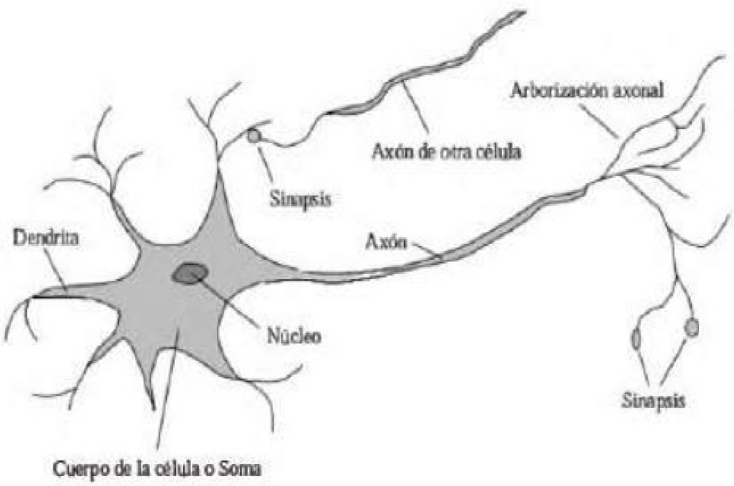
SISTEMAS INTELIGENTES ARTIFICIALES

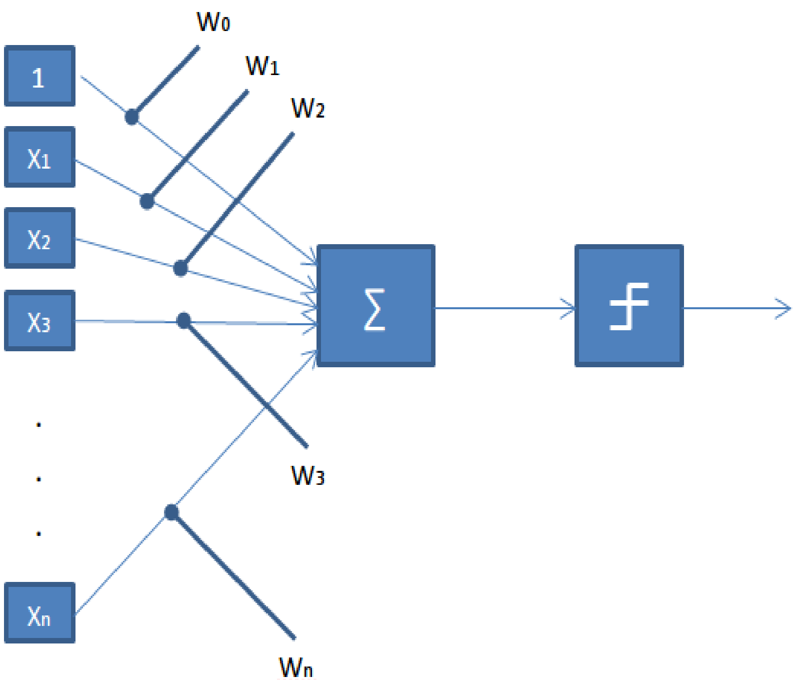
Prof.: Esp. Ing. Agustín Fernandez

MECANISMOS DE REDES NEURONALES: Perceptrón

Neurona bilógica

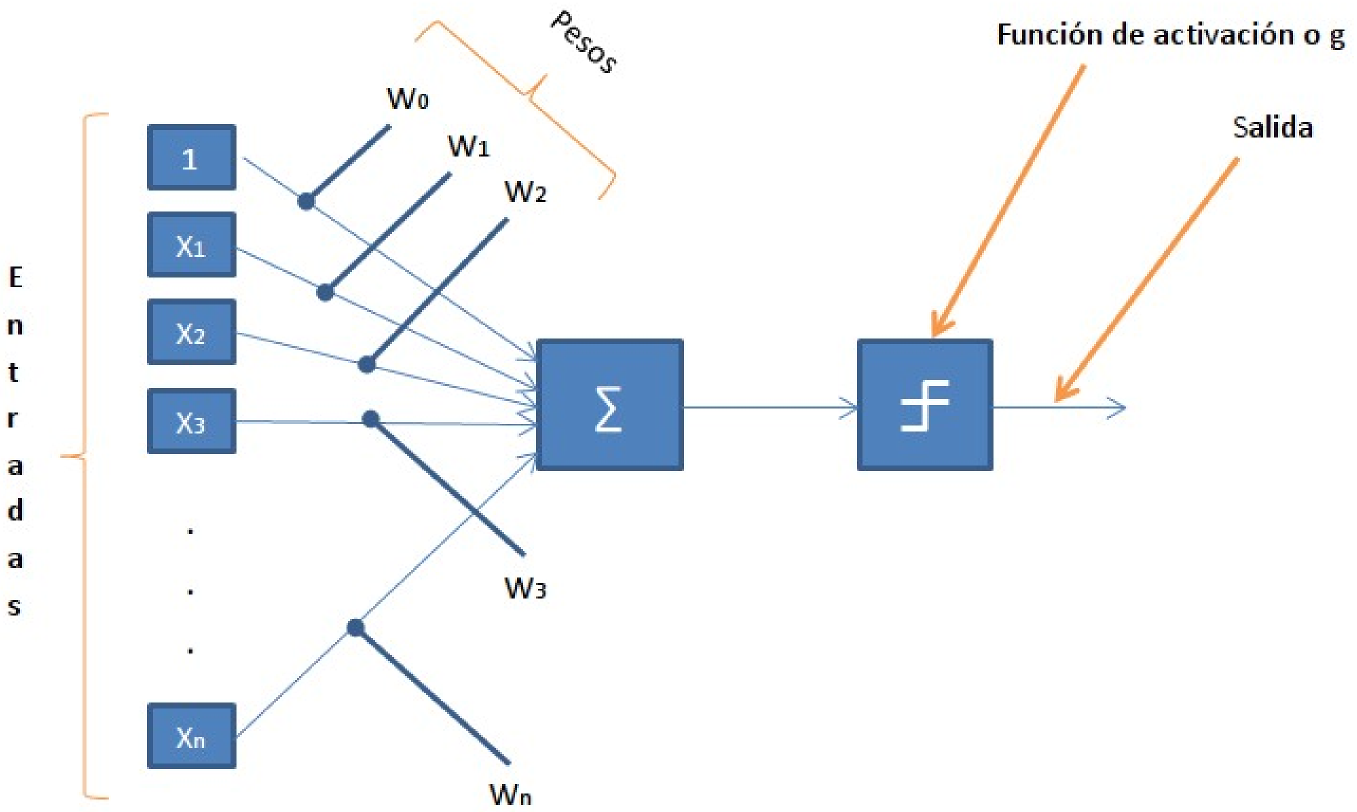


Neurona artificial



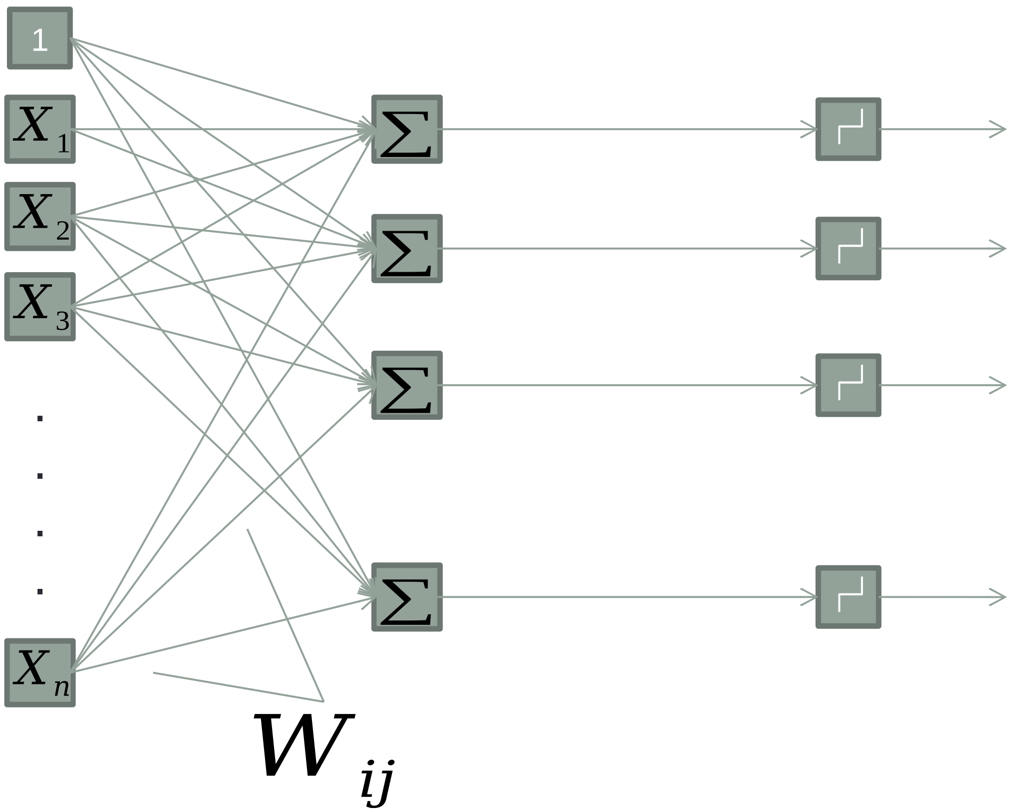
MECANISMOS DE REDES NEURONALES: Perceptrón

- **<u>Perceptrón</u>**:



MECANISMOS DE REDES NEURONALES: Perceptrón

- **Perceptrón con muchas entradas y salidas**:

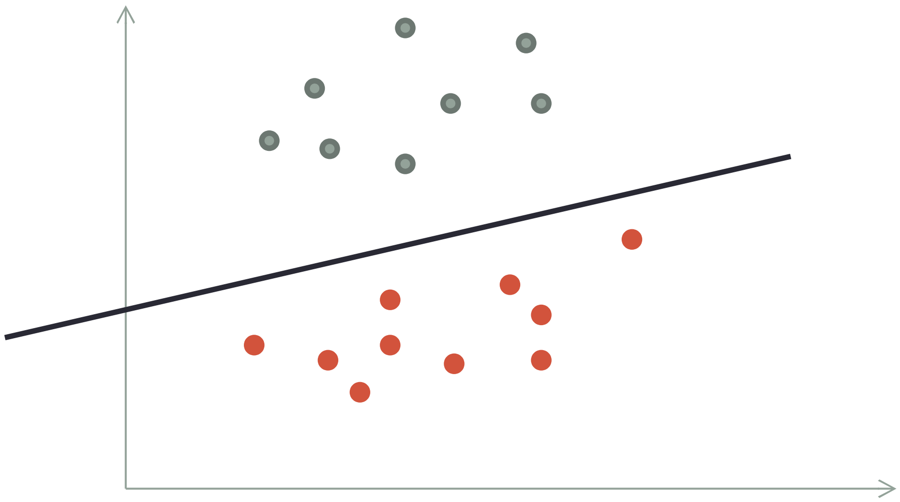


MECANISMOS DE REDES NEURONALES: Perceptrón

- Perceptrón:
- Inventado por Rosenblatt en 1962.
- Fue uno de los primeros modelos neuronales.
- Imita a una neurona tomando la suma ponderada de sus entradas y enviando a la salida un uno si la suma es más grande que algún valor umbral ajustable (si ocurre de otro modo devuelve cero).
- Los perceptrones poseen conexiones unidireccionales.
- Ideal para problemas linealmente separables.

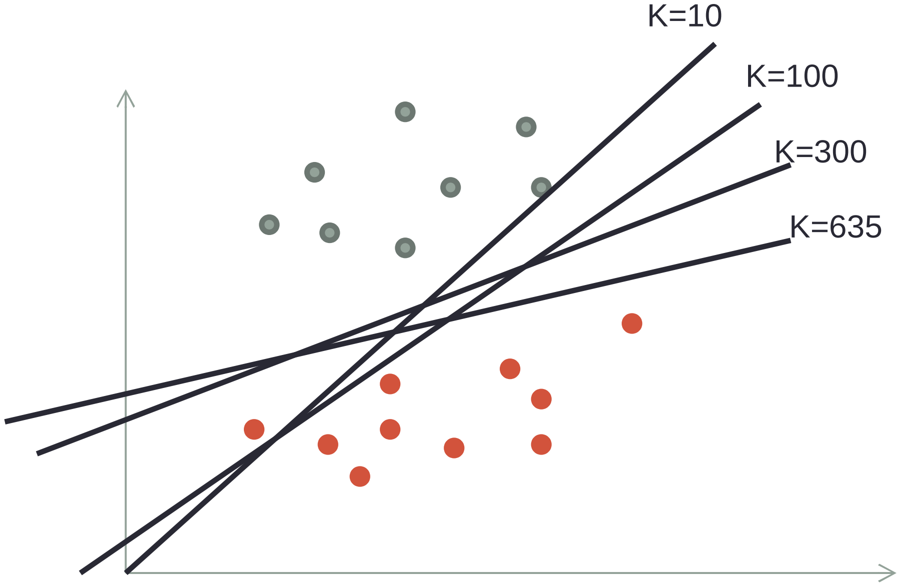
MECANISMOS DE REDES NEURONALES: Perceptrón

- <u>Perceptrón:</u>
- Un problema de clasificación de patrones linealmente separables:



MECANISMOS DE REDES NEURONALES: Perceptrón

- <u>Perceptrón:</u>
- El mismo Perceptrón aprendiendo a solucionar el problema de clasificación:



K	W_0	W_1	W_2
10	0,41	-0,17	0,14
100	0,22	-0,14	0,11
300	-0,10	-0,08	0,07
635	-0,49	-0,10	0,14

MECANISMOS DE REDES NEURONALES: Perceptrón

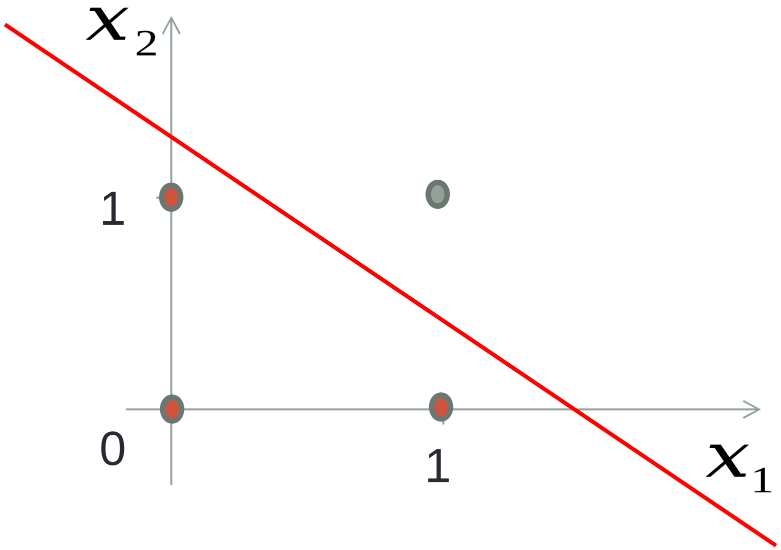
- <u>Modo de cálculo del Perceptrón:</u>
- Función suma ponderada:
- Función salida: $o(x) = \begin{cases} 0 \ si \ g(x) < 0 \\ 1 \ si \ g(x) > 0 \end{cases}$
- Si se obtiene la ecuación de la recta:
- $g(x) = w_o + w_1.x_1 + w_2.x_2$
- $x_2 = -\frac{w_1}{w_2}.x_1 - \frac{w_0}{w_2}$ Recta / superficie de decisión

MECANISMOS DE REDES NEURONALES: Perceptrón

- Modo de calculo del Perceptrón:
- $x_2 = -\frac{w_1}{w_2} . x_1 - \frac{w_0}{w_2}$ Recta / superficie de decisión

- Por ejemplo para una compuerta AND:

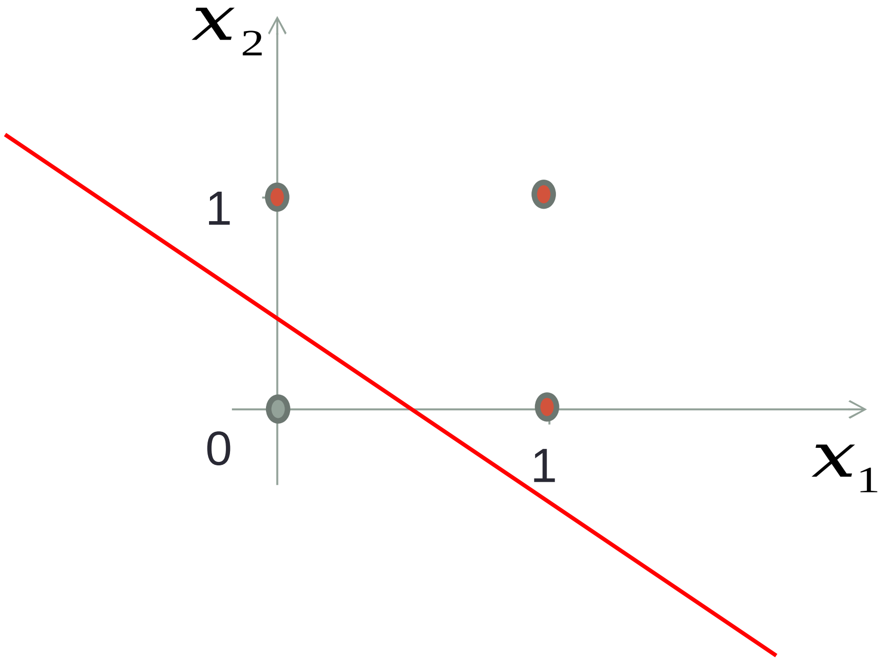
X_1	X_2	AND
0	0	0
0	1	0
1	0	0
1	1	1



MECANISMOS DE REDES NEURONALES: Perceptrón

- <u>Modo de calculo del Perceptrón:</u>
- Por ejemplo para una compuerta OR:

X_1	X_2	OR
0	0	0
0	1	1
1	0	1
1	1	1



MECANISMOS DE REDES NEURONALES: Perceptrón

- Modo de calculo del Perceptrón:
- Ejemplo de calculo para el aprendizaje:

	X_0	X_1	X_2	Y_s
P1	1	1	10	-1
P2	1	5	2	1
P3	1	7	3	1
P4	1	3	7	-1
P5	1	3,5	9	-1

Entrenamiento (P1–P3)

Test (P4–P5)

Tomaremos una taza de apreciación: α = 0,5
Los pesos asociados serán: $w_0 = 1, w_1 = 2, w_2 = 3$

MECANISMOS DE REDES NEURONALES: Perceptrón

- Modo de calculo del Perceptrón:

	X_0	X_1	X_2	Y_s
P1	1	1	10	-1
P2	1	5	2	1
P3	1	7	3	1
P4	1	3	7	-1
P5	1	3,5	9	-1

- Ejemplo de calculo para el aprendizaje:
- Entrada 1 => P1
- $\begin{bmatrix} 1 \\ 1 \\ 10 \end{bmatrix}$
- $g(x) = \sum_{i=0}^{n} w_i . x_i$
- $g(x) = 1 . 1 + 2 . 1 + 3 . 10$
- $g(x) = 33$
- $\boxed{\boldsymbol{o(x) \Rightarrow g(x) > 0 \therefore o(x) = 1}}$

MECANISMOS DE REDES NEURONALES: Perceptrón

- Modo de calculo del Perceptrón:

	X_0	X_1	X_2	Y_s
P1	1	1	10	-1
P2	1	5	2	1
P3	1	7	3	1
P4	1	3	7	-1
P5	1	3,5	9	-1

- Ejemplo de calculo para el aprendizaje:
- Entrada 1 => P1
- $g(x) = 33$
- $\boldsymbol{o(x) \Rightarrow g(x) > 0 \therefore o(x) = 1}$
- **Llamando a $\boldsymbol{o(x)}$ como $\boldsymbol{y_{obt}}$**
- **Calculamos el error paso 1:**
- $\boldsymbol{\varepsilon_1 = y_s - y_{obt} = -1 - 1}$
- $\boldsymbol{\varepsilon_1 = -2}$

MECANISMOS DE REDES NEURONALES: Perceptrón

- Modo de calculo del Perceptrón:

	X_0	X_1	X_2	Y_s
P1	1	1	10	-1
P2	1	5	2	1
P3	1	7	3	1
P4	1	3	7	-1
P5	1	3,5	9	-1

- Ejemplo de calculo para el aprendizaje:
- Entrada 1 => P1
- $\varepsilon_1 = -2$
- $\vec{w}_{k+1} = \vec{w}_k + \alpha \,.\, \varepsilon_k \,.\, \vec{x}_k$
- $\vec{w}_2 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} + 0,5 \,.\, (-2) . \begin{bmatrix} 1 \\ 1 \\ 10 \end{bmatrix}$
- $\vec{w}_2 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} + \begin{bmatrix} -1 \\ -1 \\ -10 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ -7 \end{bmatrix}$

MECANISMOS DE REDES NEURONALES: Perceptrón

- <u>Modo de calculo del Perceptrón:</u>

	X_0	X_1	X_2	Y_s
P1	1	1	10	-1
P2	1	5	2	1
P3	1	7	3	1
P4	1	3	7	-1
P5	1	3,5	9	-1

- Ejemplo de calculo para el aprendizaje:
- Entrada 2 => P2
- $\begin{bmatrix} 1 \\ 5 \\ 2 \end{bmatrix}$
- $g(x) = \sum_{i=0}^{n} w_i . x_i$
- $g(x) = 0 . 1 + 1 . 5 + (-7) . 2$
- $g(x) = -9$
- $\boxed{o(x) \Rightarrow g(x) < 0 \therefore o(x) = -1}$

MECANISMOS DE REDES NEURONALES: Perceptrón

- Modo de calculo del Perceptrón:

	X_0	X_1	X_2	Y_s
P1	1	1	10	-1
P2	1	5	2	1
P3	1	7	3	1
P4	1	3	7	-1
P5	1	3,5	9	-1

- Ejemplo de calculo para el aprendizaje:
- Entrada 2 => P2
- $g(x) = -9$
- $\boldsymbol{o(x) \Rightarrow g(x) < 0 \therefore o(x) = -1}$
- **Llamando a $\boldsymbol{o(x)}$ como $\boldsymbol{y_{obt}}$**
- **Calculamos el error paso 2:**
- $\boldsymbol{\varepsilon_2 = y_s - y_{obt} = 1 - (-1)}$
- $\boldsymbol{\varepsilon_2 = 2}$

MECANISMOS DE REDES NEURONALES: Perceptrón

- Modo de calculo del Perceptrón:

	X_0	X_1	X_2	Y_s
P1	1	1	10	-1
P2	1	5	2	1
P3	1	7	3	1
P4	1	3	7	-1
P5	1	3,5	9	-1

- Ejemplo de calculo para el aprendizaje:
- Entrada 2 => P2
- $\varepsilon_2 = 2$
- $\vec{w}_{k+1} = \vec{w}_k + \alpha \,.\, \varepsilon_k \,.\, \vec{x}_k$
- $\vec{w}_3 = \begin{bmatrix} 0 \\ 1 \\ -7 \end{bmatrix} + 0,5 \,.\, 2 \,.\, \begin{bmatrix} 1 \\ 5 \\ 2 \end{bmatrix}$
- $\vec{w}_3 = \begin{bmatrix} 0 \\ 1 \\ -7 \end{bmatrix} + \begin{bmatrix} 1 \\ 5 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 6 \\ -5 \end{bmatrix}$

MECANISMOS DE REDES NEURONALES: Perceptrón

- Modo de calculo del Perceptrón:

	X_0	X_1	X_2	Y_s
P1	1	1	10	-1
P2	1	5	2	1
P3	1	7	3	1
P4	1	3	7	-1
P5	1	3,5	9	-1

- Ejemplo de calculo para el aprendizaje:
- Entrada 3 => P3
- $\begin{bmatrix} 1 \\ 7 \\ 3 \end{bmatrix}$
- $g(x) = \sum_{i=0}^{n} w_i . x_i$
- $g(x) = 1.1 + 6.7 + (-5).3$
- $g(x) = 28$
- $\boldsymbol{o(x) \Rightarrow g(x) > 0 \therefore o(x) = 1}$

MECANISMOS DE REDES NEURONALES: Perceptrón

- Modo de calculo del Perceptrón:

	X_0	X_1	X_2	Y_s
P1	1	1	10	-1
P2	1	5	2	1
P3	1	7	3	1
P4	1	3	7	-1
P5	1	3,5	9	-1

- Ejemplo de calculo para el aprendizaje:
- Entrada 3 => P3
- $g(x) = 28$
- $o(x) \Rightarrow g(x) > 0 \therefore o(x) = 1$
- **Llamando a $o(x)$ como y_{obt}**
- **Calculamos el error paso 3:**
- $\varepsilon_3 = y_s - y_{obt} = 1 - 1$
- $\varepsilon_3 = 0$

MECANISMOS DE REDES NEURONALES: Perceptrón

- Modo de calculo del Perceptrón:

	X_0	X_1	X_2	Y_s
P1	1	1	10	-1
P2	1	5	2	1
P3	1	7	3	1
P4	1	3	7	-1
P5	1	3,5	9	-1

Error = 0.
Termina el proceso de aprendizaje

- Ejemplo de calculo para el aprendizaje:
- Entrada 3 => P3
- $\varepsilon_3 = 0$
- $\vec{w}_{k+1} = \vec{w}_k + \alpha \,.\, \varepsilon_k \,.\, \vec{x}_k$
- $\vec{w}_4 = \begin{bmatrix} 1 \\ 6 \\ -5 \end{bmatrix} + 0,5 \,.\, 0 \,.\, \begin{bmatrix} 1 \\ 7 \\ 3 \end{bmatrix}$
- $\vec{w}_4 = \begin{bmatrix} 1 \\ 6 \\ -5 \end{bmatrix}$

MECANISMOS DE REDES NEURONALES: Perceptrón

- Modo de calculo del Perceptrón:

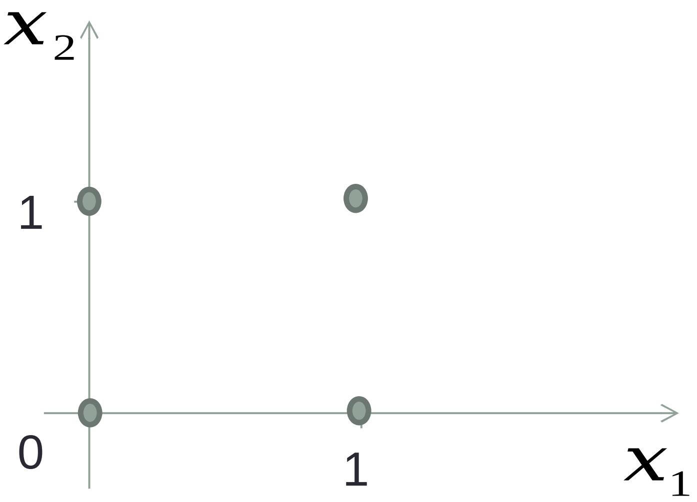
	X_0	X_1	X_2	Y_s
P1	1	1	10	-1
P2	1	5	2	1
P3	1	7	3	1
P4	1	3	7	-1
P5	1	3,5	9	-1

- Ejemplo de calculo para el aprendizaje:
- Testeamos con P4 y los pesos entrenados
- $\begin{bmatrix} 1 \\ 3 \\ 7 \end{bmatrix}$
- $g(x) = \sum_{i=0}^{n} w_i . x_i$
- $g(x) = 1 . 1 + 3 . 6 + (-5) . 7$
- $g(x) = -16$
- $\boxed{\boldsymbol{o(x) \Rightarrow g(x) < 0 \therefore o(x) = -1}}$

MECANISMOS DE REDES NEURONALES: Perceptrón

- <u>Modo de calculo del Perceptrón:</u>
- Para una compuerta XOR:
- ¿Por donde pasaría la recta de decisión?¿?¿?

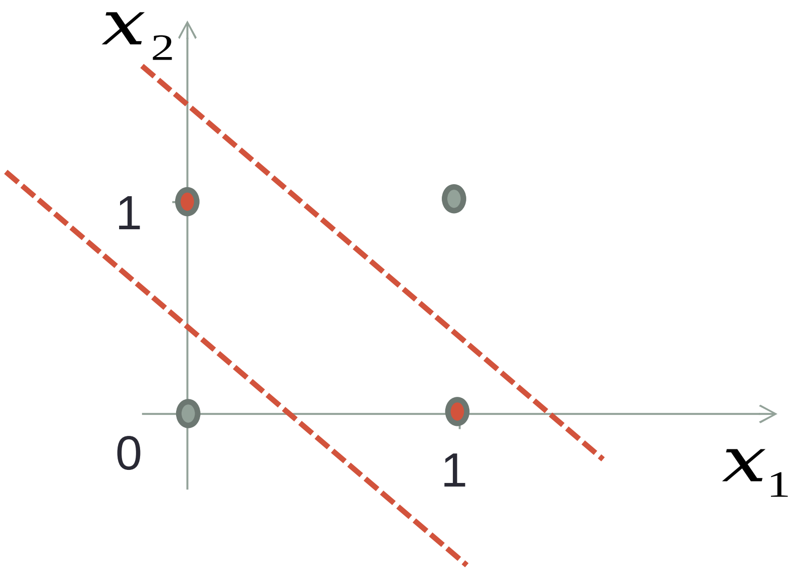
X_1	X_2	XOR
0	0	0
0	1	1
1	0	1
1	1	0



MECANISMOS DE REDES NEURONALES: Perceptrón

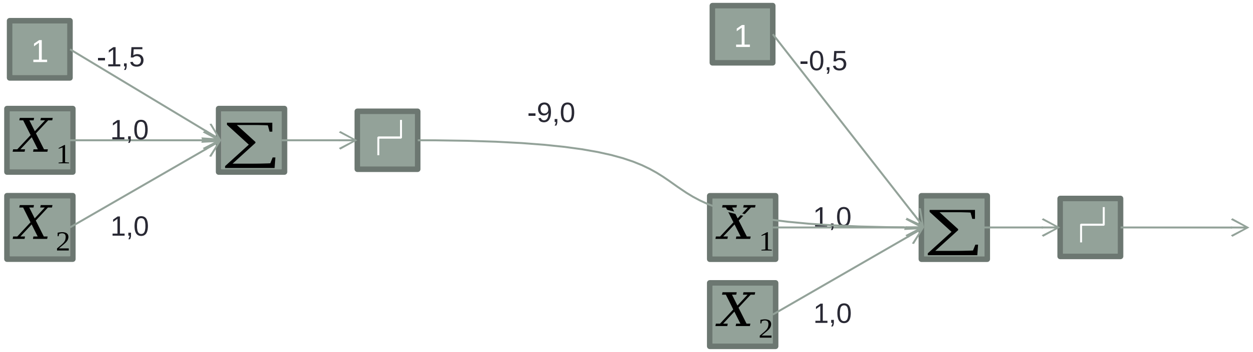
- <u>Modo de calculo del Perceptrón:</u>
- Para una compuerta XOR:
- ¿Por donde pasaría la recta de decisión?¿?¿?

X_1	X_2	XOR
0	0	0
0	1	1
1	0	1
1	1	0



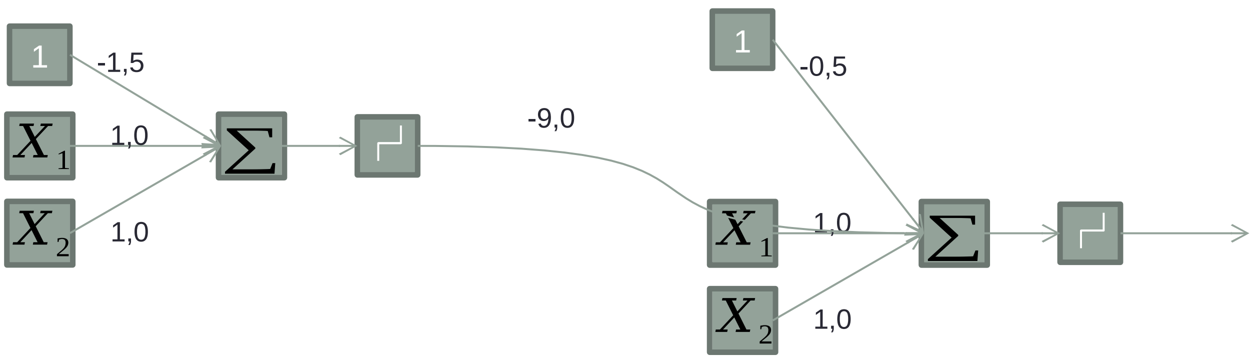
MECANISMOS DE REDES NEURONALES: Perceptrón

- <u>Modo de calculo del Perceptrón:</u>
- Para una compuerta XOR:
- ¿Cómo puedo resolverlo?¿?¿?



MECANISMOS DE REDES NEURONALES: Perceptrón

- <u>Modo de calculo del Perceptrón:</u>
- Para una compuerta XOR:
- La salida del primer perceptrón sirve como entrada del segundo perceptrón con una conexión contrapesada muy negativa.
- El inconveniente es que el algoritmo de aprendizaje del perceptrón no puede ajustar correctamente los pesos entre los perceptrones.

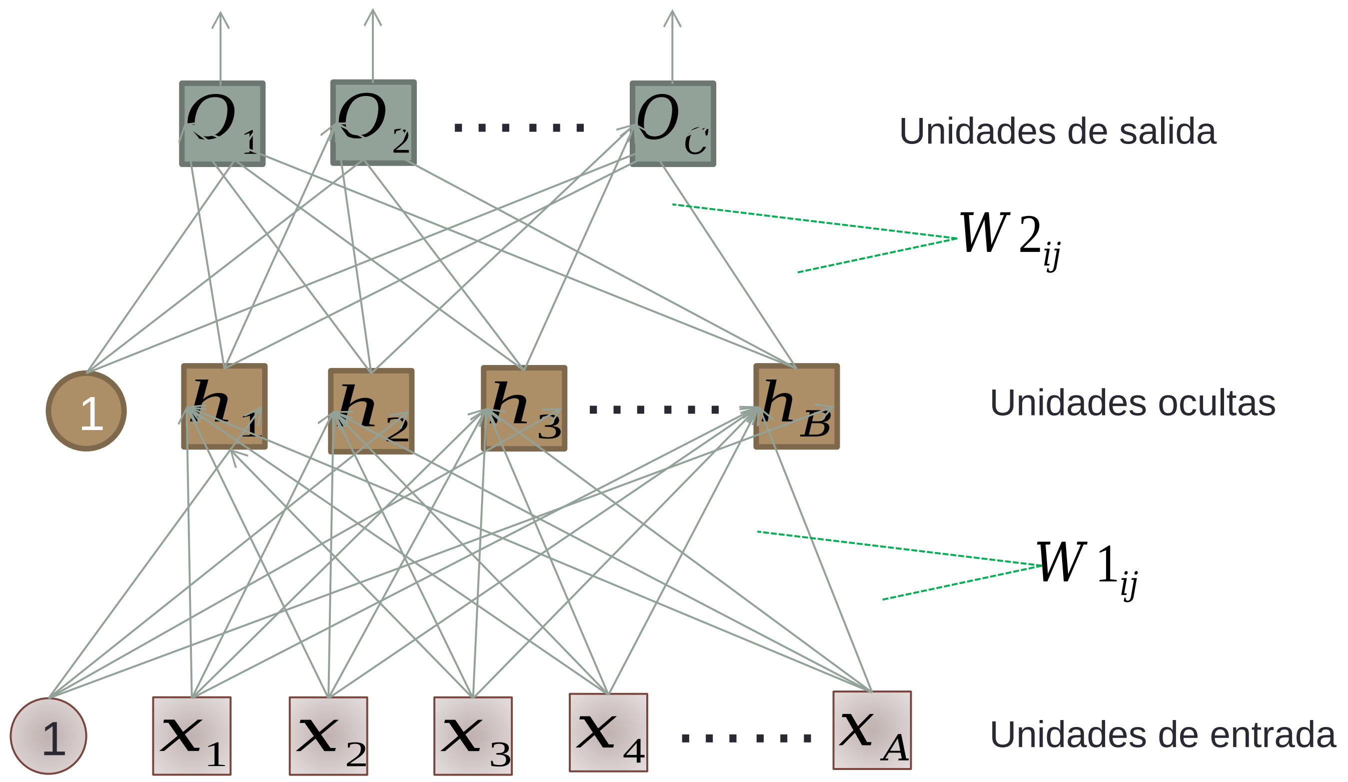


MECANISMOS DE REDES NEURONALES: Redes multicapa

- Red backpropagation (propagación hacia atrás):
- Recordemos que el objetivo es tomar una masa relativamente «amorfa» de elementos similares a las neuronas y entonces enseñarles a realizar cualquier tipo de tarea.
- Este objetivo lo logramos con redes multicapa que nos permiten calcular cualquier cosa. Y la propagación hacia atrás nos lleva en esa dirección.

MECANISMOS DE REDES NEURONALES: Redes multicapa

- Red backpropagation (propagación hacia atrás):



MECANISMOS DE REDES NEURONALES: Redes multicapa

- <u>Red backpropagation (propagación hacia atrás):</u>
- Características:
  - $W1_{ij}$ denota los pesos de las conexiones entre la capas de entrada y la oculta.
  - $W2_{ij}$ denota los pesos de las conexiones entre las capas de salida y la oculta.
  - Este ejemplo tiene tres capas pero puede tener más según necesidad.
  - Las activaciones «saltan» desde la capa de entrada hacia la capa oculta y desde ésta hacia la capa de salida.
  - El conocimiento de la red está codificado en los pesos de las conexiones entre unidades.
  - Los niveles de activación en la capa de salida determinan la salida de la red.

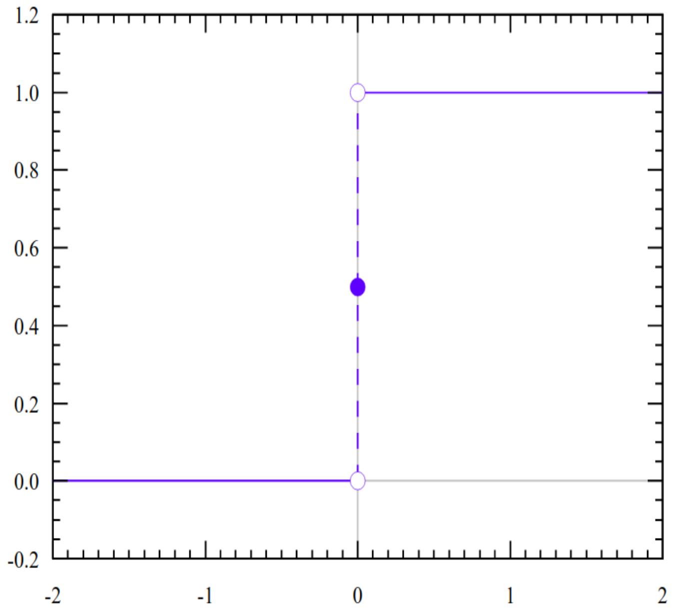
MECANISMOS DE REDES NEURONALES: Redes multicapa

- <u>Red backpropagation (propagación hacia atrás):</u>
- Como calcular la cantidad de nodos ocultos:
  - La cantidad de neuronas que debe contener cada capa oculta es difícil de determinar ya que no existe una única regla que permita calcularlas y muchas veces se hace de manera empírica.
  - Una idea es mantener el numero de neuronas por capa lo mas bajo posible ya que cada neurona es un costo de procesamiento extra.
  - Otra idea es aplicar la regla de Baum-Haussler: $N_{oculta} \leq \dfrac{N_{entren} \cdot E_{tolerable}}{N_{entrada} \cdot N_{salida}}$
  - Donde:
    - N_{entren} = Nro patrones entrenamiento
    - $E_{tolerable}$ = Error deseado de la red
    - $N_{entrada}$ = Nro de nodos de entrada
    - N_{salida} = Nro de nodos de salida

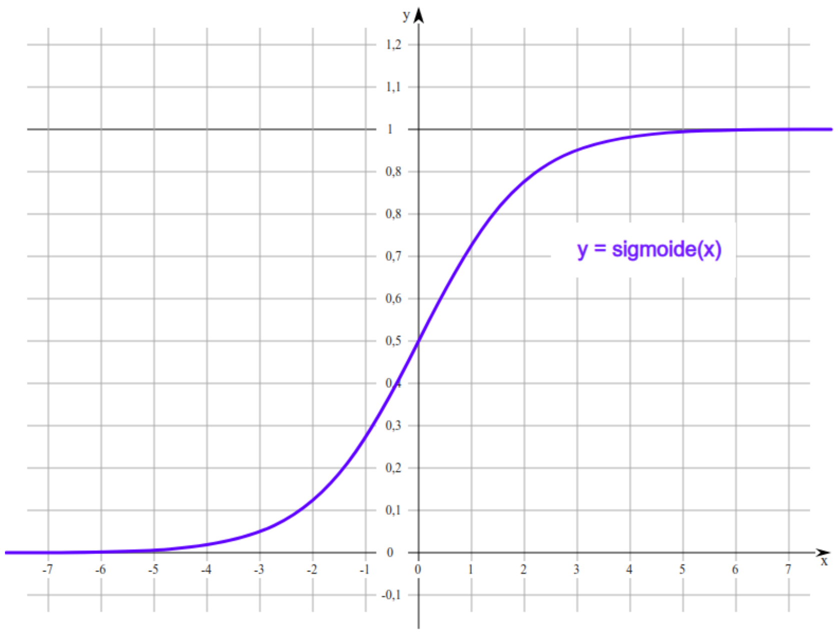
MECANISMOS DE REDES NEURONALES: Redes multicapa

Red backpropagation (propagación hacia atrás):

Función de activación: (escalón)
Perceptrón



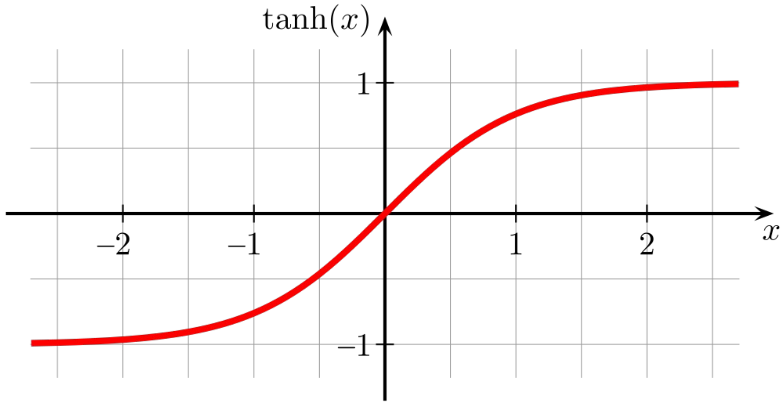
Función de activación: (sigmoide)
Backpropagation



MECANISMOS DE REDES NEURONALES: Redes multicapa

Red backpropagation (propagación hacia atrás):

Función de activación: (Tangente hiperbólica)
Backpropagation



$$\tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

MECANISMOS DE REDES NEURONALES: Redes multicapa

- Red backpropagation (propagación hacia atrás):
- Función de activación:
  - Aun hay una unidad de propagación hacia atrás que suma sus entradas ponderadas.
  - Produce un valor entre 0 y 1 basado en la función sigmoide continua y diferenciable.
  - La ecuación de salida de dicha función es:
  - Si sum = 0 la salida se hace 0,5
  - A medida que suma aumenta la salida se aproxima a 1
  - A medida que suma se hace pequeña la salida se aproxima a 0

$$salida = \frac{1}{1 + e^{-suma}}$$

MECANISMOS DE REDES NEURONALES: Redes multicapa

- Red backpropagation (propagación hacia atrás):
- **Funcionamiento simplificado del algoritmo:**
  - Comienza con un conjunto de pesos aleatorios.
  - La red ajusta sus pesos cada vez que ve un par entrada/salida.
  - Cada par requiere dos etapas:
    - **Paso hacia adelante:** se presenta un ejemplo de entrada hacia la red y permite que las activaciones continúen hasta que alcancen la capa de salida.
    - **Paso hacia atrás:** la salida que se tiene de la red en ese momento se compara con la salida objetivo y se calcula el error estimado de las unidades de salida. Se ajustan los pesos asociados a las unidades de salida para reducir dichos errores. Se deriva el error estimado de las capas de salida a las capas ocultas. Por último, los errores se propagan hacia atrás a las conexiones procedentes de las unidades de entrada.

MECANISMOS DE REDES NEURONALES: Redes multicapa

- Red backpropagation (propagación hacia atrás):
- **Funcionamiento del algoritmo:**
  - Dado: un conjunto de pares de vectores de E/S.
  - Calcular: un conjunto de pesos para una red de tres capas, que haga corresponder las entradas con las correspondientes salidas.

1. Sea A=nro de unidades de capa de entrada. Sea C=nro de unidades de capa de salida. Elegir B=nro de unidades de capa oculta. La capa de entrada y oculta poseen una unidad extra utilizada como umbral.
   a) Las unidades de la capa de entrada van desde 0 a A.
   b) Las unidades de la capa oculta van desde 0 a B.
   c) Los pesos entre la capa de entrada y oculta se denotan por $w1_{ij}$ donde i=indexa las unidades de la capa de entrada y j=indexa las unidades ocultas.
   d) Los pesos entre la capa oculta y la de salida se denotan por $w2_{ij}$ donde i=indexa las unidades de la capa oculta y j=indexa las unidades de salida.
2. Inicializar los pesos de la red con valores aleatorios entre -0,1 y 0,1

$$w1_{ij} = aleatorio(-0{,}1; 0{,}1) \; \forall i = 0, \ldots, A; j = 1, \ldots, B$$
$$w2_{ij} = aleatorio(-0{,}1; 0{,}1) \; \forall i = 0, \ldots, B; j = 1, \ldots, C$$

3. Inicializar las activaciones de las unidades umbral (nunca deben cambiar): $X_0 = 1$ y $h_0 = 1$
4. Elegir un par entrada/salida. Suponga que el vector de entrada es X_i y que el vector de salida es Y_i. Asigne los niveles de activación a las unidades de entrada.

MECANISMOS DE REDES NEURONALES: Redes multicapa

- Red backpropagation (propagación hacia atrás):
- **Funcionamiento del algoritmo:**

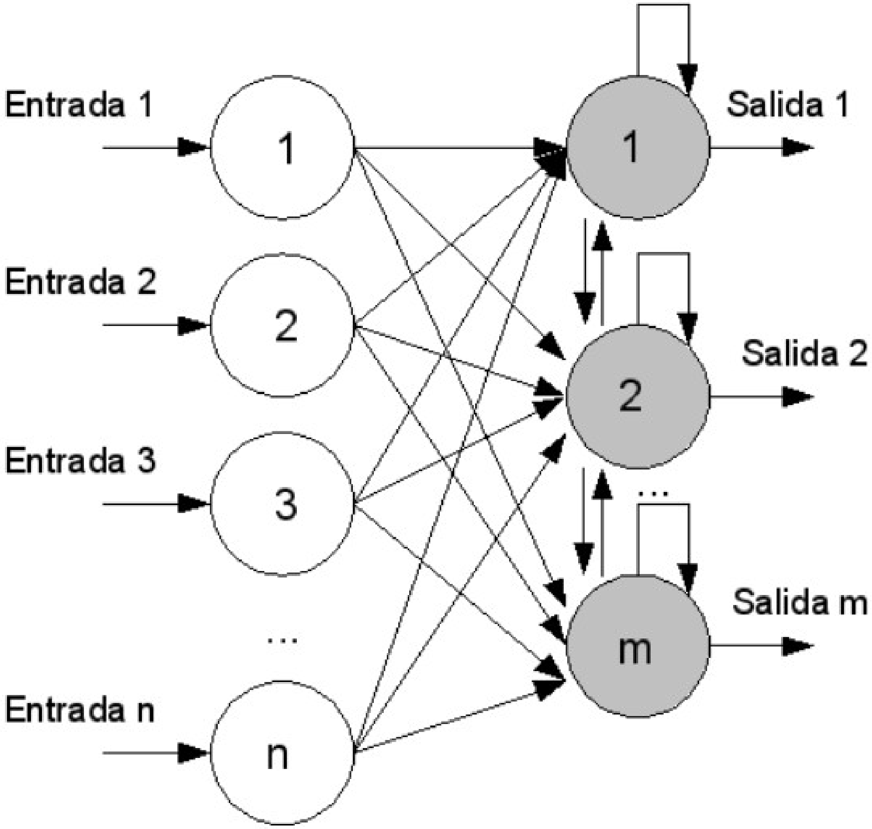
5. Propagar las activaciones desde las unidades entrada hacia las unidades de la capa oculta utilizando la función de activación: $h_j = \frac{1}{1+e^{-(\sum_{i=0}^{A} w1_{ij}\, x_i)}}$
6. Propagar las activaciones de las unidades ocultas a las unidades de salida : $O_j = \frac{1}{1+e^{-(\sum_{i=0}^{B} w2_{ij}\,.\, h_i)}}$
7. Calcular los errores de la capa de salida denominado
   a) $\delta 2_j = (1 - O_j).(y_j - O_j) \quad \forall\ j = 1, \dots, C$
8. Calcular los errores de las unidades de la capa oculta denominado $\delta 1_j = (1 - h_j).\sum_{i=1}^{C} \delta 2_j\,.w2_{ji} \quad \forall \quad j = 1, \dots, B$
9. Ajustar los pesos entre la capa oculta y la de salida: $\propto = 0{,}35$
   a) $\Delta w2_{ij} = \alpha\,.\,\delta 2_j\,.h_i \ \forall\ i = 0, \dots, B\,; j = 1, \dots, C$
10. Ajustar los pesos entre la capa de entrada y la oculta:
    a) $\Delta w1_{ij} = \alpha\,.\,\delta 1_j\,.x_i \ \forall\ i = 0, \dots,\ A\,; j = 1, \dots, B$
11. Ir al paso 4 y repetir. Cuando se hayan introducido en la red todos los pares de entrada/salida se completa una etapa. Repetir los pasos del 4 al 10 para tantas etapas como se desee.

MECANISMOS DE REDES NEURONALES: Redes competitivas

- Red Kohonen:
  - Basadas en el principio del funcionamiento del cerebro que indica que hay neuronas que se organizan en zonas (mapas bidimensionales).
  - Por ejemplo en el sistema auditivo se detecta una organización según la frecuencia a la que cada neurona alcanza mayor frecuencia.
  - Una parte de esta organización es genéticamente heredada pero gran parte de ella se origina mediante el aprendizaje.
  - T. Kohonen presento, en 1982, una red con un comportamiento similar al descripto.
  - El modelo de Kohonen tiene la capacidad de formar mapas de características de manera similar a como ocurre en el cerebro humano.
  - Se aplica al reconocimiento de voz y al texto manuscrito.
  - El modelo tiene dos variantes:
    - LVQ (Learning Vector Quantization): para vectores de entrada de una sola dimensión
    - TPM (Topology Preserving Map) o SOM (Self Organization Map): para vectores de entrada bidimensionales o incluso tridimensionales en TPM.

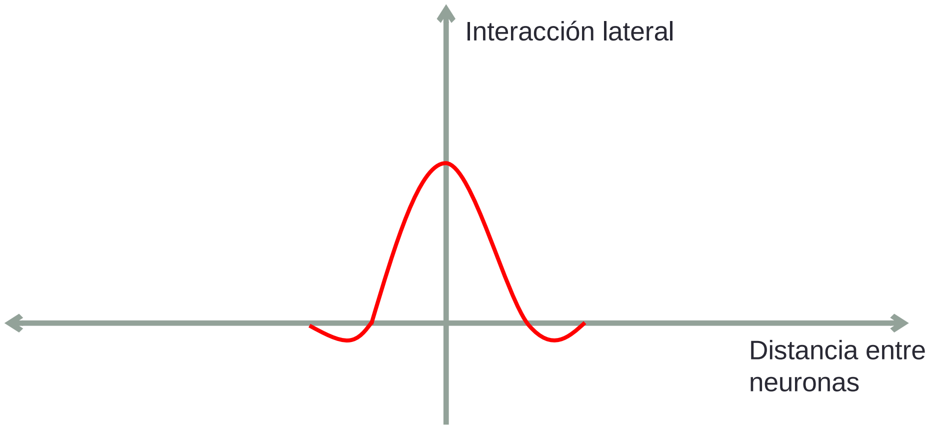
MECANISMOS DE REDES NEURONALES: Redes competitivas

- <u>Red Kohonen:</u>



MECANISMOS DE REDES NEURONALES: Redes competitivas

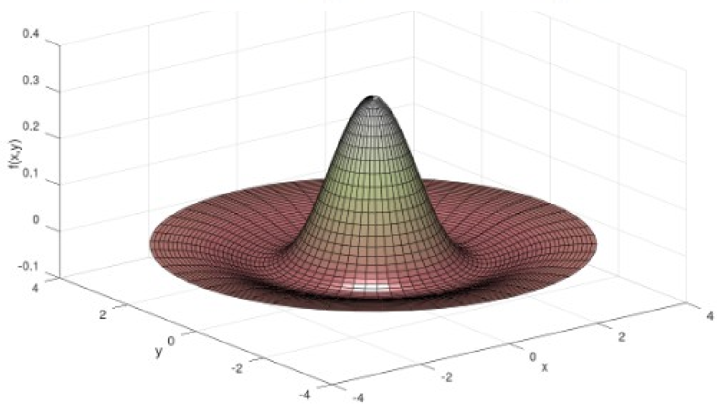
- Red Kohonen:
  - Es una red de dos capas con N neuronas de entrada y M de salida.
  - Cada una de las N neuronas de entrada se conecta con las M de salida a través de conexiones feedforward.
  - Entre las neuronas de la capa de salida existen conexiones laterales de inhibición (peso negativo) indicando que cada neurona va a tener cierta influencia sobre sus vecinas.
  - El valor que se le vaya a asignar a los pesos de las conexiones feedforward entre la capa de entrada y salida (w_{ij}), durante el aprendizaje, dependerá de dichas interacciones laterales.
  - Influencia de cada neurona de salida sobre las demás:



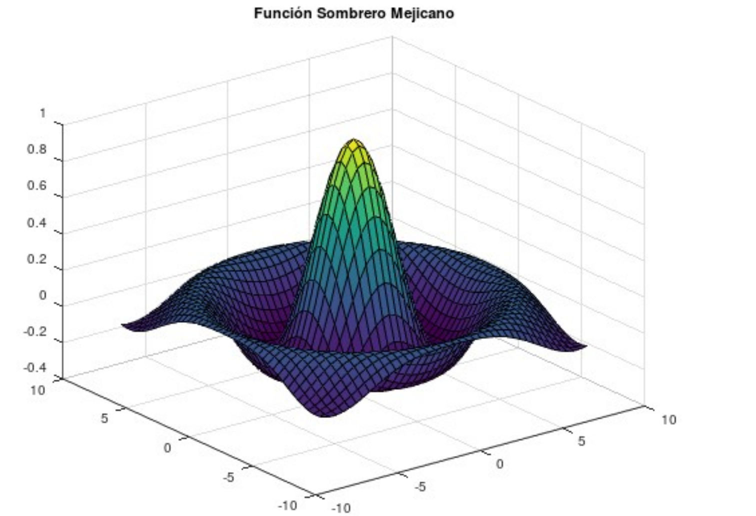
MECANISMOS DE REDES NEURONALES: Redes competitivas

- <u>Red Kohonen:</u>
  - Mas sobre la función sombrero mejicano (**ondícula de Ricker**):
    - https://wiki.seg.org/wiki/Dictionary:Ricker_wavelet/es
    - https://en.wikipedia.org/wiki/Ricker_wavelet
    - http://rjuarezs.com/t_20n.html

$$f(x,y) = \frac{1}{\pi\sigma^2}\left(1 - \frac{x^2+y^2}{2\sigma^2}\right)e^{-\frac{x^2+y^2}{2\sigma^2}}$$




```
z = sin (sqrt (x^2 + y^2)) / (sqrt (x^2 + y^2))
```



MECANISMOS DE REDES NEURONALES: Redes competitivas

- Red Kohonen :
- Funcionamiento simplificado:
  - Se presenta una entrada $E_k = (e_1{}^{(k)} e_2{}^{(k)} \ldots . \, e_N{}^{(k)})$
  - Cada una de las neuronas de la capa de salida recibe una parte de esa entrada por medio de las conexiones feedforward con pesos W_{ij}.
  - También cada neurona de salida reciben las entradas correspondientes a las interacciones laterales.
  - Salida generada por una neurona de salida j ante un vector de entrada E_k:
    - $S_j(t+1) = f(\sum_{i=1}^{N} w_{ij}\, e_i{}^{(k)} + \sum_{p=1}^{M} Int_{pj}\, S_p(t))$
    - Donde Int_{pj} es una función de tipo sombrero mejicano que representa la influencia lateral de la neurona p sobre la neurona j.

MECANISMOS DE REDES NEURONALES: Redes competitivas

- <u>Red Kohonen :</u>
- Funcionamiento simplificado:
  - La función de activación de las neuronas de salida será del tipo continuo, lineal o sigmoidal.
  - Esta red trabaja con valores reales.
  - La red evoluciona ante una entrada E_k hasta alcanzar un estado estable en el que solo hay una neurona activa (la ganadora).
  - Funcionamiento simplificado:
    - $$S_j = \begin{cases} 1 \rightarrow MIN \|E_k - W_j\| = MIN\left(\sqrt{\sum_{i=1}^{N}(e^{(k)} - w_{ij})^2}\right) \\ 0 \rightarrow rest \end{cases}$$
  - Donde $\|E_k - W_j\|$ es una medida de la diferencia entre el vector de entrada y el vector de pesos de las conexiones que llegan a la neurona j desde la entrada.
  - **Durante el funcionamiento se pretende encontrar el dato aprendido mas parecido al de entrada para averiguar que neurona se activara.**

MECANISMOS DE REDES NEURONALES: Redes competitivas

- Red Kohonen :
- Algoritmo de aprendizaje:
  - Es del tipo off line.
  - No supervisado de tipo competitivo.
  - Las neuronas de la capa de salida compiten por activarse.
  - Solo uno de ellas permanecerá activa ante una determinada entrada.
  - Pasos del algoritmo:
    - Inicializar los pesos w_{ij} con valores aleatorios pequeños y se fija la zona inicial de vecindad entre las neuronas de salida.
    - Presentar a la red una entrada: $E_k = (e_1{}^{(k)} e_2{}^{(k)} \ldots\ldots e_N{}^{(k)})$ cuyas componentes son valores **continuos.**
    - Determinar la neurona vencedora a la salida. Esta será aquella **j** cuyo vector de pesos W_j sea el mas parecido a la información de entrada E_k. Para ello se debe calcular las distancias entre ambos vectores, una para cada neurona de salida:
      - Donde:
        - $e_i{}^{(k)}$: Componente i-esimo del vector k-esimo de entrada.
        - w_{ij}: Peso de la conexión entre las neuronas i (de entrada) y j (de salida).

MECANISMOS DE REDES NEURONALES: Redes competitivas

- <u>Red Kohonen :</u>
- Algoritmo de aprendizaje:
  - Una vez localizada la neurona vencedora j* se actualizan los pesos de las conexiones feedforward que llegan a dicha neurona y a sus vecinas:
    - $w_{ij}(t+1) = w_{ij}(t) + \alpha(t)\left[e_i^{(k)} - w_{ij*}(t)\right] \qquad j \in Zona_{j*}(t)$
    - Donde:
      - $Zona_{j*}(t)$ es la zona de vecindad de la neurona vencedora j*
      - $\alpha(t)$ es el coeficiente de aprendizaje entre 0 y 1 el cual decrece con cada iteración:
        - Suele tener la siguiente expresión: $\alpha(t) = \frac{1}{t}$
  - El proceso debe repetirse hasta lograr que los pesos se estabilicen y tiendan a un cierto valor de error (pequeño) o por lo menos iterar un mínimo de 500 veces