Universidad
Francisco de Vitoria
UFV Madrid

Inteligencia Artificial II

Tema 3:

Aprendizaje Supervisado: Modelos Lineales

Departamento de Ingeniería Informática
Universidad Francisco de Vitoria

Álvaro José García Tejedor
Curso 2022/23

Objetivos del tema

- Ubicación
  - Bloque II: **COMPUTACION NEURONAL**
    - Tema 2: *Aprendizaje no supervisado: Aprendizaje competitivo*
    - Tema 3: ***Aprendizaje supervisado: Modelos lineales***

- Objetivos
  - Comprender que es predecir y las características de los predictores neuronales
  - Definir el aprendizaje supervisado
  - Entender el concepto de aprendizaje y sus asociados: entrenamiento, función de aprendizaje, error
  - Entender la Regla Delta: derivación y aplicación
  - Distinguir los modelos perceptrón y adaline y sus aplicaciones

Contenido

1. Introducción
2. El problema de la predicción
    1. La predicción
    2. Técnicas estadísticas
    3. Técnicas neuronales
3. Redes a las que hay que enseñar
4. Perceptrón
    1. Historia
    2. Arquitectura
    3. Procesamiento
    4. Aprendizaje
5. Adaline
    1. Aprendizaje
    2. Diferencias con el perceptrón

1. Introducción

- Algoritmos de aprendizaje supervisado:

  Aprender a reproducir comportamientos y razonamientos mediante estímulos

  - Basados en reglas de naturaleza global (los cambios producidos por el mecanismo de aprendizaje afectan a toda la matriz de pesos)
  - Los datos determinan la relación entrada-salida a la que la red debe de aproximarse)
  - Más amplio campo de aplicación:
    - Control
    - Predicción
    - Reconocimiento

- Solo son de interés los modelos no-lineales
- Modelos feedback y feedforward

1. Introducción
2. El problema de la predicción
    1. La predicción
    2. Técnicas estadísticas
    3. Técnicas neuronales
3. Redes a las que hay que enseñar
4. Perceptrón
    1. Historia
    2. Arquitectura
    3. Procesamiento
    4. Aprendizaje
5. Adaline
    1. Aprendizaje
    2. Diferencias con el perceptrón

2.1 La Predicción

- Proporcionar valores futuros de una variable con una cierta antelación (horizonte), en función de los valores históricos (serie temporal) que ha tenido dicha variable en el pasado
  - Necesario en problemas de planificación o control.
  - Instrumento fundamental en la toma de decisiones

- Concepto de ***Serie Temporal***

  Sucesión de medidas realizadas a intervalos regulares de tiempo

  Resultado de un proceso estocástico

- En el tratamiento de Series Temporales se busca
  - Estadísticos descriptivos (media, dispersión…)
  - Representaciones gráficas para analizar los datos
  - Construir un modelo de la serie que permita describirla y *predecir el siguiente valor*

2.1 La Predicción

- Definiciones
  - Proceso estocástico: conjunto de variables aleatorias (*sometidas al azar*) X_t cuya distribución (*evolución*) varía de acuerdo al tiempo t
    - t toma valores enteros o reales no negativos.
    - X_t toman valores en un conjunto que se denomina *espacio de estados*

    Un proceso estocástico es NO determinista

  - Proceso de Markov: El valor de la serie a tiempo ***t+1*** sólo depende del valor a tiempo ***t*** y no de los anteriores.

  - Proceso estacionario: Un proceso es estacionario en sentido amplio (o débilmente estacionario) cuando la media, varianza y covarianzas
    1. Existen,
    2. Son estables e independientes del tiempo

2.1 La Predicción

- Elementos de una serie temporal
  - Tendencia: movimiento que se mantiene durante el período de observación
    - Creciente o decreciente
    - Ajustarse a una función (lineal, cuadrática, exponencial, etc..)
  - Variación estacional: oscilaciones periódicas que dependen del período mínimo al cabo del cual aparecen patrones repetitivos en la serie temporal (ritmo anual, mensual, semanal e incluso diario)
  - Movimientos cíclicos: oscilaciones sobre la tendencia basadas en algún tipo de funciones trigonométricas (sinusoidales, etc...)
  - Proceso aleatorio: oscilaciones aleatorias que se superponen a los demás componentes resultado de las fluctuaciones estocásticas del sistema real. Extremadamente difíciles de predecir e incluso de modelizar
    - Ejemplo: la serie temporal de manchas solares

2.1 La Predicción

- Técnicas de predicción:
  - **Cuantitativas**: Estimaciones numéricas a partir de valores o propiedades que se conocen de la variable a predecir
    - Deterministas
      - Ajuste de la tendencia mediante extrapolación (técnicas de alisado)
      - Implica conocer las ecuaciones que rigen el problema (muy difícil)
      - Aunque se conozcan, la resolución puede ser muy compleja
    - Estadísticas/Estocásticas
      - Interpreta la secuencia de valores anteriores como una serie temporal
      - Intenta construir un modelo que se ajuste al problema
    - Neuronales
  - **Cualitativas**: Estimación subjetiva a partir de opiniones de expertos. Carecen de bases teóricas. Sirven para aglutinar opiniones
    - Metodologías Delphi

2.2 Técnicas estadísticas

- Métodos Tradicionales

  Sencillez de cálculo, bajo coste, utilidad práctica. Casos particulares de modelos más complejos

  - Métodos de regresión
    - Predicción = *f*(tiempo) + término de error
      - Lineal simple
      - Lineal general (regresión múltiple)
    - Se emplean como aproximación inicial
  - Suavizado Exponencial y Medias Móviles
    - Minimizan el MSE
    - Media Móvil Simple: el siguiente valor es la media de los N valores anteriores

      ➔ mejora: Suavizado Exponencial Simple
    - Media Móvil Lineal: segunda media móvil calculada sobre la anterior

      ➔ mejora: Suavizado Exponencial Lineal

2.2 Técnicas estadísticas

- Métodos de Descomposición

*Identifican las componentes deterministas de la serie (**t**endencia, **e**stacionalidad, **p**eriodicidad) descartando la componente aleatoria que se engloba en un término de error*

$$X_t = f\,(T_t, E_t, P_t, \text{Error})$$

- Proceso:
  1. Calcular media móvil de longitud N = estacionalidad (se elimina Estacionalidad y Error)
  2. Sustraer la media móvil obtenida a la serie original. Se obtiene la Tendencia y la Periodicidad
  3. Asilar la estacionalidad calculando la media de cada período estacional para ajustar E_t
  4. Identificar Tendencia (lineal, exponencial…) y calcular su valor T_t
  5. Separar T de la serie obtenida en (2) para obtener P_t
  6. Separar todos los componentes para obtener el Error

2.2 Técnicas estadísticas

- Modelos Box-Jenkins (1976)
  - Análisis univariante de series temporales
    - Utiliza como información la propia historia de la serie
    - Hipótesis básica: el comportamiento pasado se mantendrá en el futuro
    - Modelos probabilísticos lineales
  - Complejas de utilizar e imponen restricciones a las series
    - Las series deben de ser débilmente estacionarias y ergódicas
    - Si no lo son, se deben transformar mediante complejas operaciones matemáticas
  - **Modelos**
    - AR(p) *Auto Regressive* (Auto Regresión) de orden p (números de términos del pasado)
      - $\mathbf{Y_t}$ es explicada por las observaciones de **Y** en períodos anteriores (combinaciones lineales en le mejor caso), más un término de error.
      - Orden: números de términos del pasado

2.2 Técnicas estadísticas

- MA(q) *Moving Average* (Media Móvil) de orden q
  - $\mathbf{Y_t}$ es explicada en función de un término independiente y una sucesión de errores correspondientes a períodos precedentes
- ARMA(p,q) *Auto Regressive Moving Average*
  - combinación de los dos anteriores
- ARIMA(p,q,d) *Modelos Autorregresivos Integrados de Medias Móviles*
- SARIMA(p,q,d) *Modelos Estacionarios Autorregresivos Integrados de Medias Móviles*

2.2 Técnicas estadísticas

- Limitaciones de las técnicas estadísticas
  - Necesidad de conocer a priori
    - La serie temporal en si misma
    - Las relaciones funcionales entre las variables del problema

    Obliga a establecer un modelo estadístico de comportamiento de la serie y del proceso del mundo real que representa
  - Falta de precisión en las predicciones
  - Inestabilidad numérica de los modelos ante factores externos no considerados
  - Necesidad de suponer estacionariedad en la serie temporal
    - Si la serie no es estacionaria, hay que realizar ciertas modificaciones que distorsionan el modelo
  - No existe metodología comúnmente aceptada como la mejor (continuo desarrollo de nuevos modelos)

2.2 Técnicas estadísticas

- Los métodos más usados son ARMA y ARIMA
  - Ampliamente probados en el caso de problemas univariantes, pero no en multivariantes
  - A medida que se predicen situaciones más complejas las predicciones comienzan a desviarse de la realidad,
    - bien por el tipo de relación entre las variables explicativas
    - bien por la complejidad del dominio en sí
  - Precisión medida en base al Error Cuadrático Medio (Mean Square Error o MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(o_i - x_i)^2$$

2.3 Técnicas neuronales

- Primeros trabajos ➔ **Lapedes** y **Farber** (1987 en Los Alamos National Laboratory)
  - Arquitectura PMC
  - Aprendizaje por algoritmo de backpropagation
  - Fiunción de activación sigmoide
  - Entradas:
    - *n* observaciones anteriores de la serie x(*t*), x(*t*-*d*), …, x(*t*-*nd*)
  - Predicción:
    - valor de la serie en x(*t*+*p*) (*p* es el horizonte de predicción)
  - Esquema de trabajo: *ventana deslizante*
  - Válido para series temporales *multivariantes* y *no lineales*

2.3 Técnicas neuronales

- Otra posibilidad: predicción mediante redes neuronales recurrentes tipo RNN (Feedback)
  - Comportamiento interesante de reconocimiento de patrones temporales (igual que SOM reconoce patrones espaciales)
  - Mejoradas con las redes LSTM.
  - En los últimos años, gran desarrollo de modelos especializados en tratamiento de series temporales.

- Ventajas
  - Mejores niveles de precisión alcanzados
  - Mayores horizontes de predicción, con costes de desarrollo menores

2.3 Técnicas neuronales

- Características diferenciales frente a las estadísticas
  - No es necesario conocer la estructura del modelo. Solo disponer de suficientes datos históricos
  - No requieren de expertos altamente especializados.
  - Su desarrollo puede ser incremental,
    - Facilidad para reconfigurar el modelo (añadiendo nuevas neuronas en la capa de entrada)
    - Mejora continuada de las aplicaciones a fin de adaptarlas a condiciones cambiantes reentrenando la red con datos más recientes.
  - Se obtienen buenas predicciones incluso aunque no todas las variables explicativas hayan sido consideradas por la red
  - Uso indistinto de variables correladas o incorreladas: *la red descarta las innecesarias*
  - Modelo no lineal: dominios no lineales
  - Válidas para series temporales multivariantes

1. Introducción
2. El problema de la predicción
    1. La predicción
    2. Técnicas estadísticas
    3. Técnicas neuronales
3. Redes a las que hay que enseñar
4. Perceptrón
    1. Historia
    2. Arquitectura
    3. Procesamiento
    4. Aprendizaje
5. Adaline
    1. Aprendizaje
    2. Diferencias con el perceptrón

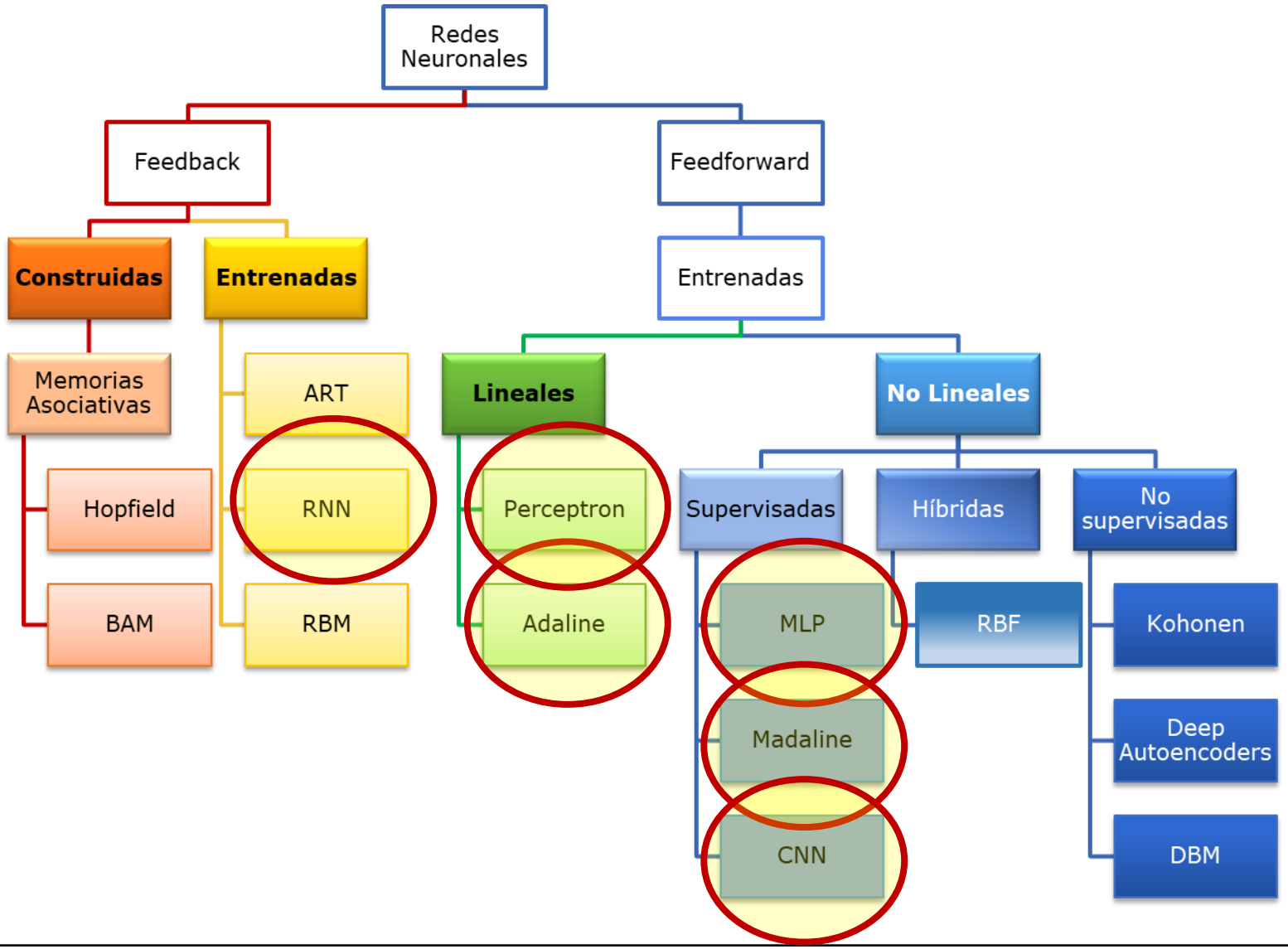
3. Redes a las que hay que enseñar

Validas para la predicción …

… y para otras tareas

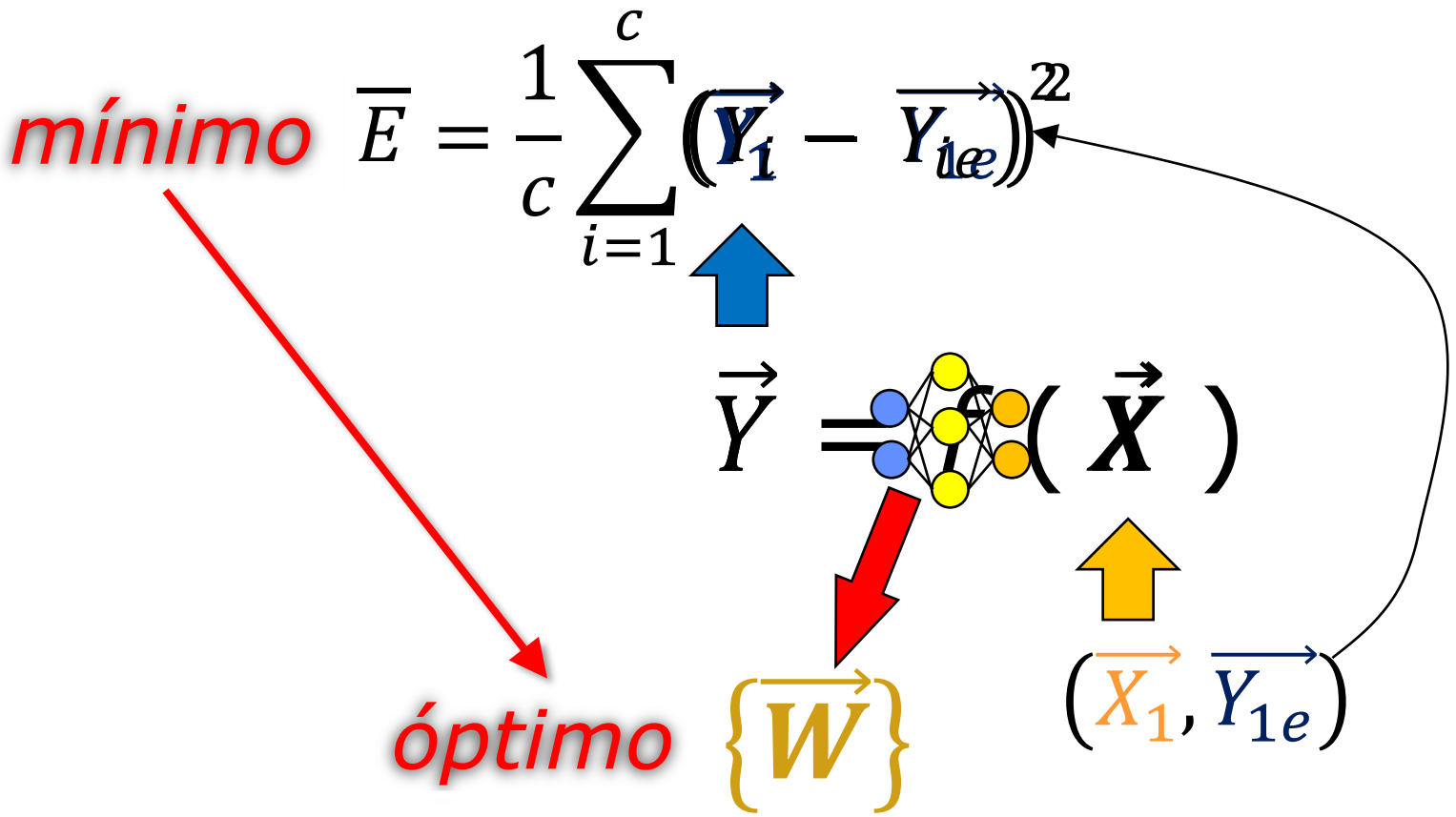
- Modelos FeedBack
  - Redes construidas
    - Memorias Asociativas Bidireccionales (BAM)
  - Redes entrenadas
    - Redes recurrentes (RNN)
- Modelos FeedForward
  - Modelos lineales
    - Perceptrón
    - Adaline
  - Modelos no lineales
    - Perceptrón Multicapa (PMC)
      - Modelo no lineal basado en el Perceptron de Rosenblatt
    - Madaline
      - Modelo no lineal desarrollado a partir de la arquitectura ADALINE

3. Redes a las que hay que enseñar



3. Redes a las que hay que enseñar

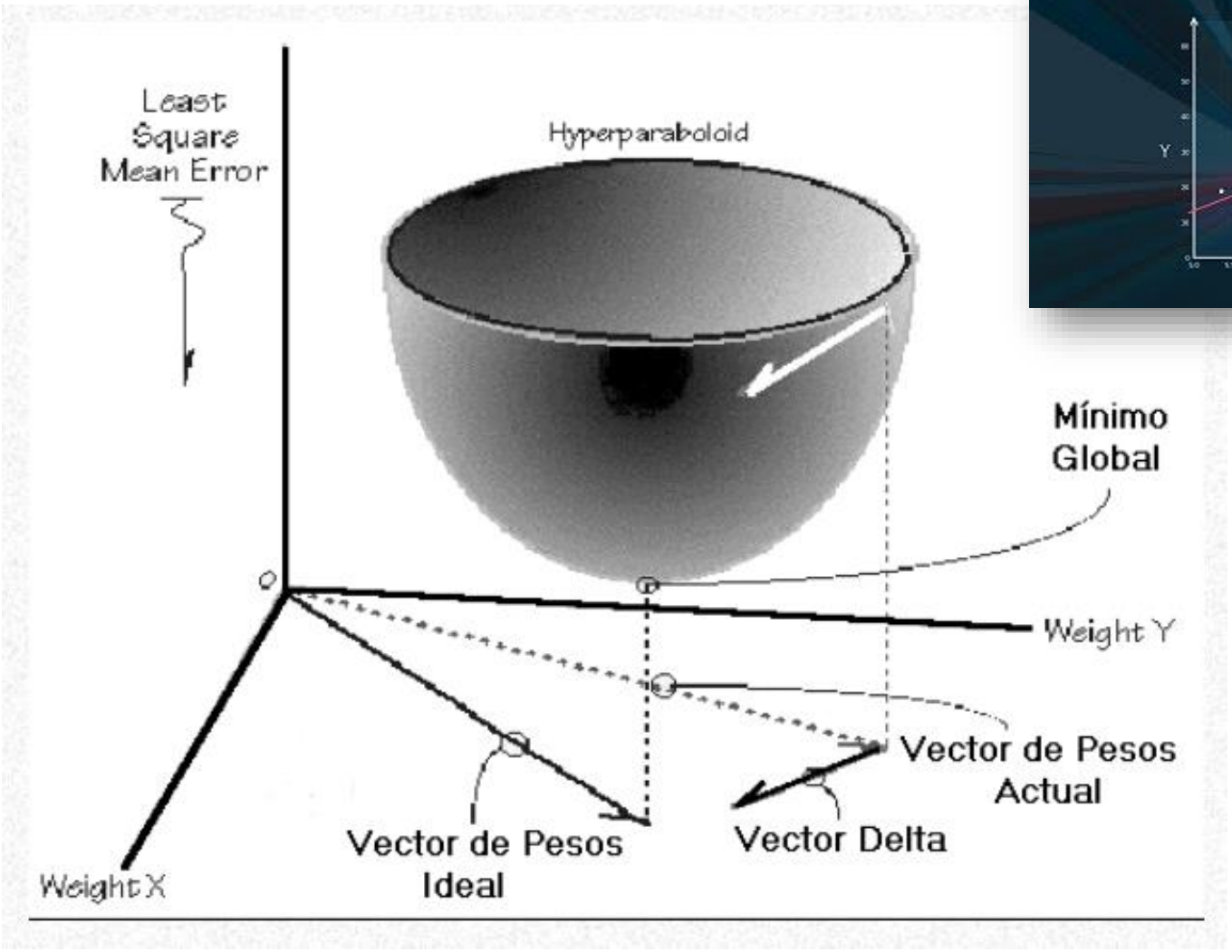
¿Por qué es necesario el aprendizaje?

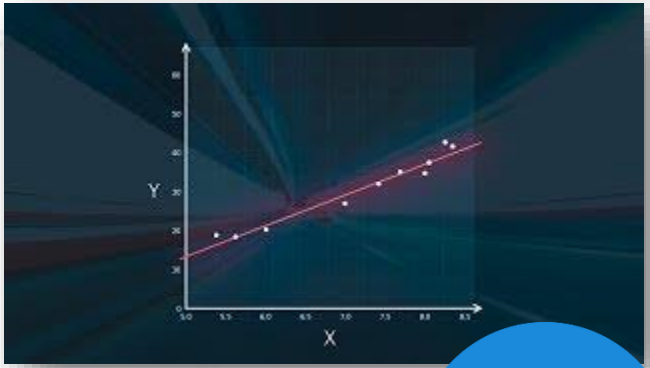


3. Redes a las que hay que enseñar

- La eficiencia de una red neuronal supervisada depende del valor de los parámetros (*matriz de pesos*) que determinan el ajuste entre las entradas y las salidas
- Es difícil determinar con precisión tantos parámetros
- Proceso para encontrar la mejor matriz de pesos
  - En una red que ajusta entradas/salidas sobre casos de un problema real se calcula el error producido en ese ajuste
  - Se busca la matriz de pesos que *minimiza el error* que la red comete en el proceso
  - Esa búsqueda en el espacio de pesos puede ser
    - Búsqueda exhaustiva
    - Búsqueda aleatoria
    - Búsqueda dirigida ➔ *Método de Aprendizaje*
      - Método de mínimos cuadrados
      - Descenso de gradiente
      - Algoritmos genéticos

3. Redes a las que hay que enseñar





3. Redes a las que hay que enseñar

- *Aprendizaje:*

Proceso dirigido de obtención de la mejor matriz de pesos en una red neuronal

- Para redes neuronales feedforward, el aprendizaje supervisado se define como:
  - Dado un conjunto de entrenamiento

$$C = \left\{\left(\overrightarrow{X^p}, \overrightarrow{Y^p}\right) \middle| \overrightarrow{X^p} \in R^N, \overrightarrow{Y^p} \in R^M, p = 1,2,\dots,c\right\}$$

  - Y una red neuronal de la que sólo conocemos su estructura (capas y número de unidades en cada capa) y que implementa una función de R^N en R^M

$$R^N \xrightarrow{f} R^M$$

  - Encontrar un conjunto de pesos $\{W_{ij}\}$ tal que la función de R^N en R^M se ajuste lo mejor posible a los ejemplos del conjunto de entrenamiento

3. Redes a las que hay que enseñar

- *Entrenamiento*

Proceso iterativo que busca minimizar el error en el conjunto de entrenamiento

- Precisa un conjunto de entrenamiento
  - del que se conozca la salida esperada
  - suficientemente amplio para que cubra el espacio muestral
- En cada paso del entrenamiento
  - Se calcula el error
  - Se modifica consecuentemente la matriz de pesos
- El entrenamiento acaba cuando se cumple la condición de terminación
  - Cuando una pasada completa del conjunto de entrenamiento ocurre sin error
  - Cuando el error alcanza un valor predeterminado

3. Redes a las que hay que enseñar

- *Ley de aprendizaje*

  Formulación matemática que ajuste la matriz de pesos en función de los pares entrada/salida con que se entrena la red

- Reglas para modelos lineales:
  - Ley de aprendizaje de *Rosemblatt* basada en la *Ley de Hebb* (Perceptrón)
  - Ley de aprendizaje *Widrow-Hoff* o *Regla Delta* basada en el *Descenso de Gradiente* (Adaline)

- La más importante: algoritmo de retropropagación del error (**backpropagation**) o *Regla Delta Generalizada*
  - Perceptrón Multicapa: Redes multicapas no lineales

- El aprendizaje correcto permite que la red *generalice*
  - Una entrada nueva (nunca vista) genera una salida correcta

3. Redes a las que hay que enseñar

Aprendizaje = Entrenamiento + Ley de Aprendizaje

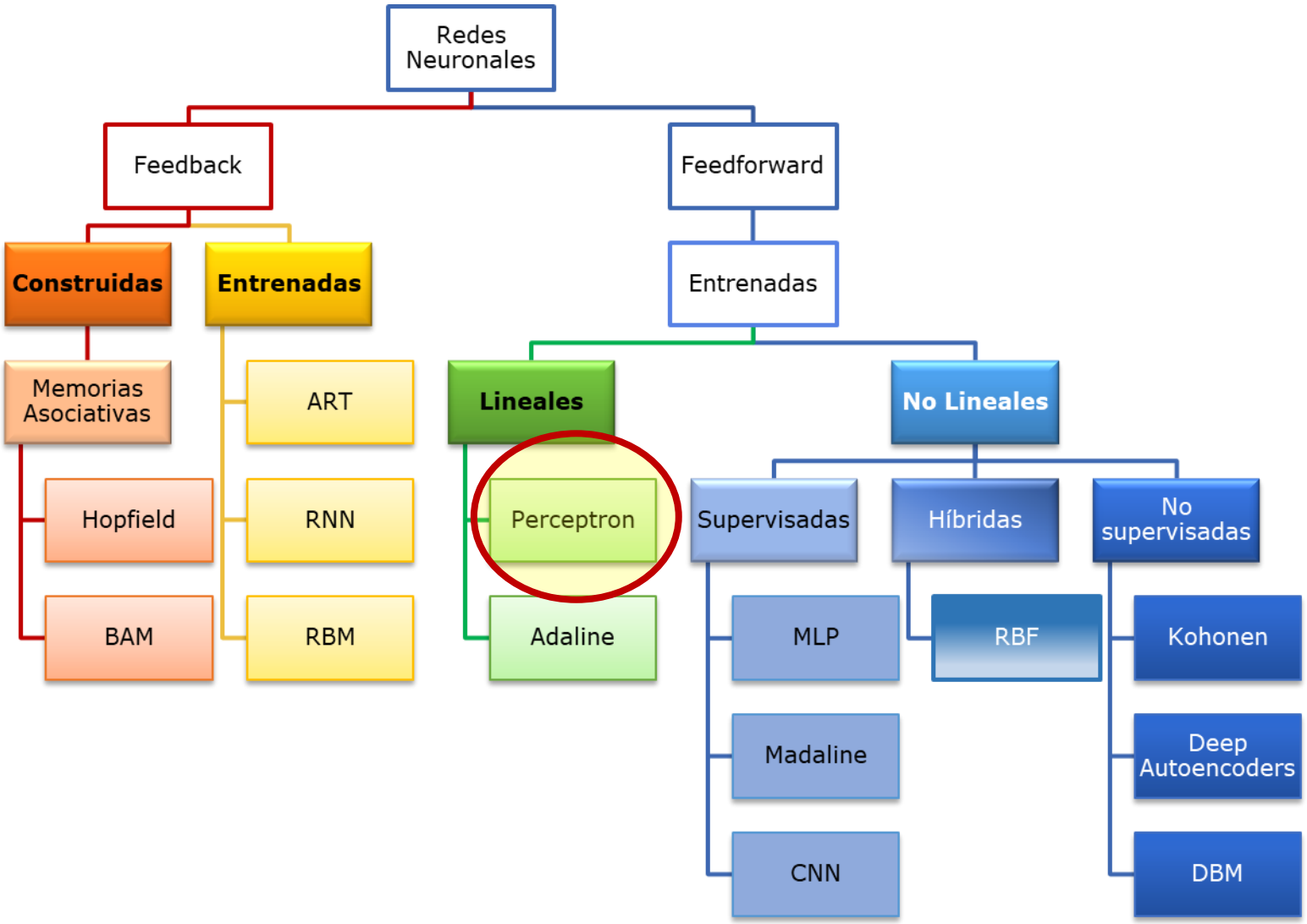
1. Se inicializa la matriz de pesos al azar $\vec{W}_t$
2. Se presenta un patrón (entrada y salida deseada)
3. Se computa la salida obtenida por la red.
4. Se calcula el error cometido para dicho patrón.
5. Se determina la variación de los pesos: descenso por la pendiente más pronunciada de la función de error.
6. Se modifican los pesos para movernos un poco más abajo en la superficie mediante la Regla Delta correspondiente.

$$\vec{W}_{t+1}=\vec{W}_t + \Delta\vec{W}_t$$

7. Se repiten los pasos del 2 al 6 para todos los patrones de entrenamiento.
8. Si el error es un valor reducido aceptable, termina el proceso. Si no, se vuelve al paso 2.

1. Introducción
2. El problema de la predicción
    1. La predicción
    2. Técnicas estadísticas
    3. Técnicas neuronales
3. Redes a las que hay que enseñar
4. Perceptrón
    1. Historia
    2. Arquitectura
    3. Procesamiento
    4. Aprendizaje
5. Adaline
    1. Aprendizaje
    2. Diferencias con el perceptrón

4. Perceptrón

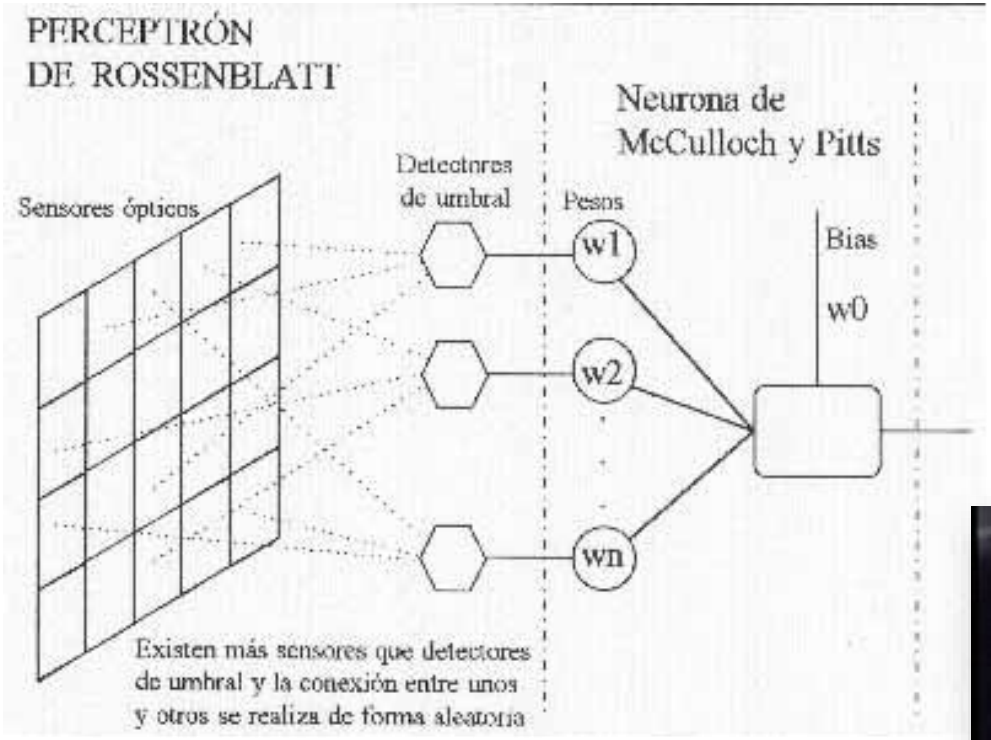


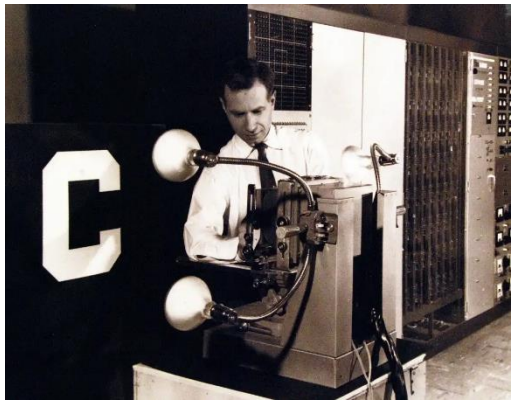
4.1 Historia

- [1959] Frank Rosenblatt: Perceptrón (modeliza la retina)
  - Usa una neurona formal binaria tipo McCulloch y Pitts (0,1)
  - Pesos sinápticos ajustables
  - Aprendizaje: Ley de Hebb para entrenar el Perceptrón.
  - Función de activación umbral
  - El aprendizaje intenta minimizar el número de casos mal resueltos

- Sistema constituido por
  - Unidades Sensoriales (S) (Input layer) que constituyen el modelo de retina. Meros fotorreceptores.
  - Unidades de Asociación (A), conectadas con unidades S, consigo mismas. No son neuronas.
  - Unidades de Respuesta (R), neuronas con umbral interconectadas entre si.

4.1 Historia

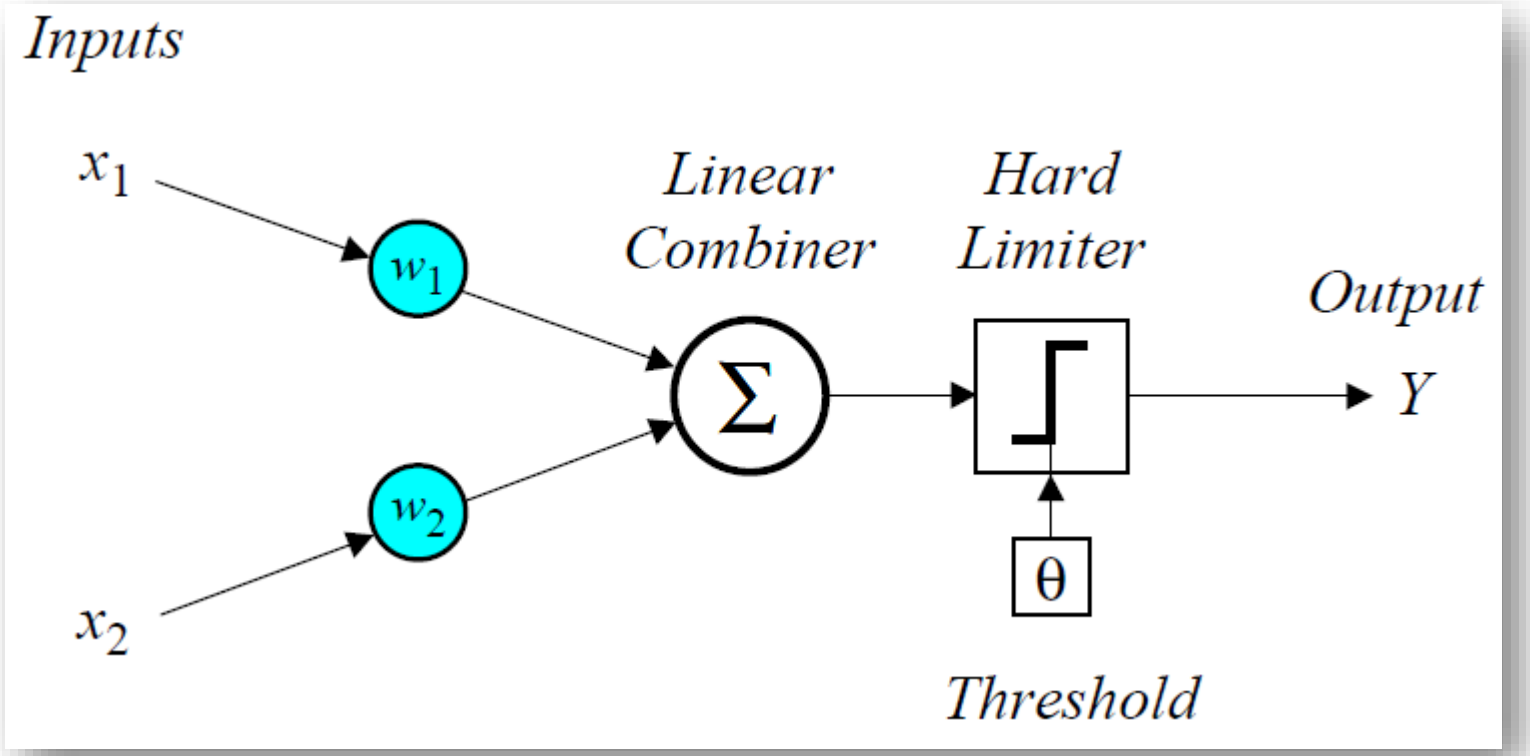




- Importante
  - No son tres capas, ¡sino 1!
  - Las conexiones iniciales son al azar
  - El objetivo es activar la neurona correspondiente a cada patron de entrada.
  - Las neuronas compiten entre sí para clasificar las entradas.

4.2 Arquitectura

- La estructura de la neurona tipo perceptrón (con dos entradas) es:



4.2 Arquitectura

- Donde
  - x_1 y x_2 son las entradas
  - Y es la salida
  - w_1 y w_2 son los pesos de las entradas 1 y 2 respectivamente
  - El *Combinador Lineal* es la suma ponderada de las entradas:

$$S = \sum_{i=1}^{n} w_i x_i$$

  - La Función de Activación

$$Y = F(S, \theta) = f\left(\sum_{i=1}^{n} w_i x_i - \theta\right)$$

    es un *HardLimiter* (función escalón) cuyos valores son:

$$F(S, \theta) = \begin{cases} +1 & si\ S \geq \theta \\ 0 & si\ S < \theta \end{cases}$$

  - θ es el *umbral* que equivale a un peso ficticio no conectado a ninguna entrada y que se resta del valor de entrada a la neurona

4.3 Procesamiento

- Podemos obtener una expresión general del comportamiento de la neurona
  - Haciendo que el índice *i* comience en 0
  - Definiendo $w_o = \theta$ y $x_0 = -1$ (valor constante)

  Con lo que obtenemos

$$Y = F(S, \theta) = f\left(\sum_{i=0}^{n} w_i x_i\right)$$

  Cuyos valores son

$$F(S, \theta) = \begin{cases} +1 \;\; si \sum_{i=0}^{n} w_i x_i \geq 0 \;\rightarrow \sum_{i=1}^{n} w_i x_i - \theta \geq 0 \\ 0 \quad si \sum_{i=0}^{n} w_i x_i < 0 \;\rightarrow \sum_{i=1}^{n} w_i x_i - \theta < 0 \end{cases}$$

4.3 Procesamiento

- El Perceptrón funciona como un CLASIFICADOR
  - Clasifica las entradas $x_1, x_2, \ldots, x_n$, linealmente separables, en dos clases A_1, A_2
  - La separación es un hiperplano cuya ecuación se obtiene para el valor 0 de la Función de Activación

$$\sum_{i=1}^{n} w_i x_i - \theta = 0$$

- Para un perceptrón de dos entradas la ecuación general es

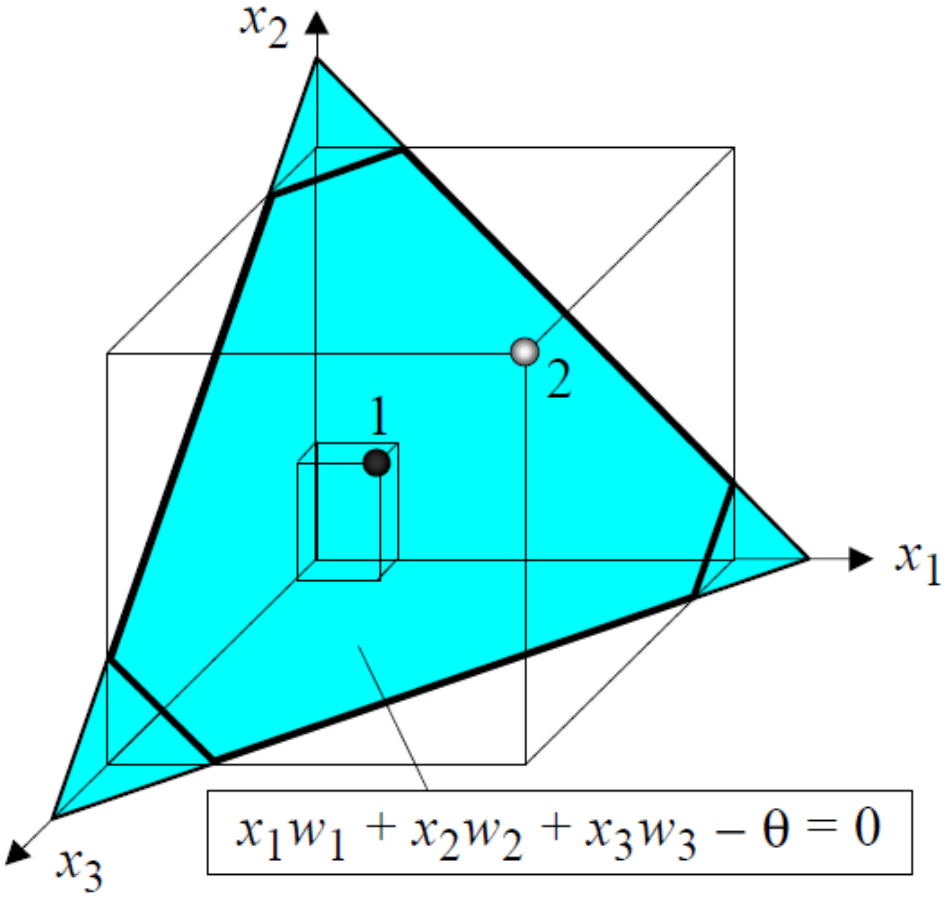
$$w_1 x_1 + w_2 x_2 - \theta = 0$$

  - La forma principal de esta ecuación es

$$x_2 = -\frac{w_1}{w_2} x_1 + \frac{\theta}{w_2}$$

  - los valores de w_1 y w_2 se encuentran por aprendizaje a partir de los datos de entrenamiento

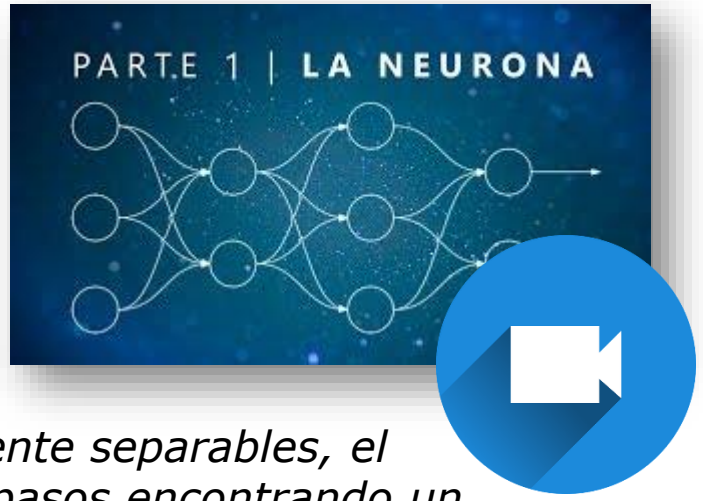
4.3 Procesamiento



Perceptrón de 3 entradas

4.3 Procesamiento

- Si los patrones de entrenamiento no son linealmente separables, entonces
  - No es posible encontrar un perceptrón que de la salida esperada para **todos** los elementos del conjunto de entrenamiento
  - El perceptrón no es capaz de aprender



- Teorema de convergencia del perceptrón:

  Si los patrones de entrenamiento son linealmente separables, el aprendizaje converge en un número finito de pasos encontrando un conjunto de pesos que clasificará las entradas correctamente (Minsky y Papert, MIT Press, 1969)

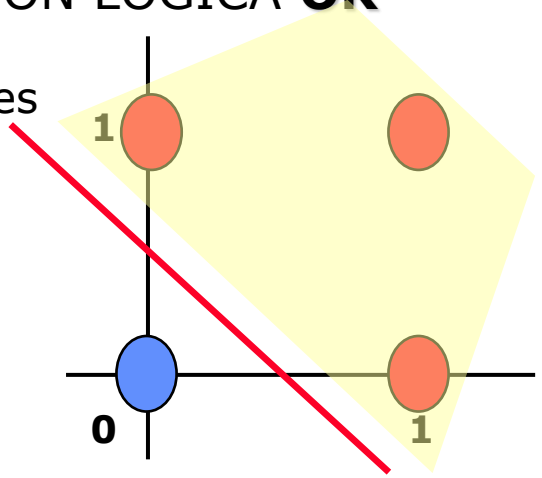
- Condición de separabilidad lineal (demostración formal del teorema de convergencia):

  Para una red de (n) de entradas, el límite entre las dos regiones en que clasifica un perceptrón es un hiperplano de dimensión (n-1) en un espacio vectorial de dimensión

4.3 Procesamiento

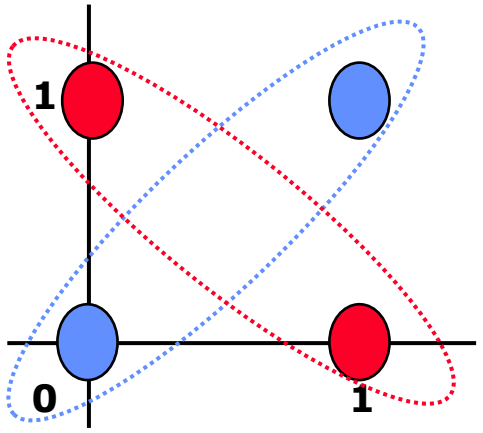
- Patrones linealmente separables: FUNCION LOGICA **OR**
  - **Clasificador**. Define la ecuación de una recta que divide el espacio en dos regiones

X1	X2	Y
0	0	0
0	1	1
1	0	1
1	1	1



- Patrones linealmente **NO** separables: FUNCION LOGICA **XOR**
  - **Comparador**. Con un solo nivel de conexiones no se simulan estas funciones

X1	X2	Y
0	0	0
0	1	1
1	0	1
1	1	0



4.4 Aprendizaje

- Ley de Aprendizaje muy simple, reajustando los pesos si la salida es incorrecta buscando la ecuación del hiperplano

$$\vec{W}_{t+1} = \vec{W}_t + \eta(\vec{d}_t - \vec{Y}_t)\vec{X}_t$$

- Donde
  - η es el Coeficiente de Aprendizaje (Learning Rate): especifica la magnitud de cambio de la red
  - $\vec{d}_t$ es la salida deseada en el instante *t*
  - $\vec{Y}_t$ es la salida obtenida en el instante *t*
  - $\vec{X}_t$ es la entrada aplicada al perceptrón
  - $\vec{W}_t$ es el valor de los pesos (matriz) en el instante t

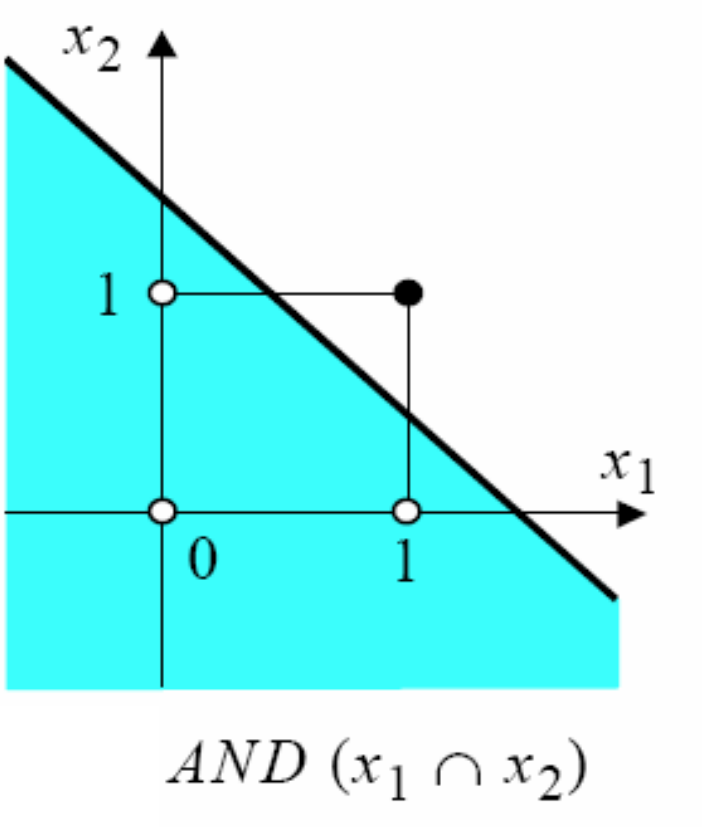
- Regla de aprendizaje por refuerzo (Hebbiana) potenciando las salidas correctas y no se considerando las incorrectas

4.4 Aprendizaje

- Proceso de entrenamiento
  1. Pesos $\vec{W}$ generados aleatoriamente
  2. Introducción de una entrada
  3. Se calcula el error
  4. Error = 0: volver al paso 2 con una nueva entrada
  5. Se inicializa $\Delta W_i = 0$
  6. Se aplica la Ley de Hebb
  7. Se actualizan los pesos
  8. Volver al paso 2 con una nueva entrada

4.4 Aprendizaje

- Ejemplo: resolución del problema AND



X1	X2	Y
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

4.4 Aprendizaje

- Ecuaciones a utilizar

  - Cálculo de la entrada a la neurona (***S***)

$$S = w_1 x_1 + w_2 x_2$$

  - Cálculo de la salida de la neurona (**Y** real)

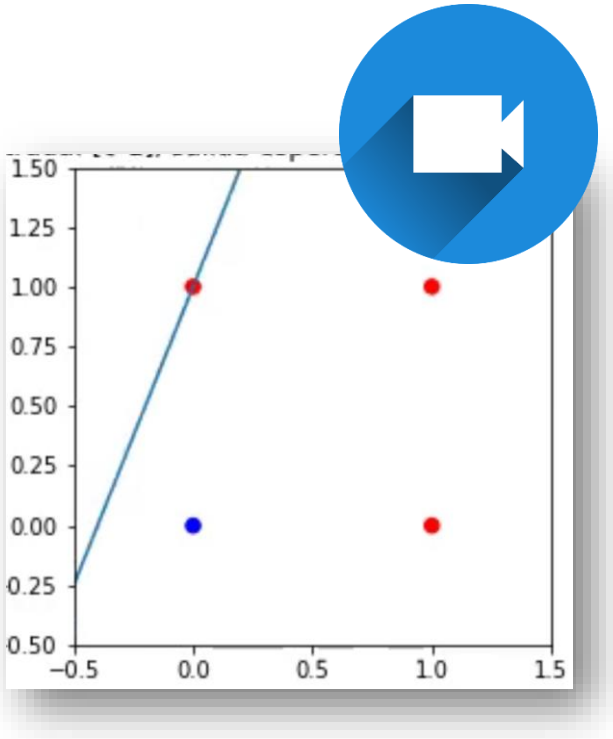
$$Y = \begin{cases} +1 & si\ S \geq \theta \\ 0 & si\ S < \theta \end{cases}$$

  - Error

$$e = d_t - y_t$$

  - Actualización de los pesos (Ley de Hebb)

$$\vec{W}_{t+1} = \vec{W}_t + \eta e \vec{X}_t$$

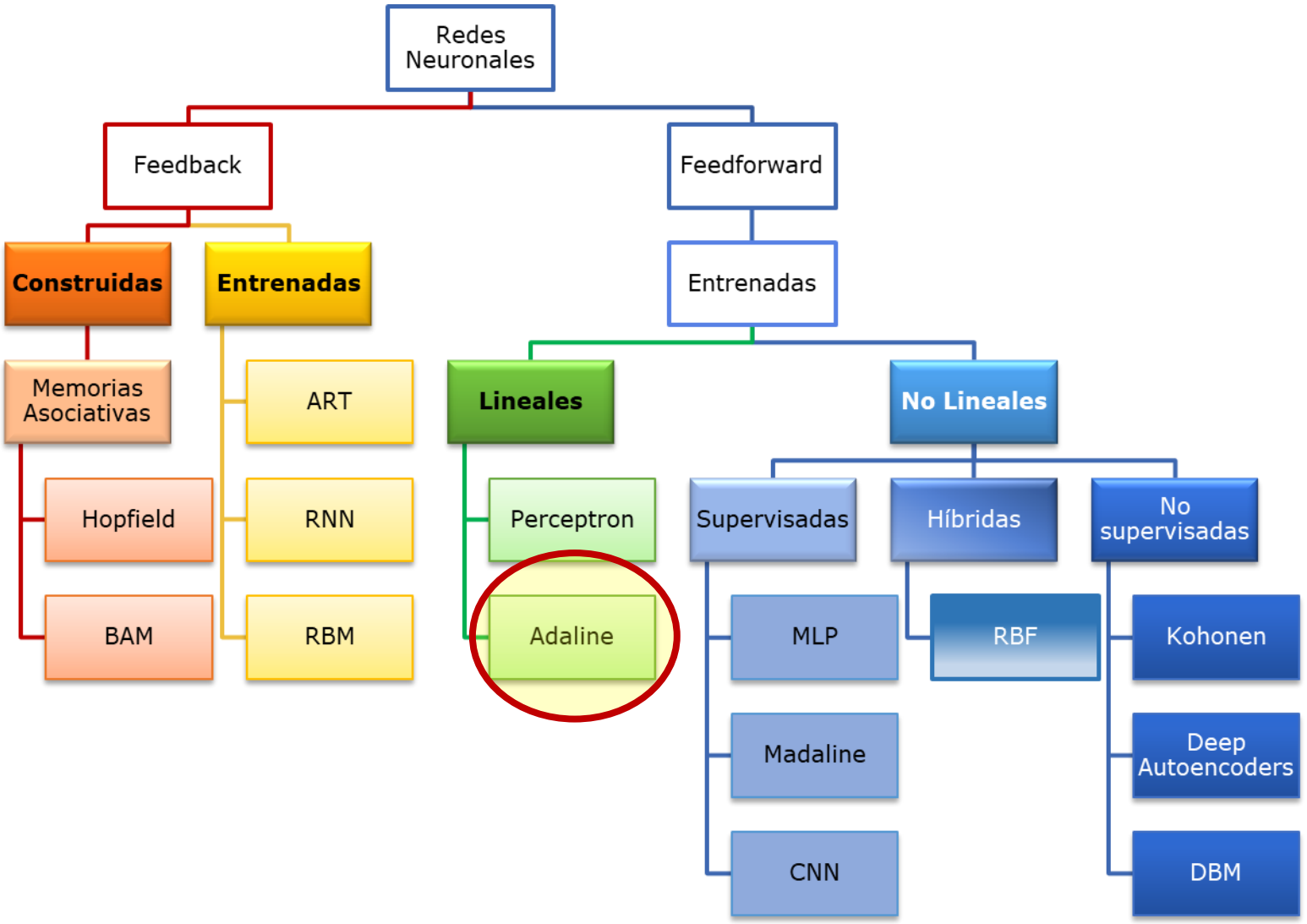


Epoch	Inputs		Desired output	Initial weights		Actual output	Error	Final weights	
	x_1	x_2	Y_d	w_1	w_2	Y	e	w_1	w_2
1	0	0	0	0.3	−0.1	0	0	0.3	−0.1
	0	1	0	0.3	−0.1	0	0	0.3	−0.1
	1	0	0	0.3	−0.1	1	−1	0.2	−0.1
	1	1	1	0.2	−0.1	0	1	0.3	0.0
2	0	0	0	0.3	0.0	0	0	0.3	0.0
	0	1	0	0.3	0.0	0	0	0.3	0.0
	1	0	0	0.3	0.0	1	−1	0.2	0.0
	1	1	1	0.2	0.0	1	0	0.2	0.0
3	0	0	0	0.2	0.0	0	0	0.2	0.0
	0	1	0	0.2	0.0	0	0	0.2	0.0
	1	0	0	0.2	0.0	1	−1	0.1	0.0
	1	1	1	0.1	0.0	0	1	0.2	0.1
4	0	0	0	0.2	0.1	0	0	0.2	0.1
	0	1	0	0.2	0.1	0	0	0.2	0.1
	1	0	0	0.2	0.1	1	−1	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1
5	0	0	0	0.1	0.1	0	0	0.1	0.1
	0	1	0	0.1	0.1	0	0	0.1	0.1
	1	0	0	0.1	0.1	0	0	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1

Threshold: $\theta = 0.2$; learning rate: $\alpha = 0.1$

1. Introducción
2. El problema de la predicción
    1. La predicción
    2. Técnicas estadísticas
    3. Técnicas neuronales
3. Redes a las que hay que enseñar
4. Perceptrón
    1. Historia
    2. Arquitectura
    3. Procesamiento
    4. Aprendizaje
5. Adaline
    1. Aprendizaje
    2. Diferencias con el perceptrón

5. Adaline

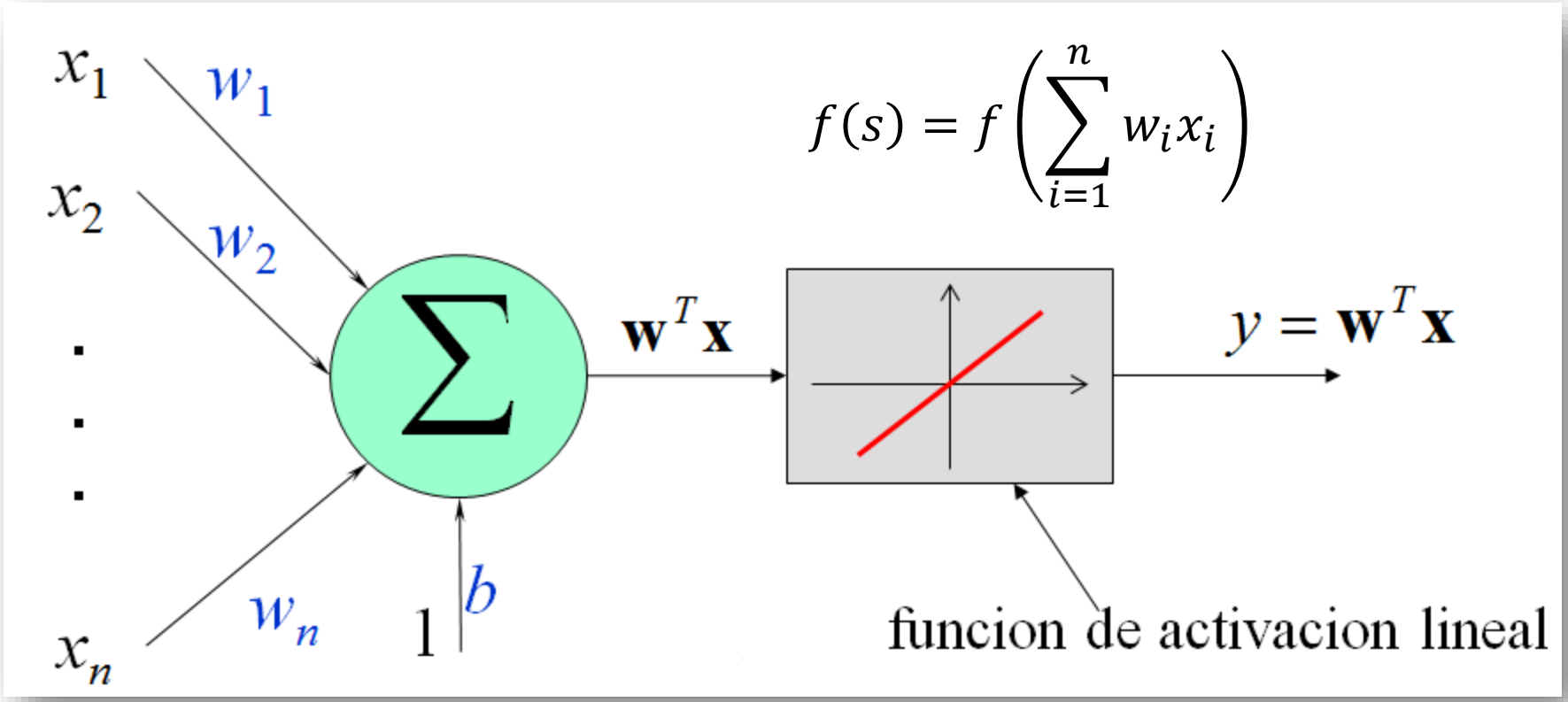

Redes Neuronales
Feedback
Feedforward
Construidas
Entrenadas
Entrenadas
Memorias Asociativas
ART
Lineales
No Lineales
Hopfield
RNN
Perceptron
Supervisadas
Híbridas
No supervisadas
BAM
RBM
Adaline
MLP
RBF
Kohonen
Madaline
Deep Autoencoders
CNN
DBM

5. Adaline

ADAptive **LI**near **NE**uron

- [1960] Bernad Widrow y Hoff lo proponen
  - Dispositivo de umbral con salidas (-1,+1) (en lugar de [0,1] del perceptrón)
  - Elementos de procesamiento lineales (continuos y derivables)
  - Unidad externa (bias) con un peso fijo de valor 1
  - A diferencia del perceptrón, al aprender no intenta minimizar el número de casos mal resueltos, sino la diferencia entre la salida esperada y la obtenida.
  - Esa diferencia se mide con el error cuadrático medio (MSE) del conjunto de entrenamiento

5. Adaline


x_1
x_2
x_n
w_1
w_2
w_n
1
b
$\sum$
$\mathbf{w}^T\mathbf{x}$
$f(s) = f\left(\sum_{i=1}^{n} w_i x_i\right)$
$y = \mathbf{w}^T\mathbf{x}$
funcion de activacion lineal

5.1 Aprendizaje

- La Ley de Aprendizaje es la Regla Delta, Regla LMS (Least Mean Square) o Regla Widrow-Hoff basada en el Descenso de Gradiente
  - Gradiente de una función de coste $C(\vec{x})$ en el punto $\overrightarrow{x^p}$ es el vector de las derivadas parciales de C en $\overrightarrow{x^p}$ (vector gradiente)

$$\nabla C\left(\overrightarrow{x^p}\right) = \frac{\partial C}{\partial \overrightarrow{x^p}} = \left[\frac{\partial C}{\partial x_1^p}, \frac{\partial C}{\partial x_2^p}, \dots, \frac{\partial C}{\partial x_n^p}\right]$$

  - Mide la sensibilidad al cambio del valor de la función con respecto a un cambio en su argumento/parámetro.
  - El descenso de gradiente es cuánto y en qué dirección debe cambiar cada parámetro x_i para minimizar C.

  - En nuestro caso,
    - La función de coste (C) es el error.
    - Los parámetros x_i son los pesos de la red w_i.

5.1 Aprendizaje

- Para aplicar el Descenso de Gradiente, es necesario que la Función de Activación *F* sea derivable (p.e.: sigmoide, lineal) ➔
  - Variación sobre el perceptrón
  - Adaline usa la función lineal de activación
- Adaline no busca la clasificación correcta de cada patrón sino minimizar el Error Cuadrático
- Error:

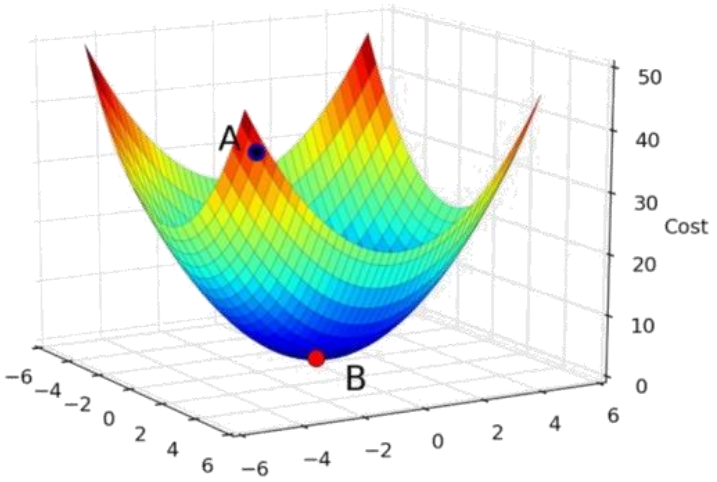
$$E(\overrightarrow{W_t}) = \frac{1}{2}(d_t - Y_t)^2$$

  - Donde
    - d_t es la salida deseada en el instante *t*
    - Y_t es la salida obtenida en el instante *t*
    - $\overrightarrow{W_t}$ es el valor de los pesos (matriz) en el instante t
    - ½ es para simplificar el cálculo posterior de la derivada

5.1 Aprendizaje

- Buscamos un mínimo de *E.*
  - *E* es función de $\vec{W}$.
  - Queremos minimizar *E*
  - Hay que encontrar un $\vec{W}$ que minimice *E*

- ¿Cómo buscar el mínimo?
  - En una superficie diferenciable la dirección de máximo crecimiento viene dada por el vector gradiente $\nabla E(\vec{W})$
  - El negativo del gradiente proporciona la dirección de máximo descenso hacia el mínimo de la superficie.



5.1 Aprendizaje

- Comenzamos con un $\vec{W}$ aleatorio modificado sucesivamente en pequeños desplazamientos en dirección opuesta al gradiente

$$\vec{W}_{t+1}=\vec{W}_t + \Delta\vec{W}_t$$

  - Y llamaremos

$$\Delta\vec{W}_t = -\eta\nabla E(\vec{W}_t)$$

  - Donde
    - η es el factor de aprendizaje que determina el tamaño del desplazamiento
    - $\nabla E(\vec{W})$ es el gradiente o vector de las derivadas parciales de *E* respecto de cada W_i

$$\nabla E(\vec{W_t}) = \frac{\partial E}{\partial \vec{W_t}} = \left[\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \ldots, \frac{\partial E}{\partial w_n}\right]$$

5.1 Aprendizaje

- *E* depende de $\overrightarrow{W}$ a través del producto escalar $S = \sum_{i=1}^{n} w_i x_i$ y S depende de $\overrightarrow{W}$. Aplicamos la regla de la cadena

$$\frac{\partial E}{\partial \overrightarrow{W_t}} = \frac{\partial E}{\partial S}\frac{\partial S}{\partial \overrightarrow{W_t}}$$

  - Donde

$$\frac{\partial S}{\partial \overrightarrow{W_t}} = \frac{\partial}{\partial \overrightarrow{W_t}}\overrightarrow{W_t}\vec{X}_t = \vec{X}_t$$

$$\frac{\partial E}{\partial S} = \frac{\partial}{\partial S}\frac{1}{2}(d_t - Y_t)^2 = -(d_t - Y_t)Y'$$

siendo $y' = {}^{\partial Factivacion}/_{\partial S}$

- El gradiente es, por lo tanto,

$$\nabla E(\vec{w}) = \frac{\partial E}{\partial \overrightarrow{W}} = -(d_t - Y_t)Y'\vec{X}_t$$

5.1 Aprendizaje

- De donde se deduce que

$$\vec{W}_{t+1} = \vec{W}_t + \eta(d_t - Y_t)Y'\vec{X}_t$$

- Es una ecuación que se parece a la de Rosenblatt para el perceptrón bipolar **solo** si la función de transferencia *F* es lineal, en cuyo caso

$$y' = \frac{\partial Factivación}{\partial S} = \frac{\partial S}{\partial S} = 1$$

- Por lo tanto la *Ley de Widrow-Hoff* queda

$$\vec{W}_{t+1} = \vec{W}_t + \eta(d_t - Y_t)\vec{X}_t$$

5.1 Aprendizaje

- En el caso de aprendizaje por lotes, tomaremos como función de error el error cuadrático medio (MSE)

$$\overline{E(\overrightarrow{W_t})} = \frac{1}{2c}\sum_{p=1}^{c}(d_t^p - y_t^p)^2$$

- Con lo que la *Ley de Widrow-Hoff* quedaría

$$\overrightarrow{W}_{t+1} = \overrightarrow{W}_t + \eta\frac{1}{c}\sum_{p=1}^{c}(d_t^p - y_t^p)\,\vec{X}_t$$

5.2 Diferencias con el perceptrón

- La Regla Delta
  - Es un algoritmo de búsqueda local
  - Converge asintóticamente hacia mínimos locales del error
  - Se obtiene el mejor hiperplano posible (no necesariamente uno válido)

Perceptrón	Adaline
Función de activación umbral	Función de activación lineal
Salida binaria	Salida continua $\in \mathcal{R}$
Diferencia entrada/salida es 0/1 (misma/distinta categoría)	Hay medida del error $E \in \mathcal{R}$
Clasifica	Regresión
Converge: En nº finito de pasos si los datos son linealmente separables	Converge: Siempre asintóticamente hacia un mínimo local del ECM
Separación: Hiperplano que separa completamente los datos	Separación: Regresión hacia el mejor hiperplano posible