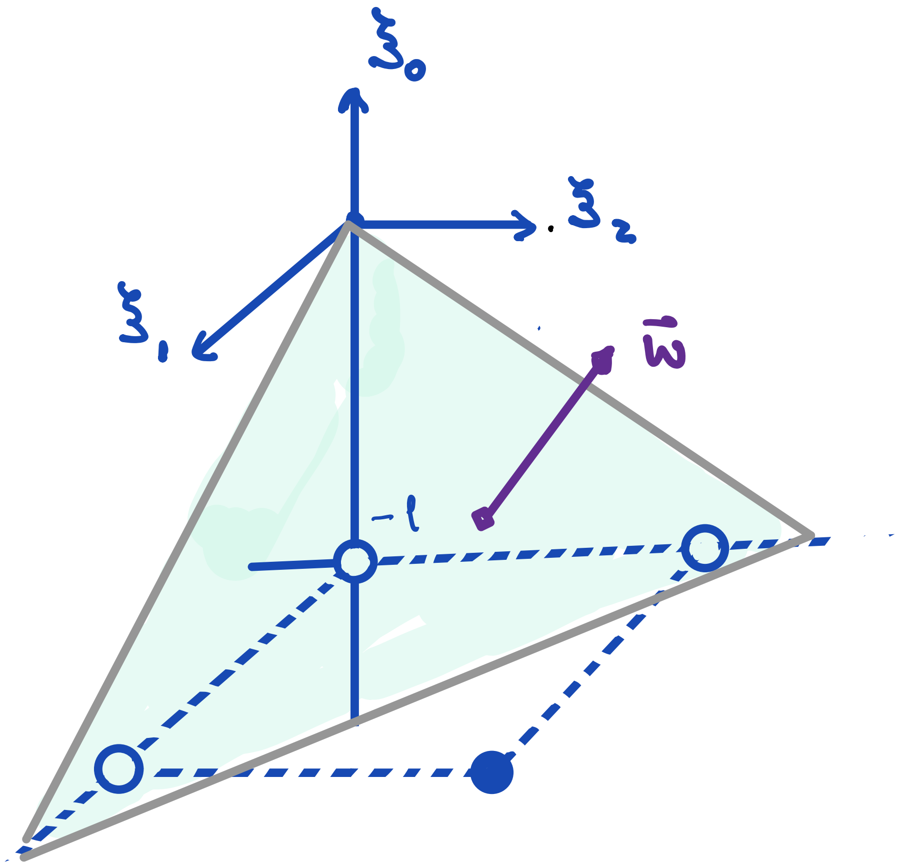
Recuerden que en la salida tenemos -1 para falso y +1 para verdadero.

ξ_1	$\wedge$	ξ_2	ζ
0		**0**	**-1**
0		**1**	**-1**
1		**0**	**-1**
1		**1**	**+1**





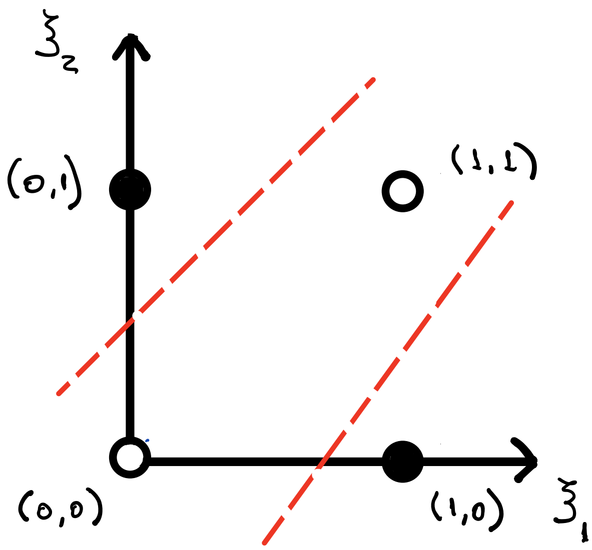
O
$w_0 = \mu = 2.5$
$w_1 = 1$
$w_2 = 2$
ξ_1
ξ_2



ξ_0
ξ_2
ξ_1
$\vec{w}$
-1

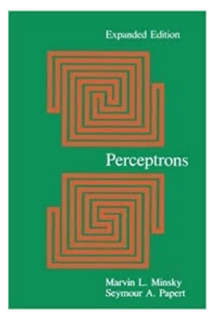
Miremos ahora la función lógica **XOR**

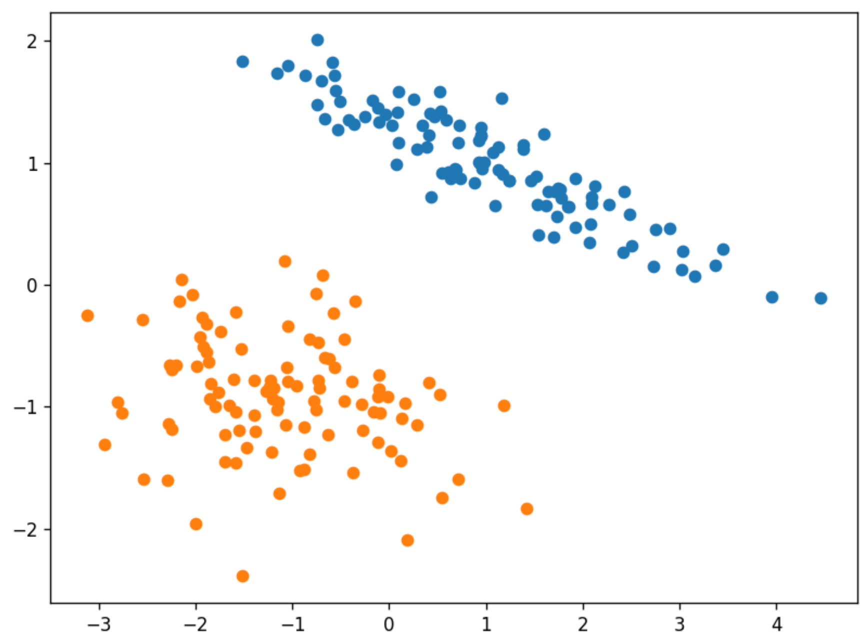
ξ_1	ξ_2	ζ
0	0	-1
0	1	+1
1	0	+1
1	1	-1



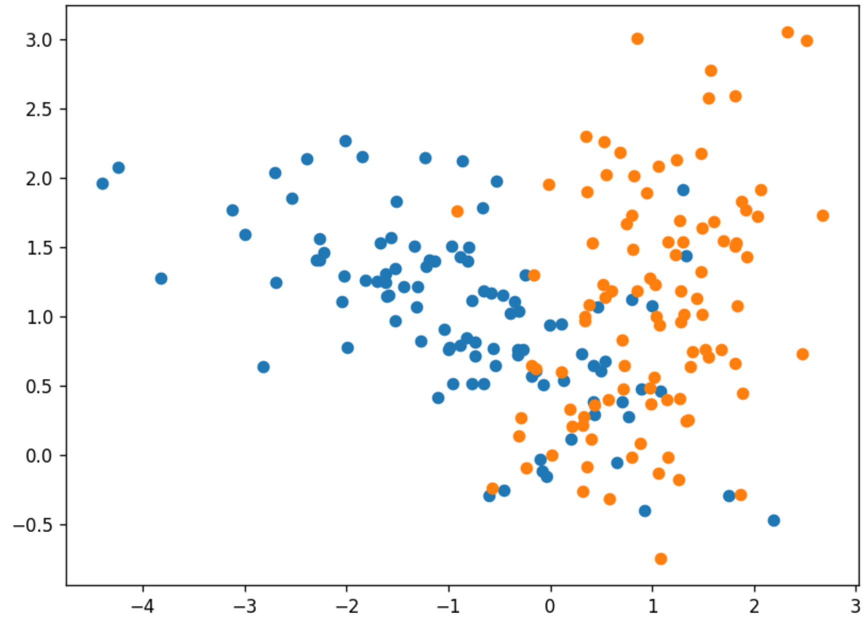
¡LA OPERACIÓN LÓGICA **XOR** NO ES LINEALMENTE SEPARABLE!

Marvin Minsky y Seymour Papert, “PERCEPTRONS”, 1969





2
1
0
−1
−2
−3
−2
−1
0
1
2
3
4



3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−4
−3
−2
−1
0
1
2
3

EL PERCEPTRON SIMPLE COMO CLASIFICADOR II

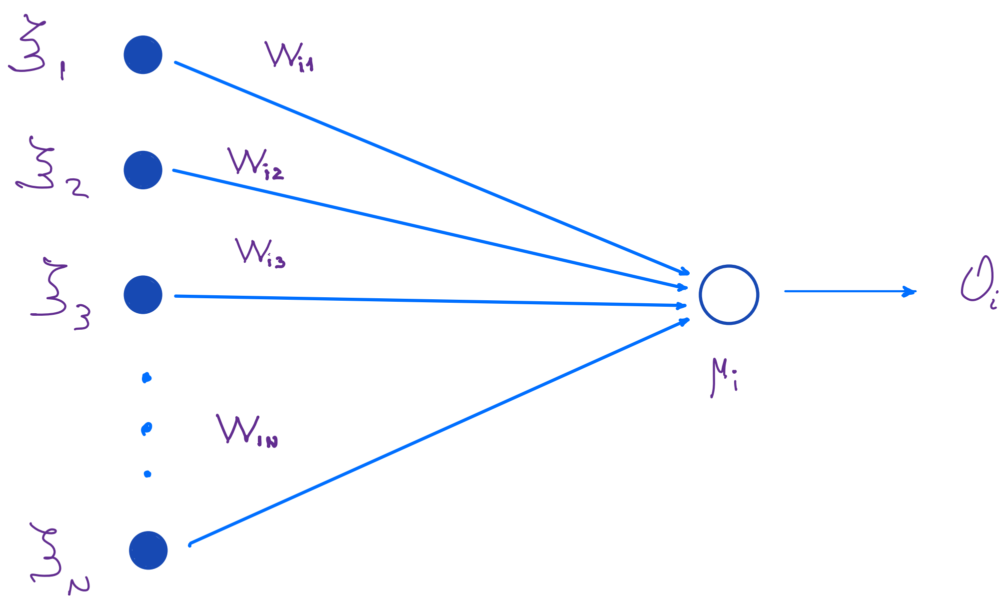
Hasta aquí vimos bajo cuáles condiciones existe una solución al problema de entrenar un perceptrón simple para la clasificación binaria, un problema de muchísima utilidad. Sin embargo, necesitamos una regla para obtener el vector $\bar{W}$ y el umbral μ (recordemos que podemos considerar al umbral como una sinápsis extra).

Vimos que el problema tiene que ser **linealmente separable**. Esta es una propiedad del conjunto de entrenamiento y no de algún algoritmo,

Si en nuestro problema tenemos un número grande de entradas ($N>>1$) y un número grande de entradas etiquetadas ($p >> 1$) en el conjunto de entrenamiento, el,problema de entrenar el,perceptron se torna muy difícil y no podemos recurrir más a una ayuda gráfica.

A continuación introduciremos nuestro primer logaritmo de

REGLA DE APRENDIZAJE DEL PERCEPTRON



Supongamos que tenemos ***p*** entradas correctamente etiquetadas (a las cuales alguien le asignó la salida correcta)

$$\vec{\xi}^{\mu} \longrightarrow \zeta^{\mu} \qquad \mu = 1, 2, \dots, p$$

Finalmente supongamos que el problema es lineamente separable

Ahora nos preguntamos:

¿CÓMO ENCONTAMOS UN $\vec{W}$ ADECUADO?

Ante la falta de otro criterio elegimos inicialmente los acoplamientos $\vec{W}$ de forma aleatoria. Por ejemplo, podemos elegir los W_i independientes entre sí, con una distribución normal de media cero y varianza σ.

$$P(\bar{W}) = \prod_{i=1}^{N} p(w_i)$$

$$p(w_i) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{w_i^2}{2\sigma^2}}$$

A partir de estos valores iniciales aleatorios de las componentes del vector $\vec{W}$ le presentamos secuencialmente al perceptrón cada uno de los elemntosnetiquetados del conjunto de entrenamiento, uno tras otro.

Sea $\mu=1$.

Repetimos, hasta que $\mu=p$

Presentamos $\vec{\xi}^{\mu}$

Evalúo la salida $O^{\mu} = \text{signo}\,(\vec{\xi}^{\mu}\cdot\vec{W})$

Si $O^{\mu}=\xi^{\mu}$ entonces no hacemos nada

pero si $O^{\mu}=-\xi^{\mu}$ entonces cambiamos levemente $\vec{W}$

Hacemos $\mu=\mu+1$

Volvemos a “Repetimos”

FIN

Supongamos $\mu=1$ y calculemos la salida:

$$O^{\mu} = g(h^{\mu}) = g\left(\sum_{k=1}^{N} W_k\,\xi_k^{\mu}\right) = \text{signo}\,(\vec{W}\cdot\vec{\xi}^{\mu})$$

Si $\quad O^\mu = \zeta^\mu \quad$ no cambiamos nada, pero si

$O^\mu = -\zeta^\mu \quad$ cambiando un poco W

$$\vec{W}^{\,nuevo} = \vec{W}^{\,anterior} + \Delta\vec{W}$$

O, en término de las componentes:

$$W_k^{nuevo} = W_k^{anterior} + \Delta W_k \qquad k = 1, 2, \ldots, N$$

donde

$$\Delta W_k^\mu = \begin{cases} 2\eta\, \zeta^\mu \xi_k^\mu & si \quad O^\mu = -\zeta^\mu \\ 0 & si \quad O^\mu = \zeta^\mu \end{cases}$$

Esto se puede escribir como:

$$\begin{aligned} \Delta W_k^\mu &= \eta\,(1 - \zeta^\mu O^\mu)\, \zeta^\mu \xi_k^\mu \\ &= \eta\,(\zeta^\mu - O^\mu)\, \xi_k^\mu \end{aligned}$$

o para todo el vector:

$$\Delta\vec{W} = \eta\,(\zeta^\mu - O^\mu)\, \overbrace{\overline{\xi}^\mu \xi^\mu}^{\bar{X}^\mu}$$

La regla establece que si la red clasifica mal el ejemplo μ del conjunto de entrenamiento, el vector $\vec{W}$ debe desplazarse en la dirección de $\zeta^\mu \vec{\xi}^\mu$.

- Si $\zeta^\mu = 1$ y se equivocó, $\overline{W}$ se desplaza un poco hacia $\bar{\xi}^\mu$.

$$\Delta \bar{w} = 2\eta\, \bar{\xi}^\mu$$

- Si $\zeta^\mu = -1$ y se equivocó, $\overline{W}$ se desplaza un poco hacia $-\bar{\xi}^\mu$.

$$\Delta \bar{w} = -2\eta\, \bar{\xi}^\mu$$

- Si no se equivocó, $\overline{W}$ no cambia.

$$\Delta \bar{w} = \bar{0}$$

El parámetro η se llama razón de aprendizaje y hoy lo presentamos para que nos acompañe para siempre.

η regula cuán bruscos serán los cambios de $\bar{W}$ durante el proceso de de aprendizaje del conjunto de entrenamiento. Veremos que es un parámetro **MUY DELICADO** y que nos dará mucho trabajo.

Como es un parámetro que tendrá el mismo valor en todo el aprendizaje se lo denomina híper parámetro y deberemos ajustarlo con delicadeza (fine tuning).

Nota: en IA se denomina "*parámetros*" a las sinapsis y umbrales, en analogía con el problema de regresión por cuadrados mínimos.

Para que el algoritmo haga bien su tarea con el ejemplo del conjunto de entrenamiento debe cumplirse que:

$$O^\mu = \text{signo}(\bar{w} . \bar{\xi}^\mu) = \zeta^\mu$$

o

$$\zeta^\mu \, \text{signo}(\bar{w} . \bar{\xi}^\mu) = \zeta^\mu \zeta^\mu = 1$$

$$\text{signo}(\zeta^\mu (\bar{w} . \bar{\xi}^\mu)) = 1$$

$$\zeta^\mu (\bar{w} . \bar{\xi}^\mu) = +1$$

Con pedir esto alcanzaría, pero podemos ser más exigentes con la estabilidad (lo cual redundará en un mejor funcionamiento de la red con ejemplos nuevos):

$$\zeta^\mu (\bar{w} . \bar{\xi}^\mu) > N x$$

$$\boxed{\Delta w_k = \eta\, \Theta\left(N\kappa - \zeta^\mu h^\mu\right) \zeta^\mu \xi_k^\mu}$$

donde

$$\Theta(z) = \begin{cases} 1 & \text{si} \quad z \geq 0 \\ 0 & \text{si} \quad z < 0 \end{cases}$$

Esta regla se llama **REGLA DE APRENDIZAJE DEL PERCEPTRÓN** y se debe a *Frank Rosenblatt* quien la desarrolló en 1962.

Se puede demostrar (la demostración está en el libro) que si el problema es linealmente separable, la regla converge a una solución $\tilde{W}$ en un número finito de pasos.

Es útil cambiar a una representación en la cual las variables son:

$$\bar{X}^\mu = \zeta^\mu \bar{\xi}^\mu$$

Noten que tanto las entradas como las etiquetas de los elementos del conjunto de entrenamiento son fijos y nosotros no podemos cambiarlos. Solo cambiamos $\bar{W}$.

Ahora la regla del perceptrón es:

$$\Delta \vec{w} = \eta\, \Theta\left(N\kappa - \zeta^{\mu}\left(\vec{w} \cdot \vec{\xi}^{\mu}\right)\right) \zeta^{\mu} \vec{\xi}^{\mu}$$

$$= \eta\, \Theta\left(N\kappa - \vec{w} \cdot \left(\zeta^{\mu} \vec{\xi}^{\mu}\right)\right) \zeta^{\mu} \vec{\xi}^{\mu}$$

$$= \eta\, \Theta\left(N\kappa - \vec{w} \cdot \vec{x}^{\mu}\right) \vec{x}^{\mu}$$