

Aprendizaje Automatizado

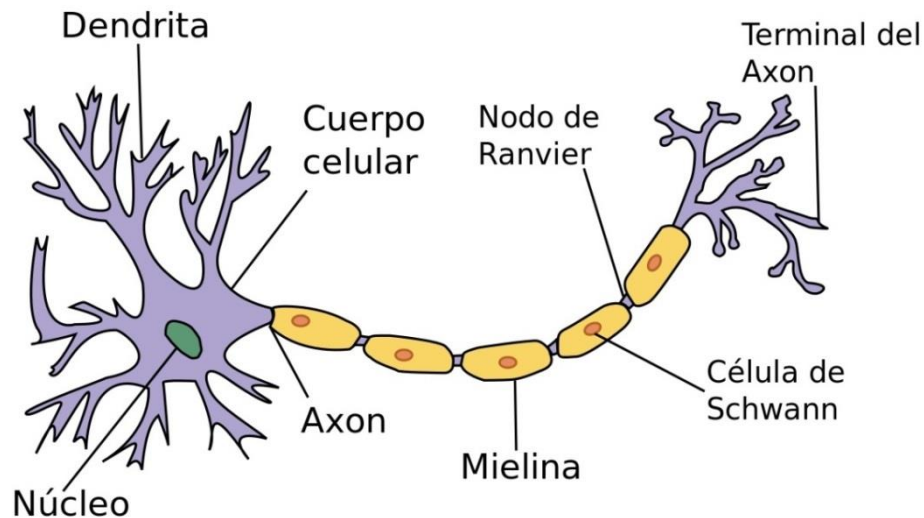
Redes Neuronales
Artificiales



Introducción RNA

- Una aproximación robusta para aproximar funciones objetivos a valores reales y discretos.
- Inspiraciones biológicas:
 - Usar RN para modelar y estudiar procesos de aprendizaje
 - Obtener algoritmos de ML muy efectivos, inspirados en el cerebro humano, mediante modelos artificiales.

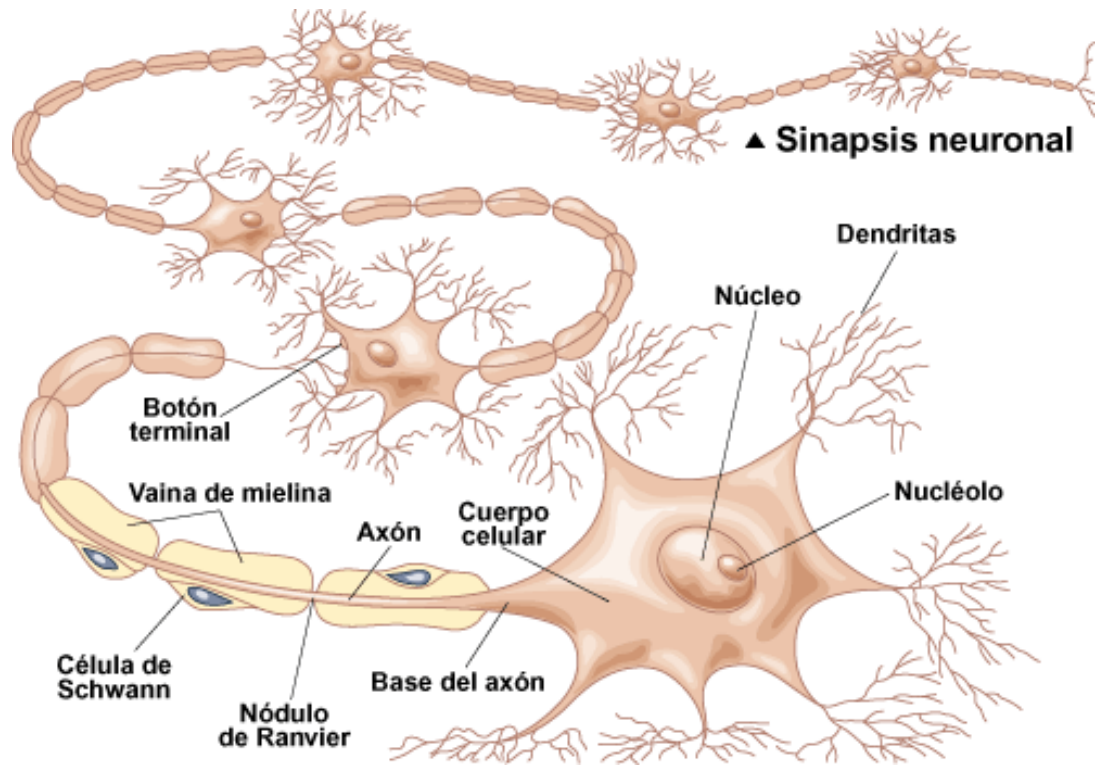
Fundamentos - El modelo biológico



- El cerebro humano contiene más de cien mil millones de neuronas.
- La clave para el procesamiento de la información son las conexiones entre ellas llamadas **sinápsis**.

Estructura biológica

- Las **dendritas** son la vía de entrada de las señales que se combinan en el cuerpo de la neurona.
- El **axón** es el camino de salida de la señal generada por la neurona.
- La neurona es estimulada por sus entradas y cuando alcanza cierto umbral, **se activa** pasando una señal hacia el axón.



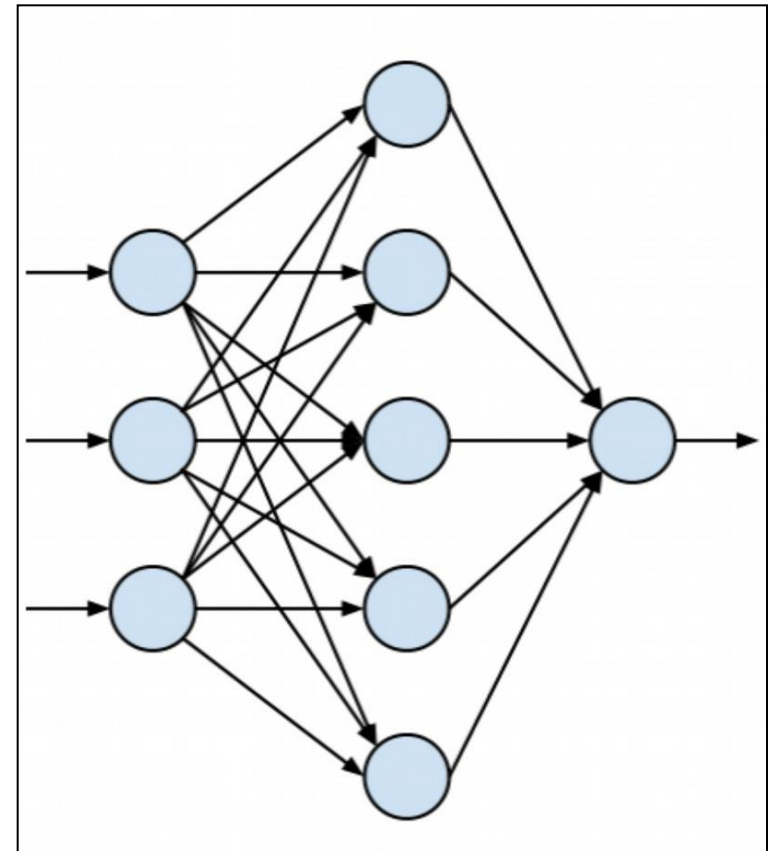
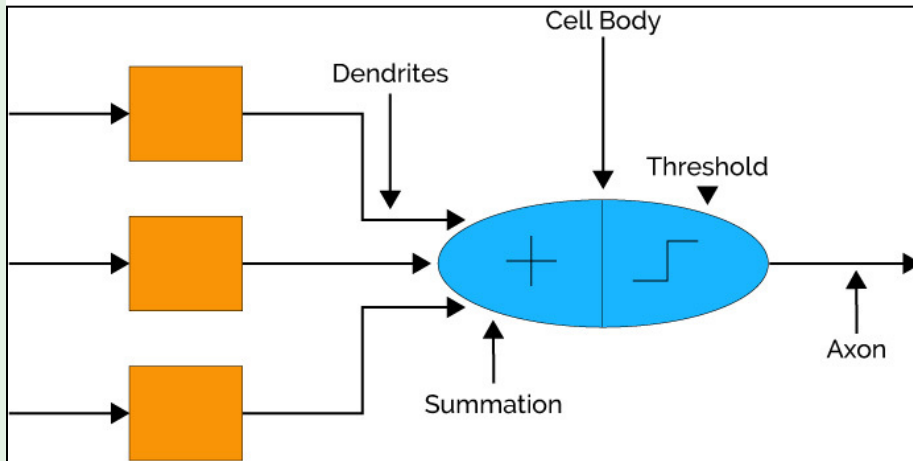


MODELO NEURONAL ARQUITECTURA DE REDES

Definición

- Una Red Neuronal Artificial (RNA) es un sistema para el tratamiento de la información, cuya unidad básica de procesamiento está inspirada en **la neurona**.
- Las neuronas son un componente relativamente simple pero que conectadas de a muchas, forman un poderoso sistema.

Descripción de las RNA



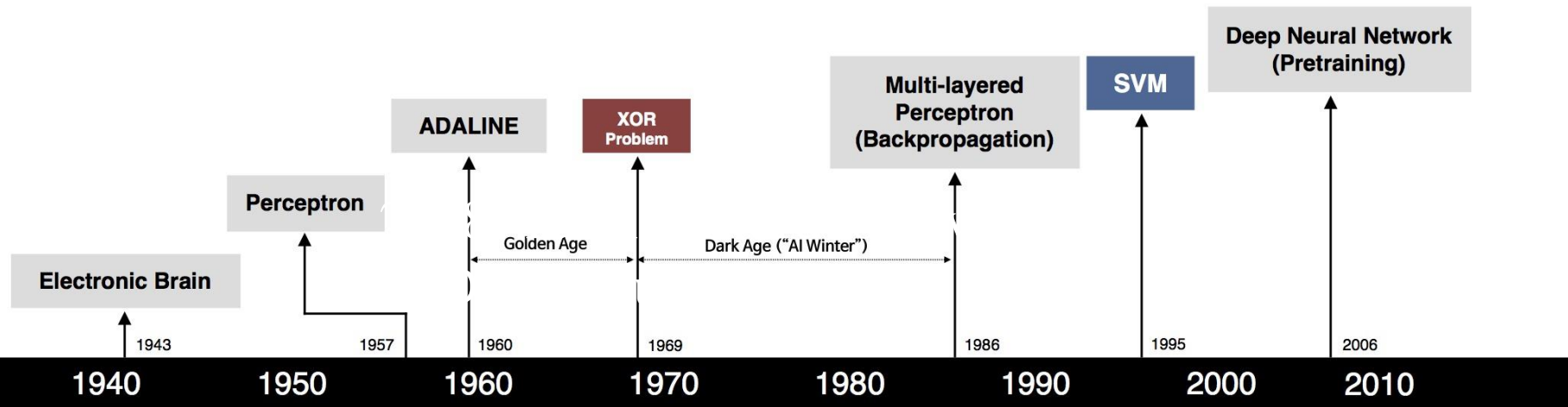
Descripción de las RNA

- Se basan en una red de unidades de procesamiento que intercambian datos o información.
- Tienen la capacidad de aprender y mejorar su funcionamiento.
- Se utilizan para reconocer patrones, incluyendo imágenes, manuscritos, tendencias financieras, etc.
- Son uno de los métodos más utilizados en ML!!!

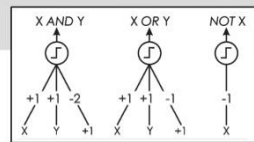
RNA- Tipos de problemas

- Problemas de Regresión y clasificación multiclase.
- Entradas de alta dimensión.
- Los ejemplos de entrenamiento pueden contener errores.
- Son aceptables largos tiempos de entrenamiento.
- Se requiere rápida evaluación de la función objetivo aprendida.
- La comprensión de la función objetivo no es importante.

Evolución de las RNA



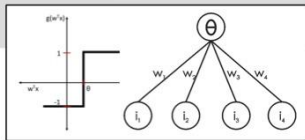
S. McCulloch – W. Pitts



- Adjustable Weights
- Weights are not Learned



F. Rosenblatt



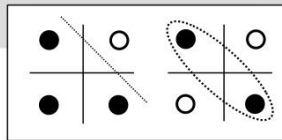
- Learnable Weights and Threshold



B. Widrow – M. Hoff



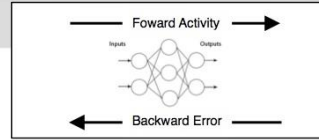
M. Minsky – S. Papert



- XOR Problem



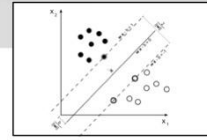
D. Rumelhart – G. Hinton – R. Williams



- Solution to nonlinearly separable problems
- Big computation, local optima and overfitting



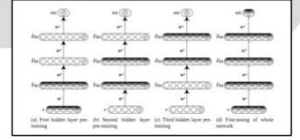
V. Vapnik – C. Cortes



- Limitations of learning prior knowledge
- Kernel function: Human Intervention

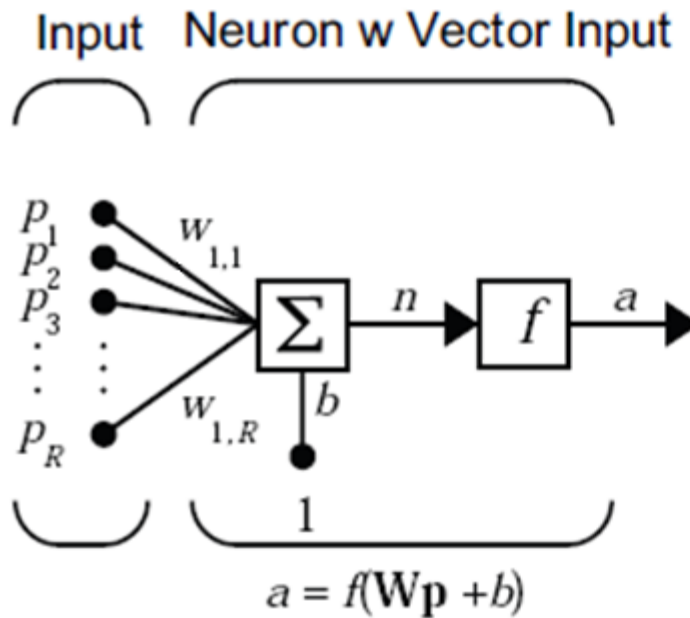


G. Hinton – S. Ruslan



- Hierarchical feature Learning

Neurona artificial

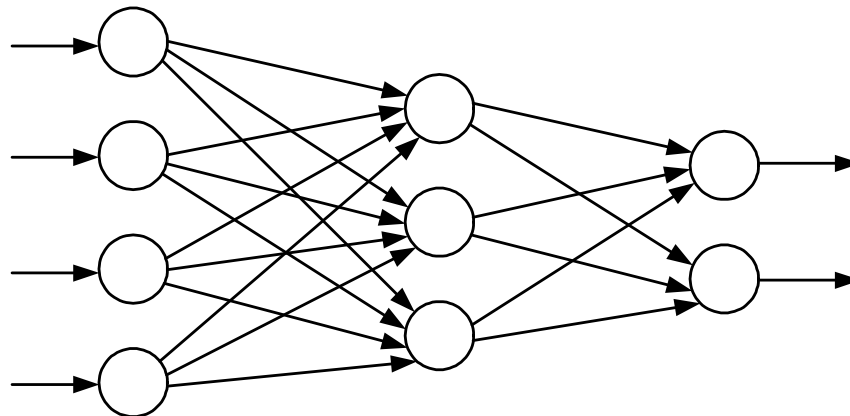


Where...

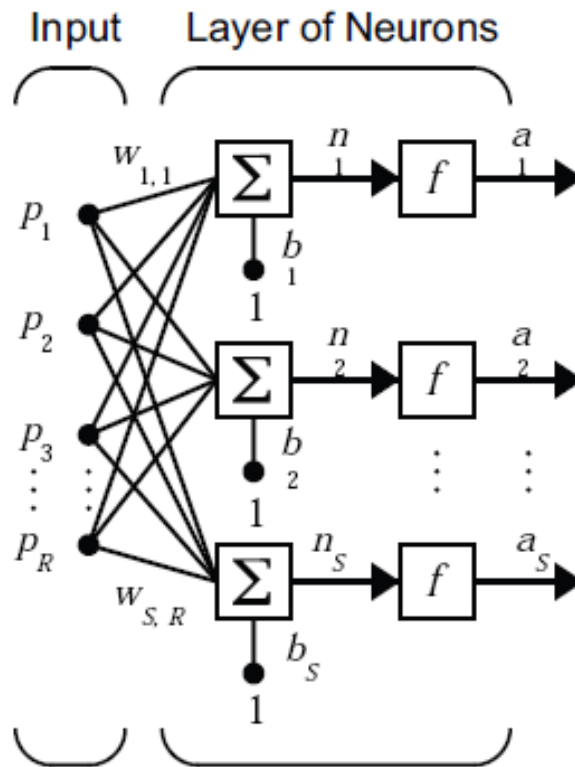
R = number of
elements in
input vector

Estructura Artificial

- Se interconectan neuronas en tres tipos de capas:
 - De entrada: reciben estímulos externos.
 - Oculta: elementos internos de procesamiento (se pueden estructurar en varias capas).
 - De salida: reciben la información procesada y retornan la respuesta del sistema al exterior.



Una capa de neuronas



Where...

R = number of
elements in
input vector

S = number of
neurons in layer

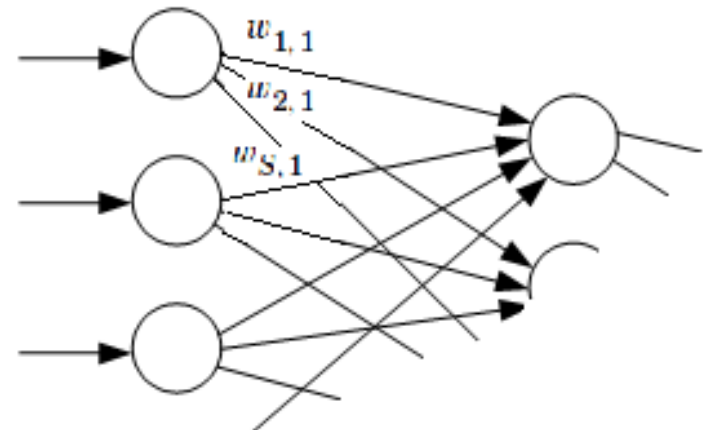
$$\mathbf{a} = \mathbf{f}(\mathbf{Wp} + \mathbf{b})$$

Una capa de neuronas

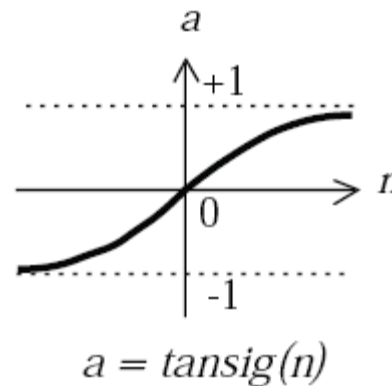
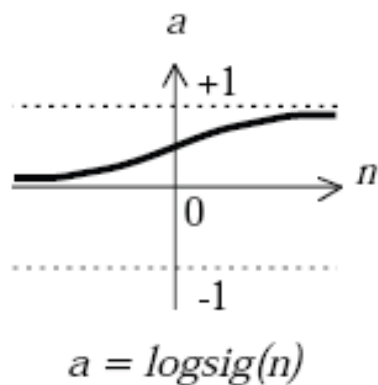
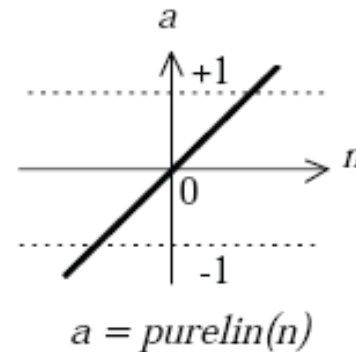
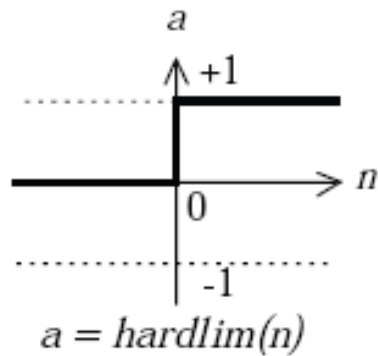
$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & \dots & w_{1,R} \\ w_{2,1} & w_{2,2} & \dots & w_{2,R} \\ \vdots & \vdots & \ddots & \vdots \\ w_{S,1} & w_{S,2} & \dots & w_{S,R} \end{bmatrix}$$

La fila 1 de la matriz de pesos, contiene los pesos de las conexiones que van hacia la primer neurona.

La columna 1 de la matriz de pesos, contiene los pesos de las conexiones que salen desde la primer entrada.



Funciones transferencia



$$a = \text{logsig}(n) = 1/(1+\exp(-n))$$

$$a = \text{tansig}(n) = 2/(1+\exp(-2n)) - 1$$

Características de las RNA

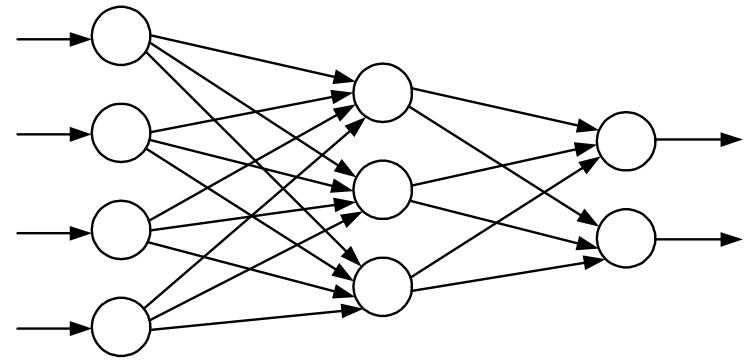
- Arquitectura (Topología).
 - Tipo de RN. Función transferencia
 - Número de capas.
 - Número de neuronas por capa.
 - Tipo de conexiones. Normalmente, todas las neuronas de una capa reciben señales de la capa anterior (más cercana a la entrada) y envían su salida a las neuronas de la capa posterior
 - Feedforward
- Tipo de aprendizaje.

Work flow

El diseño general del trabajo con RNA tiene los siguientes pasos básicos:

- Collect data
- Create the network
- Configure the network
- Initialize the weights and biases
- Train the network
- Validate the network (post-training analysis)
- Use the network

Redes *feedforward*



- Las neuronas de una capa se conectan con las neuronas de la capa siguiente (hacia adelante).
- Las más conocidas son:
 - Perceptrón
 - Adaline
 - Backpropagation
- Son muy útiles en aplicaciones de reconocimiento o clasificación de patrones.

Redes *feedforward-Multiple-layer*

Multiple-layer networks are quite powerful!!!

- For instance, a network of two layers, where one layer is sigmoid and the other layer is linear, can be trained to approximate any function (with a finite number of discontinuities) arbitrarily well.
- This kind of two-layer network is used extensively

Mecanismo de Aprendizaje

- Biológicamente se acepta que la información memorizada en el cerebro se relaciona con los valores sinápticos de las conexiones.
- En las RNA se considera que el conocimiento se encuentra representado en los pesos de las conexiones.
- **El proceso de aprendizaje se basa en cambios en estos pesos.**

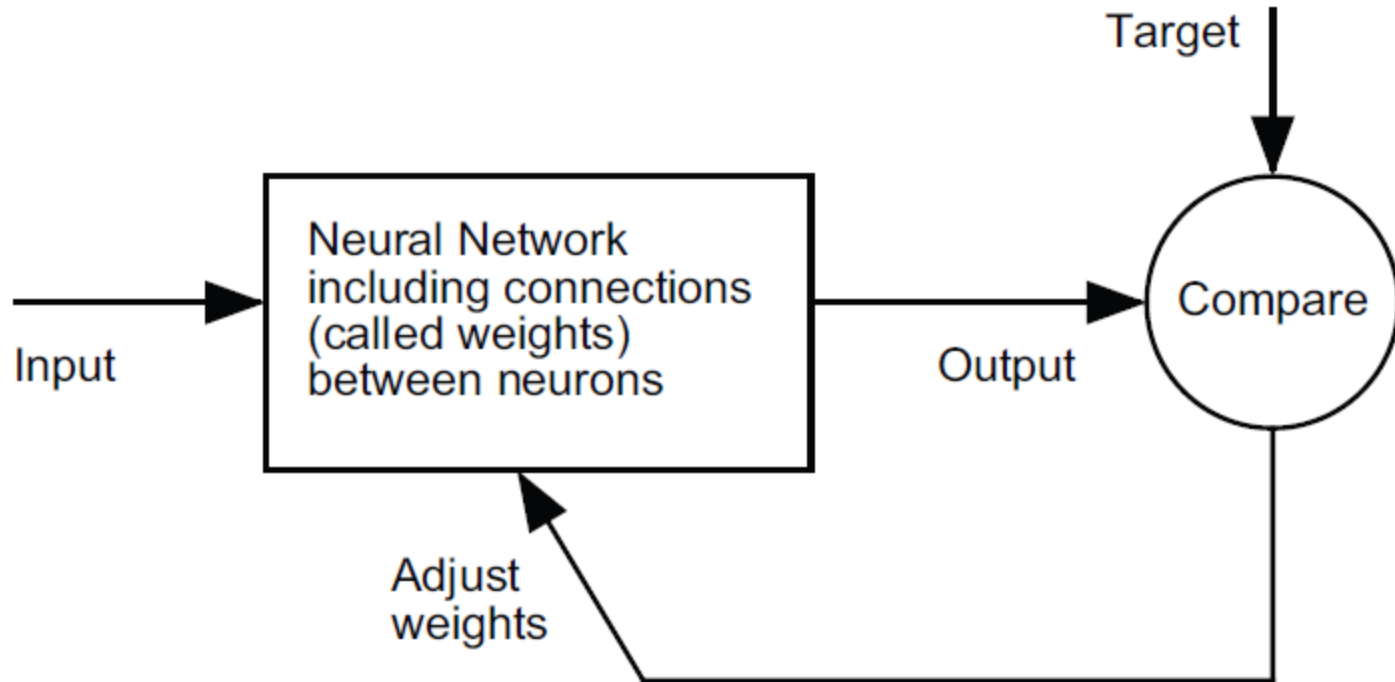
Mecanismo de Aprendizaje

- Los cambios en el proceso de aprendizaje se reducen a destrucción, modificación y creación de conexiones entre las neuronas.
- La creación de una conexión implica que el peso de la misma pasa a tener un valor distinto de cero.
- Una conexión se destruye cuando su valor pasa a ser cero.

Reglas de Aprendizaje

- Una regla de aprendizaje es un procedimiento para modificar los pesos y umbrales de una red.
- Este proceso también puede ser llamado *algoritmo de entrenamiento*.
- El aprendizaje puede ser *supervisado* o *no supervisado*.

Entrenamiento

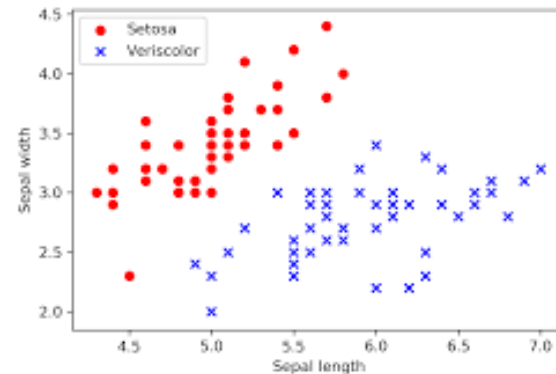
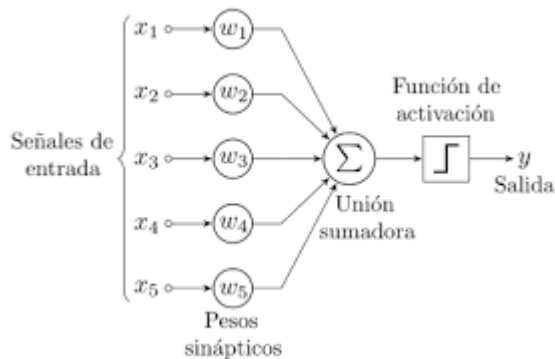




PERCEPTRÓN

La neurona perceptrón

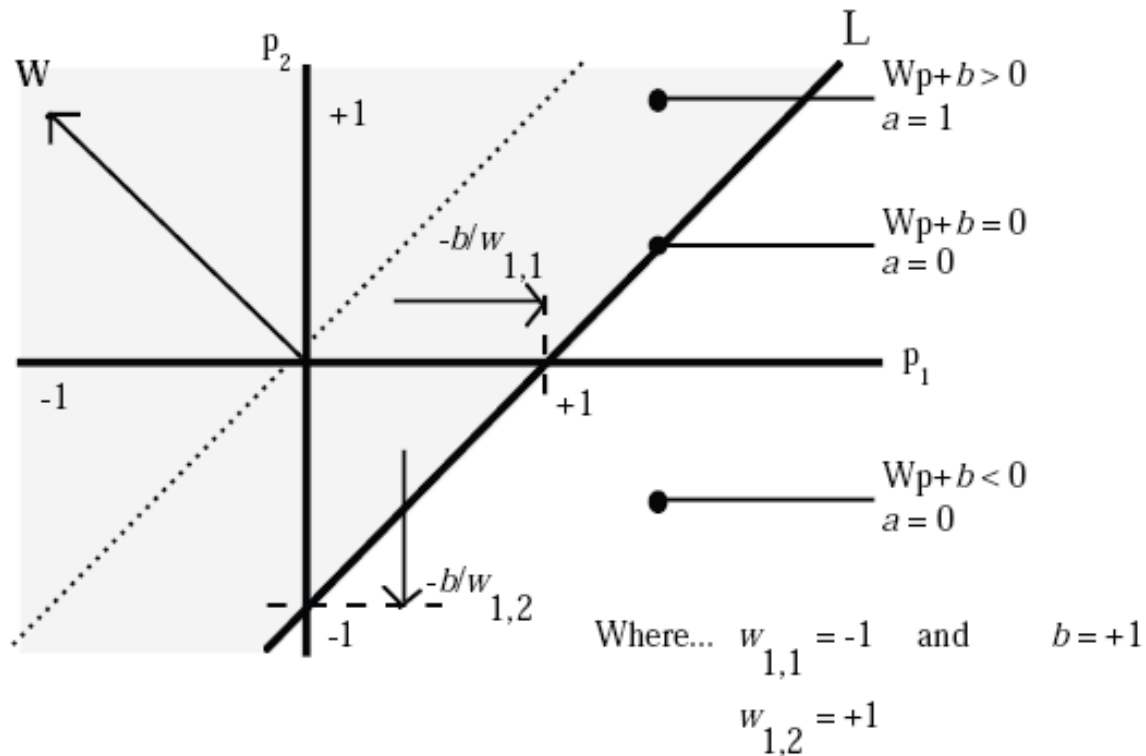
- Utiliza la función transferencia escalón



- Produce un 1 si la entrada a la función es mayor o igual que 0 y produce un 0 en caso contrario.
- Esto permite clasificar vectores de entrada dividiendo el espacio de entrada en dos regiones.

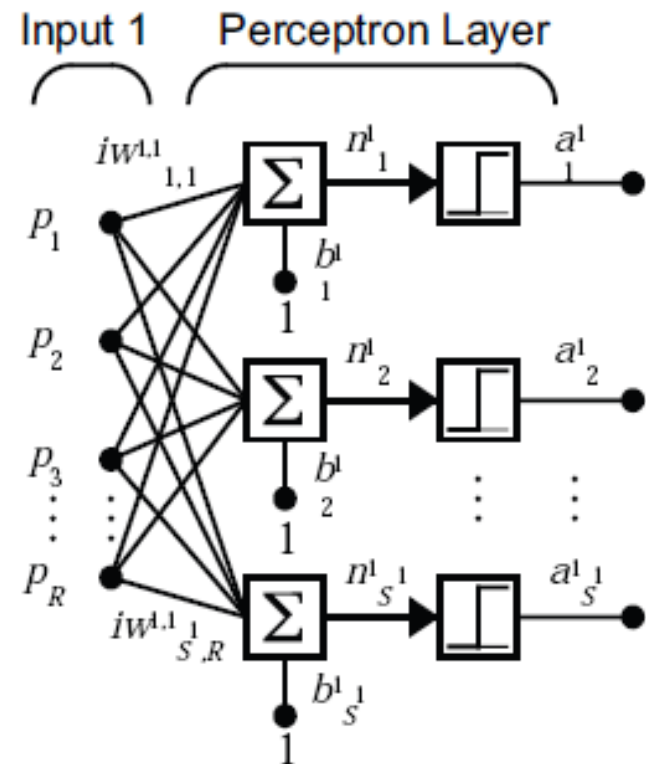
La neurona perceptrón

- Ejemplo:



Arquitectura

- La red Perceptrón consiste de una única capa oculta de S neuronas perceptrón conectadas a R entradas a través de conexiones con pesos $w_{i,j}$.
- Los índices i y j indican que $w_{i,j}$ es el peso de la conexión desde la j -ésima entrada hacia la i -ésima neurona.



Regla de Aprendizaje de la red Perceptrón I

- El perceptrón es entrenado con aprendizaje supervisado.
- Un conjunto de pares de E/S (patrón/target) representan el comportamiento deseado.
- El objetivo es reducir el error e , que es la diferencia entre el valor deseado t y el resultante a :

$$e = t - a$$

Regla de Aprendizaje de la red Perceptrón II

- Se puede probar que la regla de aprendizaje del perceptrón converge hacia una solución en un número finito de pasos.
- La estrategia consiste en modificar el vector \mathbf{w} haciendo que éste apunte hacia los patrones cuya salida es **1** y en contra de los patrones cuya salida es **0**.
- Hay una función que realiza el cálculo de la variación de \mathbf{w} .

Regla de Aprendizaje de la red Perceptrón III

- **CASO 1:** $a = t$ ($e = 0$). El vector \mathbf{w} no se altera.
- **CASO 2:** $a = 0$ y $t = 1$ ($e = 1$). El vector de entrada \mathbf{p} se suma a \mathbf{w} . Esto provoca que \mathbf{w} se cierre hacia \mathbf{p} , incrementando la posibilidad de que \mathbf{p} sea clasificado como **1** en el futuro.
- **CASO 3:** $a = 1$ y $t = 0$ ($e = -1$). A \mathbf{w} se le resta el vector de entrada \mathbf{p} . Esto provoca que \mathbf{w} se aleje de \mathbf{p} , incrementando la posibilidad de que \mathbf{p} sea clasificado como **0** en el futuro.

Regla de Aprendizaje de la red Perceptrón IV

- En consecuencia, la regla de aprendizaje es:

$$W_i \leftarrow W_i + \Delta W_i \text{ donde } \Delta W_i = \eta(t-a)p = \eta ep$$

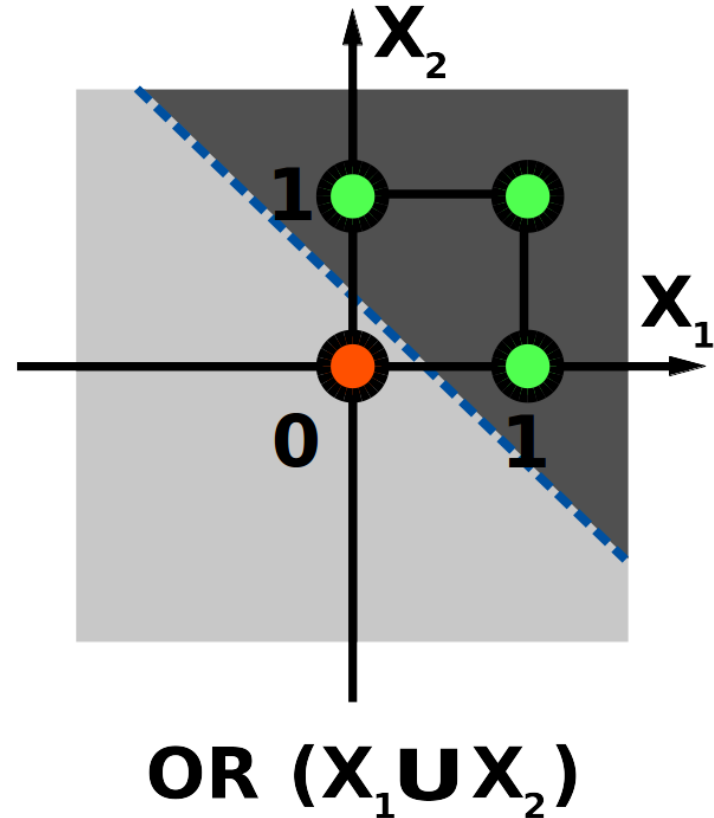
- η es un learning rate pequeño (para dar pasos chicos)
- Si consideramos el umbral como un peso cuya entrada siempre es 1 tenemos que los b se actualizan de igual forma

Regla de Aprendizaje de la red Perceptrón V

- El proceso de encontrar nuevos pesos se repite hasta que el error sea aceptable.
- Se garantiza la convergencia en un número finito de pasos para todos **los problemas que pueden ser resueltos por un perceptrón**
 - estos son todos los problemas de clasificación "linealmente separables".

Perceptrón - clasificación

- AND?
- OR?
- XOR?



Entrenamiento

- La función de entrenamiento (`train`) realiza una pasada por todas las entradas calculando para cada una la salida y el error.
- Con cada entrada al perceptrón se van actualizando sus pesos y se introduce la entrada siguiente.
- Se necesita una pasada posterior por todas las entradas para evaluar si se ha alcanzado el objetivo.
 - *Versión de entrenamiento estocástica*



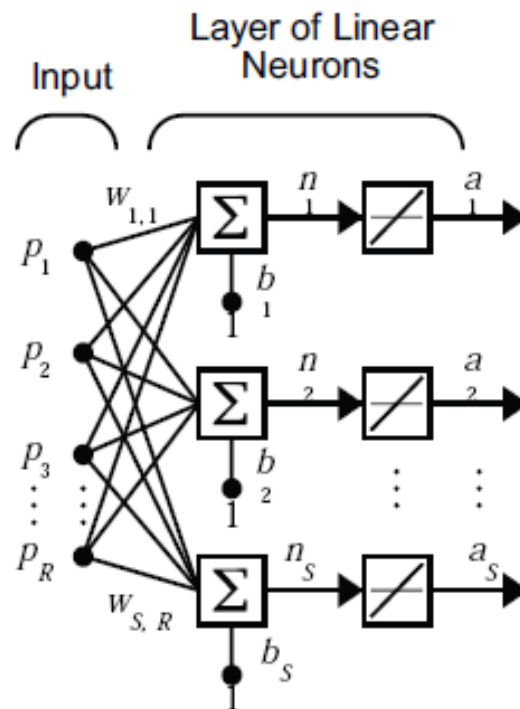
RED ADALINE

REDES LINEALES ADAPTATIVAS

Redes lineales

- Son similares al perceptrón pero su función transferencia es lineal.
- Al igual que el perceptrón estas redes resuelven problemas linealmente separables.
- En este caso se busca minimiza el error cuadrático medio sobre todos los patrones de entrenamiento.

Arquitectura de la ADaptive Linear NEtwork



$$a = \text{purelin}(Wp + b)$$

Error cuadrático medio (LMS)

- Sea un conjunto de Q entradas y targets:

$$\{p_1, t_1\}, \{p_2, t_2\}, \dots, \{p_Q, t_Q\}$$

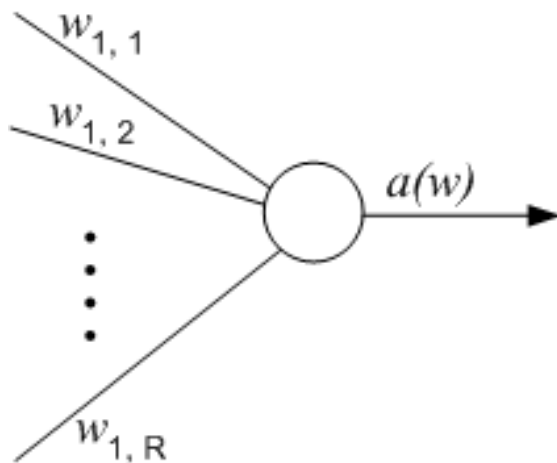
Se aplican las Q entradas calculando cada error como la diferencia entre el objetivo y la salida.

Se desea minimizar el promedio de los cuadrados de estos errores:

$$mse = \frac{1}{Q} \sum_{k=1}^Q e(k)^2 = \frac{1}{Q} \sum_{k=1}^Q (t(k) - a(k))^2$$

Algoritmo LMS (*least mean square*)

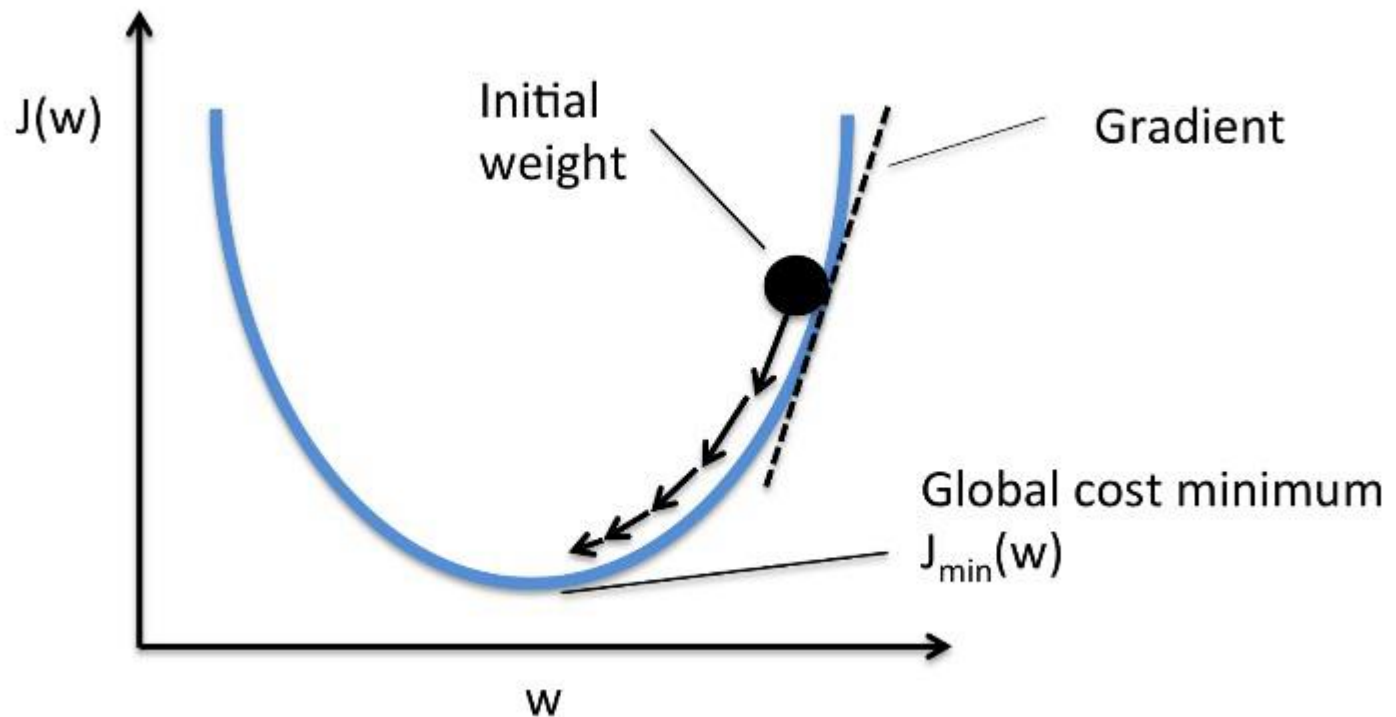
- También llamado algoritmo de Widrow-Hoff.
- La idea es ajustar los pesos para minimizar el *mse* (mean square error).
- ¿Cuáles deben ser los próximos pesos?



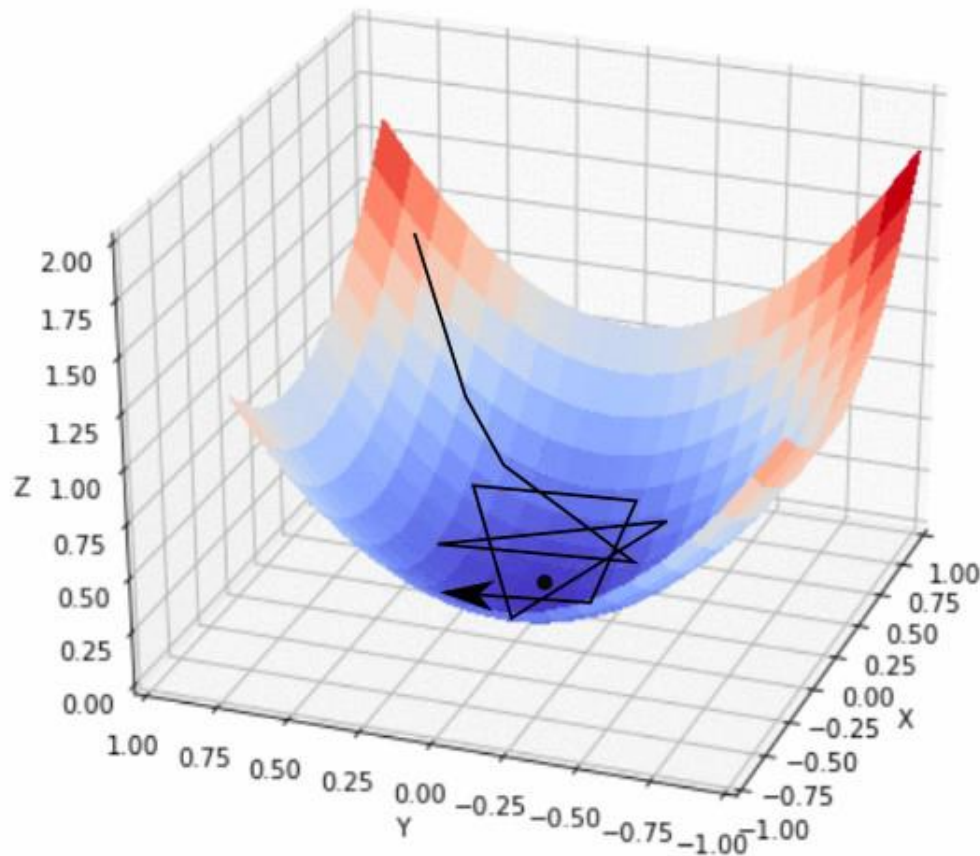
$$e^2(w) = (t - a(w))^2$$

El gradiente apunta hacia la
dirección de crecimiento
más rápido

Algoritmo LMS (*least mean square*)



Algoritmo LMS (*least mean square*)



Algoritmo LMS (*least mean square*)

$$\nabla E = \frac{1}{Q} \sum_{k=1}^Q \nabla e^2(w)$$

$$e^2(w) = (t - a(w))^2$$

$$W = W - \alpha \nabla E$$

α is a *learning rate*.

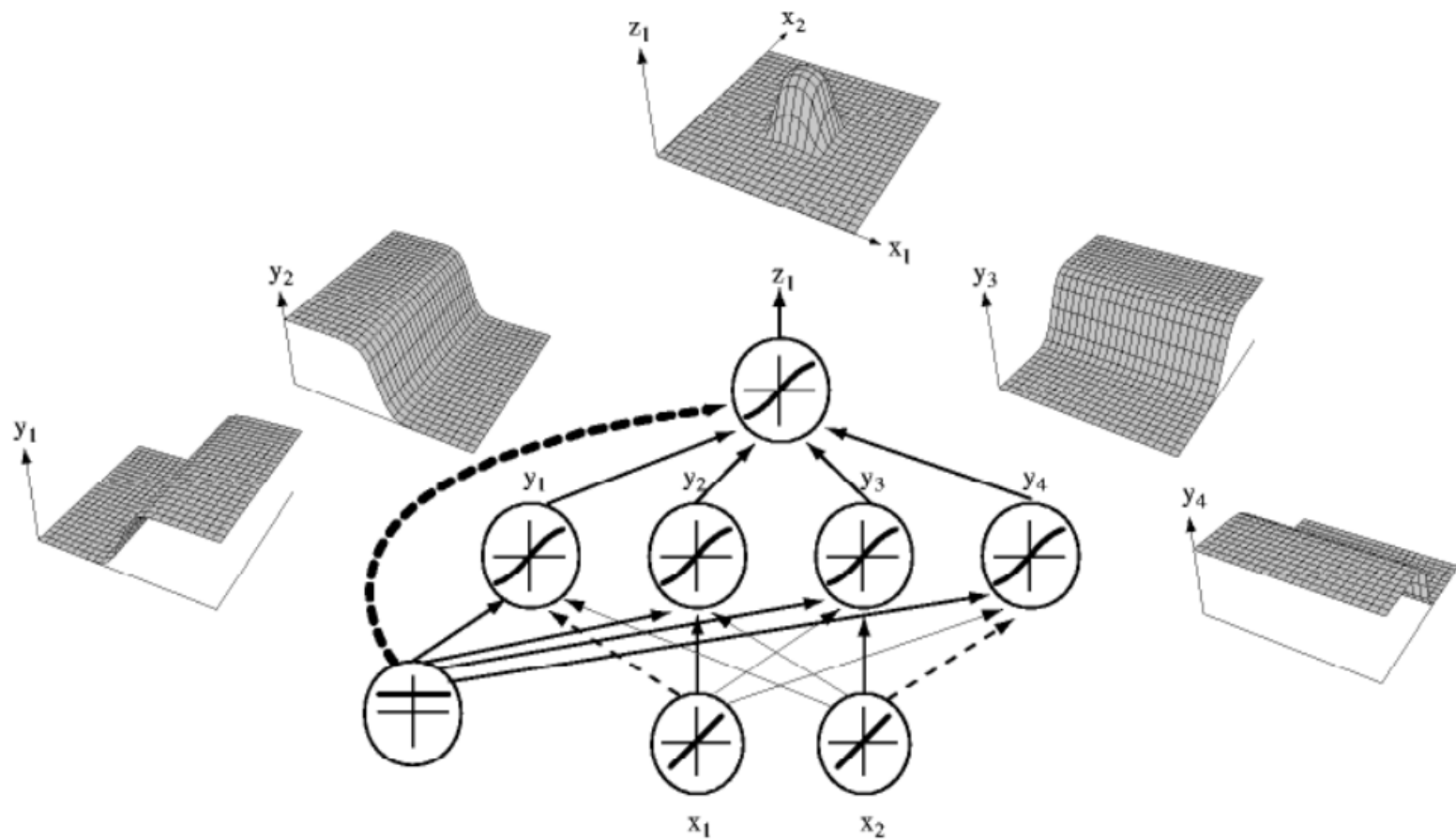


REDES BACKPROPAGATION

Redes Backpropagation

- Son una generalización de las redes lineales.
- Admiten múltiples capas y funciones derivables no lineales.
- Con una capa sigmoidea y una capa lineal son capaces de aproximar cualquier función con un número finito de discontinuidades.

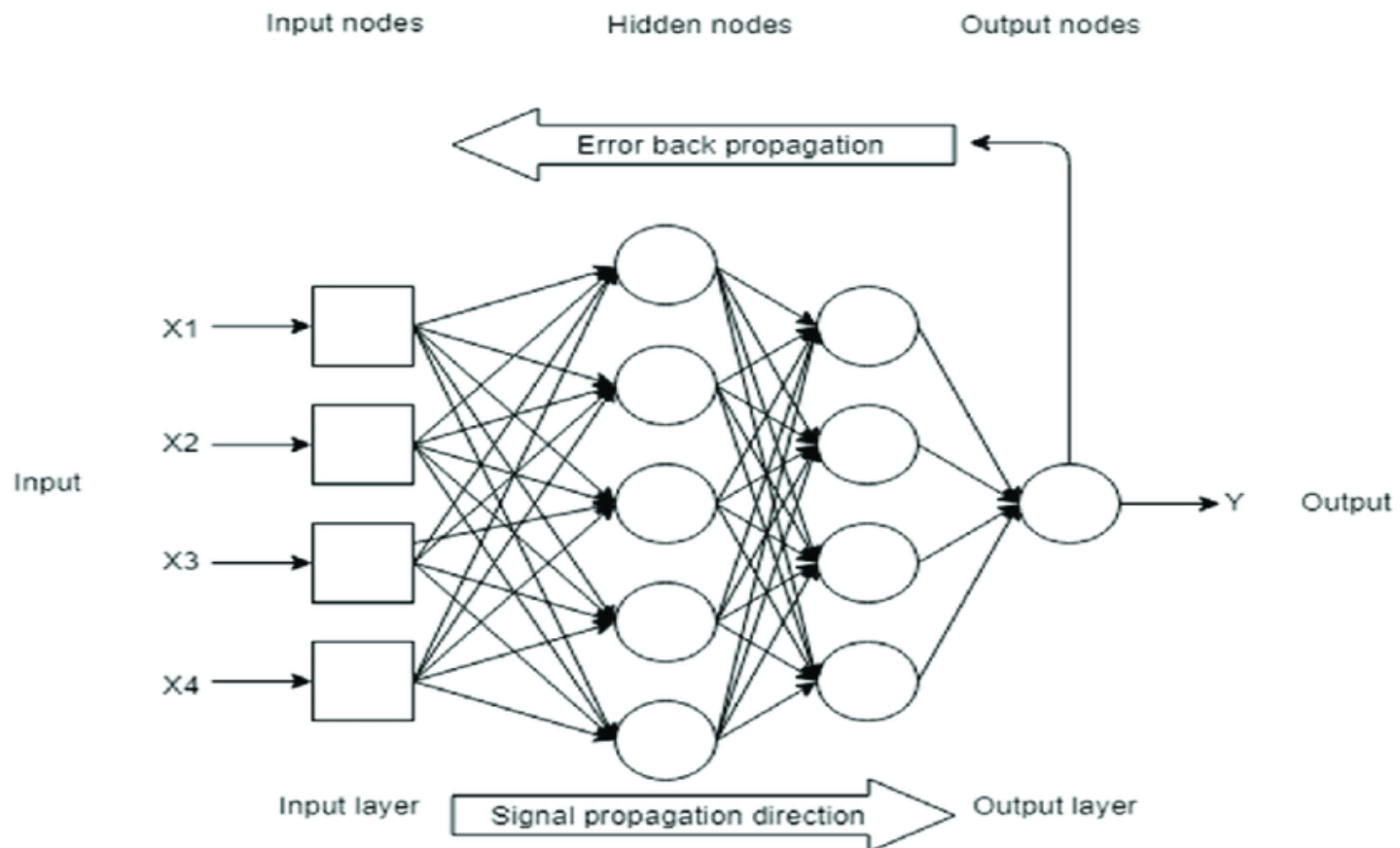
Ejemplo



Redes Backpropagation

- Utilizan la regla de aprendizaje de Widrow-Hoff con una serie de variaciones.
- El término backpropagation se refiere a cómo se aplica el cálculo del gradiente hacia capas anteriores (el nuevo peso deseado en la capa s es el target de la capa $s-1$).

Redes Backpropagation



Algoritmo de Backpropagation

- Pueden consultar Bibliografía Mitchel
- Slides: Pablo Granitto (pag 20-32)

https://drive.google.com/file/d/1inHd3a5G2K9L43rAx4jFVoTSG_KXyjN-/view

Redes Backpropagation

- Las múltiples capas con funciones no lineales permiten que estas redes puedan resolver problemas "no linealmente separables".
- Una capa de salida con función transferencia lineal permite que la red produzca valores más allá del $(-1, 1)$.
- Si se quiere restringir la salida, por ejemplo al $(0, 1)$ se puede utilizar la función sigmoidea para la capa de salida.

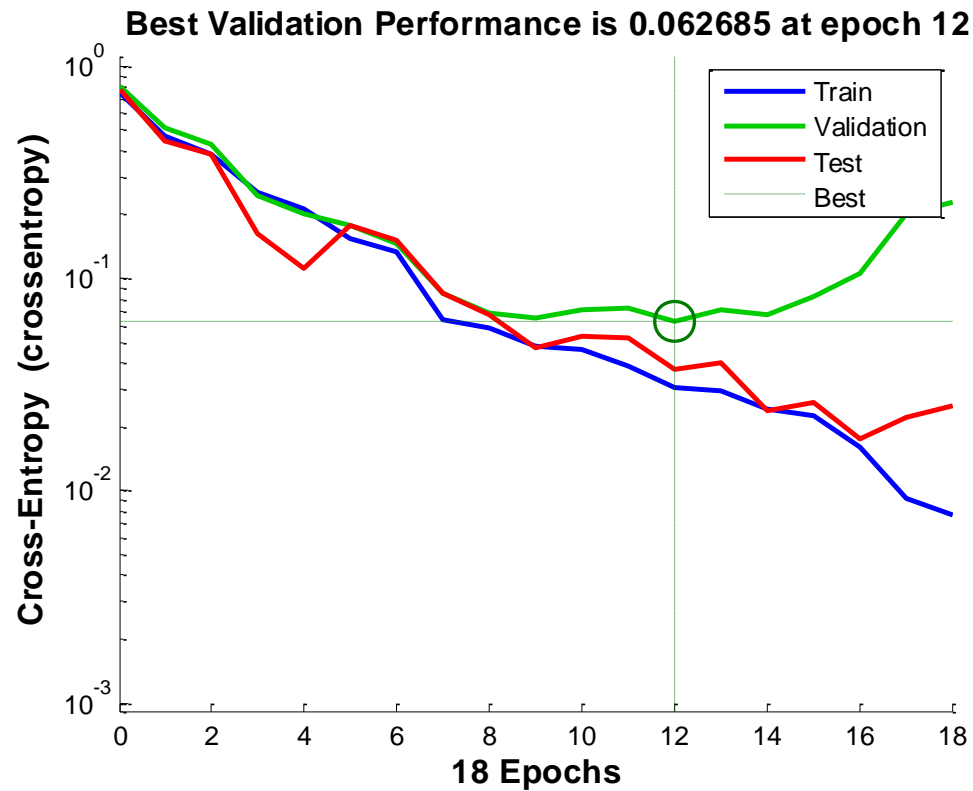
Reconocimiento de patrones

- Se puede entrenar redes para reconocimiento de patrones.
- Son redes feedforward que pueden classificar entradas en clases (conjuntos categóricos).
- Los datos objetivo (target) se presentan como vectores cuyos valores son todos cero excepto un 1 en la posición i , donde i es la clase que representan.

Conjuntos de entrenamiento, validación y test

- El dataset es dividido aleatoriamente en tres conjuntos:
 - 60% de los datos se utilizan para entrenamiento (ajuste de pesos y umbrales).
 - 20% de los datos se utilizan para validación (ajuste de otros parámetros - learning rate, etc).
 - 20% de los datos se utilizan para test (valoración del modelo obtenido).

Errores



Redes Neuronales - Resumen (I)

- Capacidad de representación: muy alta. Fronteras de representación no lineales.
- Legibilidad: ninguna. El modelo resultante consiste en vectores de pesos para las conexiones entre neuronas.
 - Hay avances en redes profundas...
- Tiempo de cómputo on-line: Rápido. Las operaciones son sumas y multiplicaciones.

Redes Neuronales - Resumen (II)

- Tiempo de cómputo off-line: muy lento. Gran cantidad de pesos a ajustar iterativamente.
- Parámetros a ajustar: muchos, se debe ajustar la estructura de la red, el tamaño del paso (*learning rate*), condiciones de terminación, etc.
- Robustez ante ejemplos de entrenamiento ruidosos: alta.
- Sobreajuste: puede ocurrir, se debe controlar.



SELECCIÓN DEL MODELO A UTILIZAR

Criterios de selección del modelo

- Dos decisiones fundamentales:
 - El tipo de modelo (árboles de decisión, redes neuronales, modelos probabilísticos, etc.)
 - El algoritmo utilizado para construir o ajustar el modelo a partir de las instancias de entrenamiento (existen varias maneras de construir árboles de decisión, varias maneras de construir redes neuronales, etc.)

Selección del modelo y/o algoritmo

- Capacidad de representación.
- Legibilidad.
- Tiempo de cómputo on-line.
- Tiempo de cómputo off-line.
- Dificultad de ajuste de parámetros.
- Robustez ante el ruido.
- Sobreajuste.
- Minimización del error.

Selección del modelo y/o algoritmo

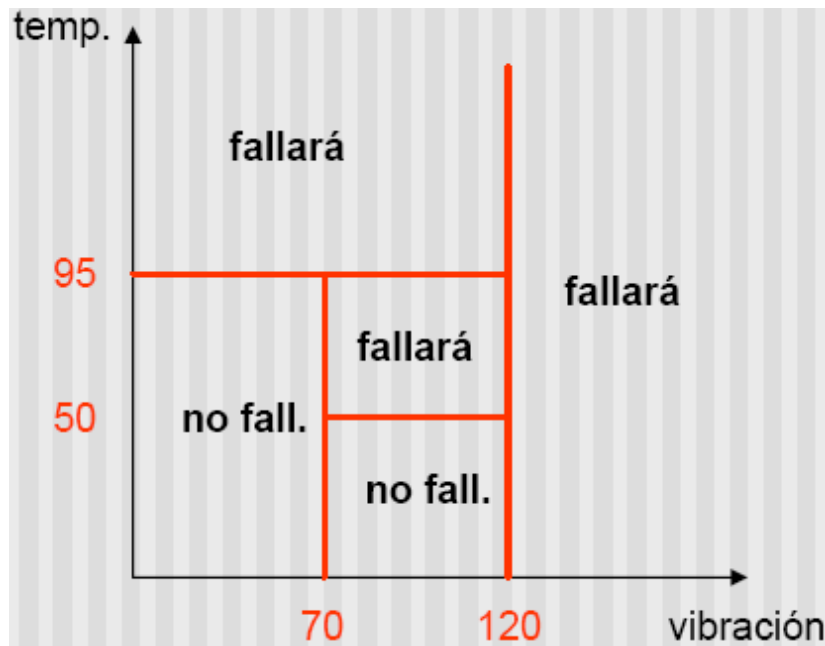
Capacidad de representación

- Relacionado con el tipo de **fronteras de decisión** que se pueden expresar.
- **Fronteras de decisión**: separación de clases distintas.
- Cada modelo crea diferentes fronteras.

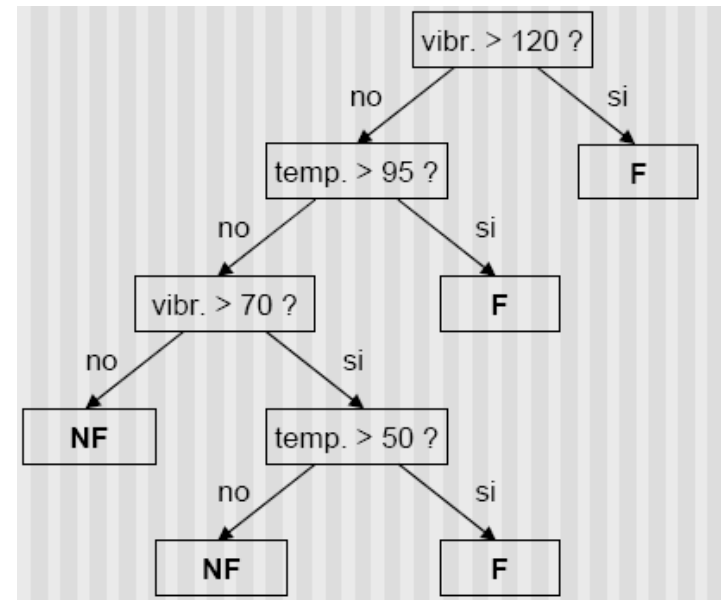
Selección del modelo y/o algoritmo

Capacidad de representación

- Ejemplo con sólo dos atributos:

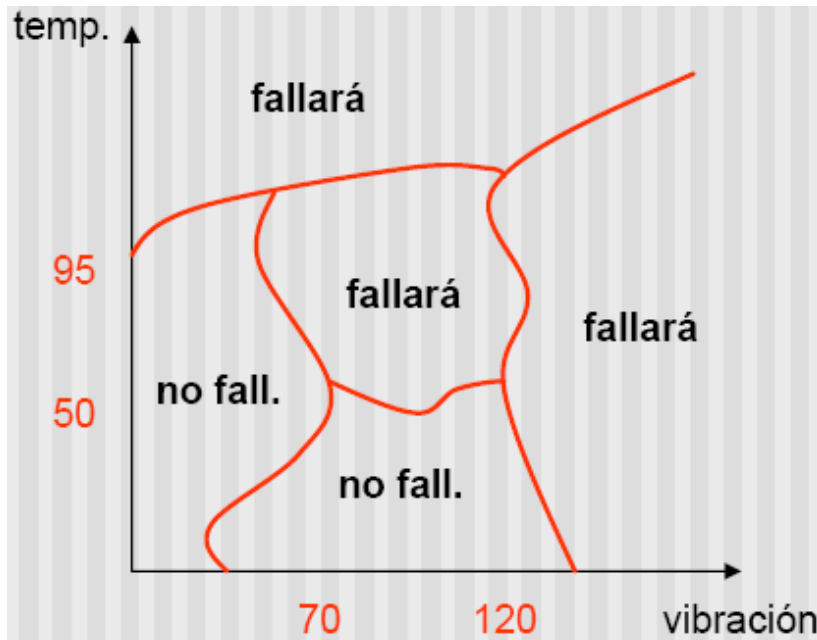


Árboles de decisión:
fronteras perpendiculares a
los ejes



Selección del modelo y/o algoritmo

Capacidad de representación (III)



Redes Neuronales (NN),
fronteras no lineales:

- Mayor capacidad de representación.
- Permiten representar conceptos más complejos que los árboles de decisión.

Selección del modelo y/o algoritmo

Legibilidad

- Capacidad de ser interpretado por un humano.
- **Árboles de decisión**: fáciles de entender e interpretar:
 - conjunto de reglas.
 - en los niveles más altos están los atributos más importantes.
- **Redes neuronales**: difíciles (o imposibles) de interpretar:
 - Avances en redes profundas, interpretación de capas/unidades

Selección del modelo y/o algoritmo

Legibilidad (II)

- Un modelo legible puede **ofrecer información** sobre el problema que se estudia (ej. indicar qué atributos afectan la probabilidad de fallo de una máquina y cómo).
- Un modelo no legible sólo puede ser usado como un **clasificador** (ej. permite predecir si una máquina fallará o no aplicando el modelo).

Selección del modelo y/o algoritmo

Tiempo de cómputo on-line (I)

- Es el tiempo necesario para clasificar una instancia:
 - **Árboles de decisión**: tiempo necesario para recorrer el árbol, evaluando las funciones lógicas de cada nodo.
 - **Redes neuronales**: tiempo necesario para realizar las operaciones (sumas, productos, sigmoides) incluidas en la red.

Selección del modelo y/o algoritmo

Tiempo de cómputo on-line (II)

- Este tiempo se consume cada vez que se debe clasificar una nueva instancia.
- Algunas aplicaciones requieren clasificar miles de instancias.
 - Ejemplo: clasificación de cada uno de los píxeles de una imagen área de un cultivo, río, ruta, etc.
 - Se requiere clasificar millones de instancias.
 - El tiempo de cómputo es muy importante.

Selección del modelo y/o algoritmo

Tiempo de cómputo off-line (I)

- Es el tiempo necesario para construir o ajustar el modelo a partir de los ejemplos de entrenamiento.
 - **Árboles de decisión**: tiempo necesario para elegir la estructura del árbol, los atributos a situar en cada nodo y la optimización mediante la poda.
 - **Redes neuronales**: tiempo necesario para ajustar los pesos de las conexiones (puede tomar valores muy grandes).

Selección del modelo y/o algoritmo

Tiempo de cómputo off-line

- Sólo se consume una vez, cuando mediante la utilización de los ejemplos de entrenamiento se genera y selecciona el resultado (modelo) más adecuado.
- Dependiendo de la aplicación no es un problema que el tiempo de cómputo off-line sea elevado (se deja una computadora procesando uno o tres días enteros).

Selección del modelo y/o algoritmo

Robustez ante el ruido

- Instancia de entrenamiento ruidosa:
 - etiquetada incorrectamente (ejemplo: una máquina que no falló, etiquetada como que sí falló).
 - algún atributo no está valorizado.
- Algunos algoritmos pueden funcionar adecuadamente aunque haya instancias ruidosas en el conjunto de entrenamiento (ej. árboles de decisión, redes neuronales).
- Otros algoritmos no ofrecen buenos resultados (ej. k-vecinos más cercanos).

Selección del modelo y/o algoritmo

Sobreajuste (*overfitting*)

- Problema muy común.
- El modelo está demasiado ajustado a las instancias y no funciona adecuadamente con nuevos casos.
- El modelo no es capaz de **generalizar**.
- Normalmente, fronteras de decisión muy complejas producen sobreajuste.

Bibliografía

- Machine Learning - Tom Mitchell – McGrawHill
- Introducción al Aprendizaje Automatizado, Pablo Granitto
<https://sites.google.com/site/aprendizajeautomatizadoou/nr/Inicio/apuntes>
- Curso de doctorado "Aprendizaje Automatizado y Data Mining" Grupo de Ingeniería de Sistemas y Automática (Universidad Miguel Hernández)
<http://isa.umh.es/asignaturas/aprendizaje/index.html>