



Universidad  
Francisco de Vitoria  
**UFV** Madrid

## *Inteligencia Artificial II*

---

**Tema 2:**  
**Aprendizaje No Supervisado:**  
**Aprendizaje Competitivo**

# Objetivos del tema



- Ubicación
  - Bloque II: **COMPUTACION NEURONAL**
    - Tema 2: *Aprendizaje no supervisado: Aprendizaje competitivo*
- Objetivos
  - Comprender que es **clasificar** y los problemas que conlleva
  - Definir que es el **aprendizaje NO supervisado**
  - Modelos de **aprendizaje competitivo**
  - Entender el procesamiento de un Mapa de Kohonen y las **matemáticas asociadas**, así como el significado de cada parámetro que controla el aprendizaje
  - Saber construir un **clásificador neuronal no supervisado**



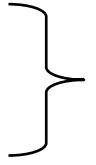
1. Introducción
2. Clasificación
  1. El problema de clasificar
  2. Tipos de clasificadores
3. Redes que aprenden solas
  1. Aprendizaje NO supervisado
  2. Tipos de aprendizaje NO supervisado
4. Mapas Autoorganizativos
  1. Historia
  2. Arquitectura
  3. Procesamiento
  4. Aprendizaje
  5. Mapas topológicos
  6. Parámetros
  7. Modificaciones al proceso básico
5. Construyendo un clasificador



- 1. Introducción**
- 2. Clasificación**
  1. El problema de clasificar
  2. Tipos de clasificadores
- 3. Redes que aprenden solas**
  1. Aprendizaje NO supervisado
  2. Tipos de aprendizaje NO supervisado
- 4. Mapas Autoorganizativos**
  1. Historia
  2. Arquitectura
  3. Procesamiento
  4. Aprendizaje
  5. Mapas topológicos
  6. Parámetros
  7. Modificaciones al proceso básico
- 5. Construyendo un clasificador**



# 1. Introducción

- Algoritmos de aprendizaje NO supervisado están generalmente relacionados con la detección de características:
    - Clustering
    - Clasificación
-   
*Reconocimiento de patrones*



- 1. Introducción**
- 2. Clasificación**
  - 1. El problema de clasificar**
  - 2. Tipos de clasificadores**
- 3. Redes que aprenden solas**
  - 1. Aprendizaje NO supervisado**
  - 2. Tipos de aprendizaje NO supervisado**
- 4. Mapas Autoorganizativos**
  - 1. Historia**
  - 2. Arquitectura**
  - 3. Procesamiento**
  - 4. Aprendizaje**
  - 5. Mapas topológicos**
  - 6. Parámetros**
  - 7. Modificaciones al proceso básico**
- 5. Construyendo un clasificador**

## 2.1 El problema de clasificar



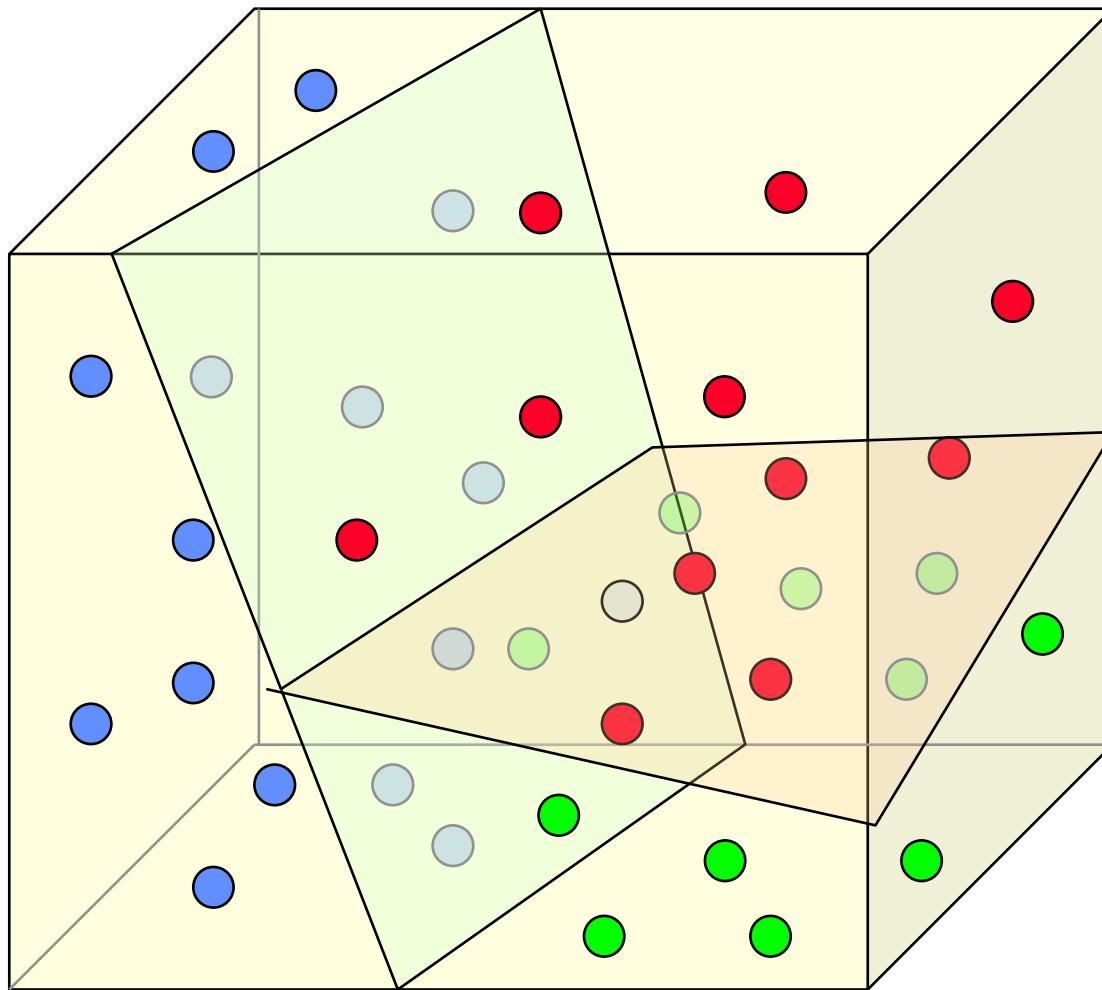
- El problema de la clasificación de datos surge en gran variedad de campos
  - **Punto de partida:** existencia en una población de cierto número de grupos cuando la población es observada a la luz de un determinado comportamiento
    - Son necesarios los dos elementos: Población y Criterio
  - **Problema:** predecir el grupo al que pertenece un individuo dado a partir de un conjunto de características del individuo relevantes respecto al criterio de clasificación
    - No siempre esas características están disponibles
  - **Solución:** procedimientos que permitan, con datos indirectos, determinar la pertenencia de un individuo a una determinada clase
    - Con o sin información adicional

## 2.1 El problema de clasificar



Métodos de clasificación:

- Llevan a cabo el **CLUSTERING**
  - Dividir el espacio N-dimensional de características ...
    - al que pertenecen los vectores definidos por las N características de cada individuo de la población
  - ... en K regiones excluyentes
    - correspondientes a los K posibles grupos
- Para luego hacer **CLASIFICACIÓN**
  - Asignar cada patrón a uno de los grupos/clases definidos previamente
    - los patrones usados en el clustering
    - o cualquier nuevo patrón que se presente



## 2.1 El problema de clasificar



- Ningún método de clasificación tiene validez universal
- Requerimientos matemáticos artificiales para la separabilidad de categorías

*Las características N-dimensionales deben de ser multinormales con iguales covarianzas*

- Un clasificador es tanto más potente cuanto más información relevante para el problema pueda tratar
  - Representada como un vector N-dimensional
  - Valor bajo de N puede implicar insuficientes criterios de clasificación
  - Valor alto de N
    - Difícil de interpretar tantos criterios
    - Procesos de cálculo largos y costosos



## 2.2 Tipos de clasificadores

### Discriminación

- *Discriminantes lineales (p.e.: de Fisher) o no lineales*
  - Clasificación supervisada
  - Con información contenida en un conjunto de muestras previamente clasificadas determinan un criterio para etiquetar cada individuo como perteneciente a alguno de los grupos
  - Parten de los valores de una serie limitada de parámetros
  - La clasificación se realiza comparando características del individuo y de los grupos
  - Dividen el espacio de estados en regiones excluyentes (lineales o no lineales)
- *Regresión logística*



## 2.2 Tipos de clasificadores

### Clustering

- *Métodos de clustering (o estimación de densidades)*
  - Clasificación No supervisada
  - No se dispone de una muestra previamente clasificada. A priori no se conocen los grupos y lo que precisamente se desea es establecerlos a partir de los datos que poseemos
  - La agrupación de individuos se realiza minimizando ciertas funciones de distancia mediante técnicas estadísticas

– Distancia Euclídea       $\sqrt{|x_1 - x_2|^2 + |y_1 - y_2|^2}$

– Distancia de Manhattan       $|p_1 - q_1| + |p_2 - q_2|$

– Distancia de Mahalanobis



## 2.2 Tipos de clasificadores

### Clasificadores neuronales

- Trabajan como si conociesen las reglas explícitas que permiten clasificar (obtenidas por entrenamiento)
- Aprovecha sus propiedades de dividir espacios no linealmente separables entre sí
- Tipos
  - Supervisados
  - No supervisados
- Estructura:
  - Patrón de entrada: conjunto de números reales en forma de vector

$$\vec{X} = \{x_1, x_2, x_3, \dots, x_n\}$$

- Capa de entrada: una por cada elemento del vector de entrada (componente del patrón)
- Capa de salida: tantas neuronas como categorías o una neurona con múltiples valores



- 1. Introducción**
- 2. Clasificación**
  1. El problema de clasificar
  2. Tipos de clasificadores
- 3. Redes que aprenden solas**
  1. Aprendizaje NO supervisado
  2. Tipos de aprendizaje NO supervisado
- 4. Mapas Autoorganizativos**
  1. Historia
  2. Arquitectura
  3. Procesamiento
  4. Aprendizaje
  5. Mapas topológicos
  6. Parámetros
  7. Modificaciones al proceso básico
- 5. Construyendo un clasificador**

### 3. Redes que aprenden solas

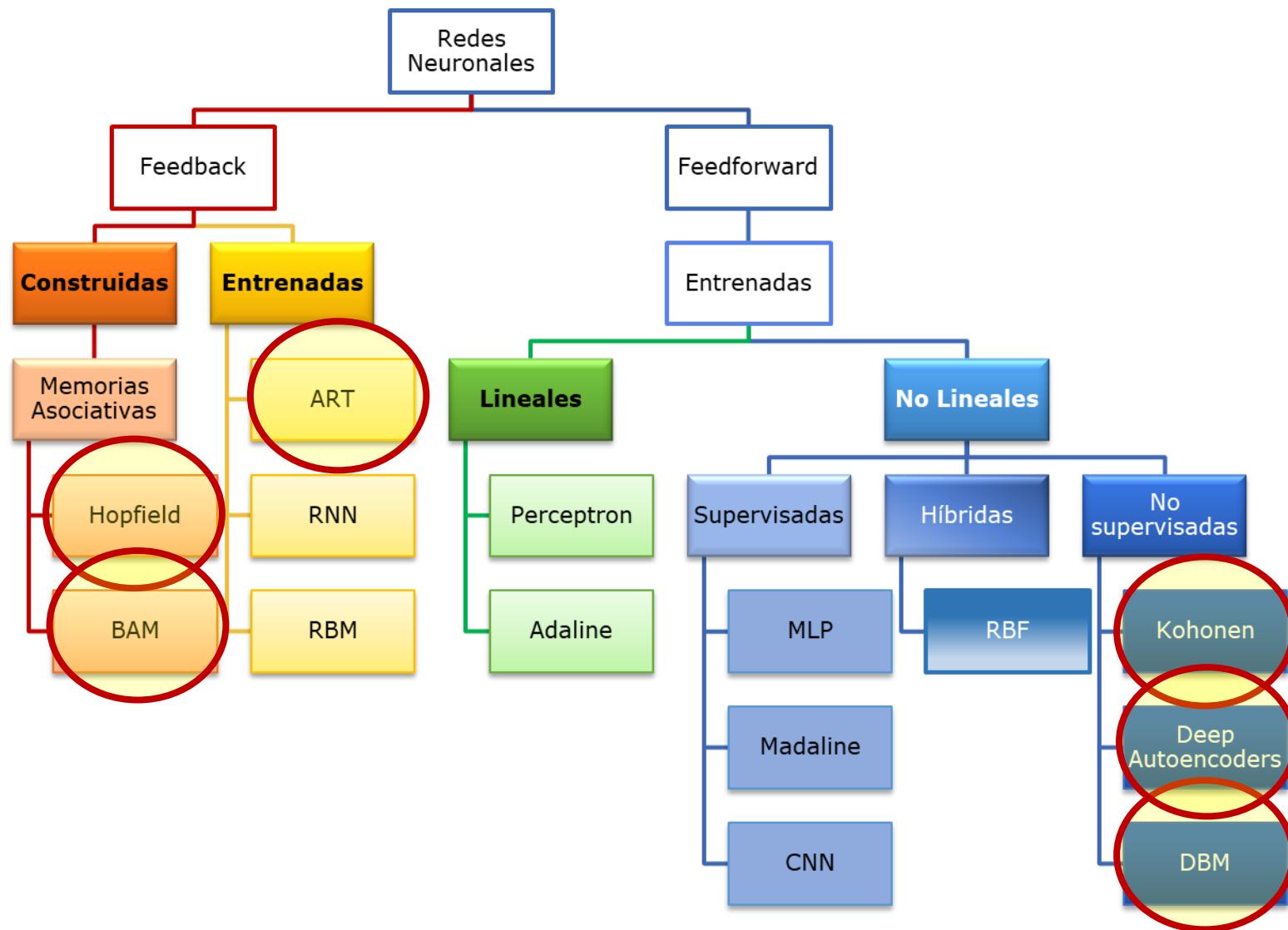


**Validas para la clasificación ...  
... y para otras tareas**

- Modelos FeedBack
  - Redes construidas
    - Memorias Asociativas:
      - Modelo de Hopfield
  - Redes entrenadas
    - Modelos ART
- Modelos FeedForward
  - Modelos no lineales de aprendizaje no supervisado
    - Mapas autoorganizativos (SOM, redes/mapas de Kohonen)
    - *Neocognitrón* (solo el modelo original, el actual es supervisado)
  - Modelos Deep Learning
    - Autoencoders



### 3. Redes que aprenden solas



### 3.1 Aprendizaje no supervisado



- Clasificadores Neuronales NO supervisados
  - reflejan la forma en que el cerebro de un niño se auto-organiza y adapta a condiciones cambiantes.  
(comportamiento similar al del bebé recién nacido)*
- ¿Puede un organismo categorizar el mundo (poner etiquetas) por simple interacción con el entorno, sin conocimiento previo?
  - Tras un período de tiempo adquiere las habilidades para realizar tareas complejas
    - Andar
    - Reconocer a los padres
  - El organismo crea criterios de clasificación extraídos de su propia interacción con el mundo
  - Genera sus propias categorías del mundo

### 3.1 Aprendizaje no supervisado



- Fundamento biológico
  - El cerebro humano está controlado por el *córtex cerebral*, una estructura muy compleja de  $10^9$  neuronas
  - El *córtex* incluye áreas responsables de diferentes actividades humanas asociadas con dispositivos sensoriales:
    - Motora
    - Visual
    - Auditiva
    - Somatosensorial
  - Cada entrada sensorial está mapeada a su correspondiente área del *córtex cerebral*

*El córtex es un mapa computacional auto-organizado dentro del cerebro humano*

# 3.1 Aprendizaje no supervisado



- Fundamento computacional
  - Entrenamiento de la red neuronal → sólo datos de entrada
    - Recibe patrones de entrada
    - Extrae inductivamente propiedades estadísticas relevantes en los patrones del conjunto de entrenamiento
    - Aprende como clasificar esos patrones en categorías
  - Procesamiento feedforward
  - Más natural y biológico que el aprendizaje supervisado porque:
    - **NO** requiere propagar cierta información (el error cometido) hacia atrás, en sentido contrario al flujo natural de la información (esto no sucede en las neuronas reales)
    - **NO** requiere de un instructor que proporcione la salida deseada. Las personas aprendemos a comprender frases o escenas visuales sin instrucciones explícitas para ello
    - Tiende a seguir la organización neurobiológica del cerebro



## 3.1 Aprendizaje no supervisado

- Algoritmos de aprendizaje NO supervisado:
  - *Descubren modelos o características significativas a partir de los datos de entrada*
  - Basados en reglas de naturaleza local (cambios aplicado al vector de pesos de una neurona en detrimento de las demás)
- **Autoorganización:** proceso en el cual, por medio de interacciones locales, se obtiene un ordenamiento global mediante un proceso de aprendizaje NO supervisado.
- En las redes autoorganizadas se emplean varias reglas de aprendizaje. Entre otras
  - Aprendizaje Hebbiano
  - Aprendizaje por Componentes Principales
  - Aprendizaje Competitivo

## 3.2 Tipos de aprendizaje no supervisado



### Aprendizaje Hebbiano o Asociativo (Hopfield)

- De acuerdo al postulado de Donald Hebb (1949)  
*"Cuando un axón de la célula A excita la célula B y, repetidamente la activa, algún proceso de crecimiento o cambio metabólico toma lugar en una o ambas células tal que la conexión sináptica se fortalece y B se vuelve más sensible a los estímulos de A"*
- La Ley de Hebb puede representarse como dos reglas:
  1. *Si dos neuronas en cada lado de una conexión se activan sincrónicamente, entonces el peso de esa conexión se incrementa*
  2. *Si dos neuronas en cada lado de una conexión se activan asincrónicamente, entonces el peso de esa conexión se decrementa*



## 3.2 Tipos de aprendizaje no supervisado

- La información se almacena en las conexiones entre las neuronas en forma de pesos.
  - Fenómeno local sin intervención del exterior.
  - Aprendizaje a partir de los datos, sin supervisión.
  - No depende del entorno.
- Durante el aprendizaje, la activación repetida de las neuronas conectadas entre si cambia gradualmente la fuerza de los pesos, lo que lleva a conexiones más fuertes.
- El cambio de peso entre neuronas es proporcional al producto de los valores de activación de las neuronas.
- Proporciona las bases de aprendizaje no supervisado.



## 3.2 Tipos de aprendizaje no supervisado

- Aprendizaje:

$$\vec{W}^{nuevo} = \vec{W}^{viejo} + cambio$$

- Sea  $\vec{x}$  el vector de entrada a la neurona. El vector de salida es  
$$\vec{y} = \vec{W}\vec{x}$$
- El vector de pesos que satisface esa ecuación para un par concreto  $(\vec{y}_1, \vec{x}_1)$ , es

$$\vec{W} = \vec{y}_1 \vec{x}_1^\top$$

- Si todos los patrones de entrada son ortonormales, entonces

$$\vec{W} = \vec{y}_1 \vec{x}_1$$

- Con lo que la ley de aprendizaje de Hebb quedaría:

$$\vec{W}^{nuevo} = \vec{W}^{viejo} + \vec{y}_i \vec{x}_i$$



## 3.2 Tipos de aprendizaje no supervisado

- El aprendizaje no depende del entorno, solo de la entrada y la salida de la neurona
- Esta ley es útil siempre y cuando los vectores sean ortonormales.
  - Condición es muy estricta
  - El número de ejemplos para entrenamiento estaría limitado a la dimensionalidad de  $x$ 
    - Para un espacio n-dimesional, el número de patrones de entrada que pueden asociarse será  $n$ .
- Ejemplo de aplicación de la ley de Hebb en  $n = 2$ 
  - con dos patrones de entrada  $\vec{x}=(x_1 \ x_2)$ 
    - (0 1) y (1 0) (obviamente!)
  - dos patrones de salida  $\vec{y}$ 
    - 2 y 5 respectivamente (iporque si!)



## 3.2 Tipos de aprendizaje no supervisado

- Para el primer par  $\vec{x}$  e  $\vec{y}$  ( $[0 \ 1]$ , 2)

$$\vec{W}^{nuevo} = \vec{W}^{viejo} + \vec{y}_1 \vec{x}_1 = [0 \ 0] + 2 [0 \ 1] = [0 \ 2]$$

- Para el segundo par  $\vec{x}$  e  $\vec{y}$  ( $[1 \ 0]$ , 5) se actualizan los pesos para establecer la nueva asociación:

$$\vec{W}^{nuevo} = \vec{W}^{viejo} + \vec{y}_2 \vec{x}_2 = [0 \ 2] + 5 [1 \ 0] = [5 \ 2]$$

- Comprobamos si los pesos se ajustaron para reconocer las asociaciones.

- Si la entrada es  $[1 \ 0]$  la salida y sería

$$\vec{y} = \vec{W} \vec{x} = [5 \ 2] [1 \ 0] = 5$$

- Si la entrada hubiera sido  $[0 \ 1]$ , la salida y sería

$$\vec{y} = \vec{W} \vec{x} = [5 \ 2] [0 \ 1] = 2$$

- Además  $[1 \ 0]$  y  $[0 \ 1]$  son ortonormales (unitarios y ortogonales)

## 3.2 Tipos de aprendizaje no supervisado



### Aprendizaje por Principal Component Analysis (PCA)

- Expresa un conjunto de individuos como una combinación lineales de factores no correlacionados entre sí.
  - Los factores dan cuenta una fracción cada vez más débil de la variabilidad de los datos
- El objetivo es encontrar  $M$  vectores ortogonales de longitud unidad que modelicen la mayor parte posible de la variabilidad de los datos de dimensión  $N$ .
  - Típicamente  $M \ll N$
  - Reduce la dimensionalidad preservando la mayor información de entrada posible.
  - Aprendizaje: análisis de las similitudes compartidas por los patrones de entrada

## 3.2 Tipos de aprendizaje no supervisado



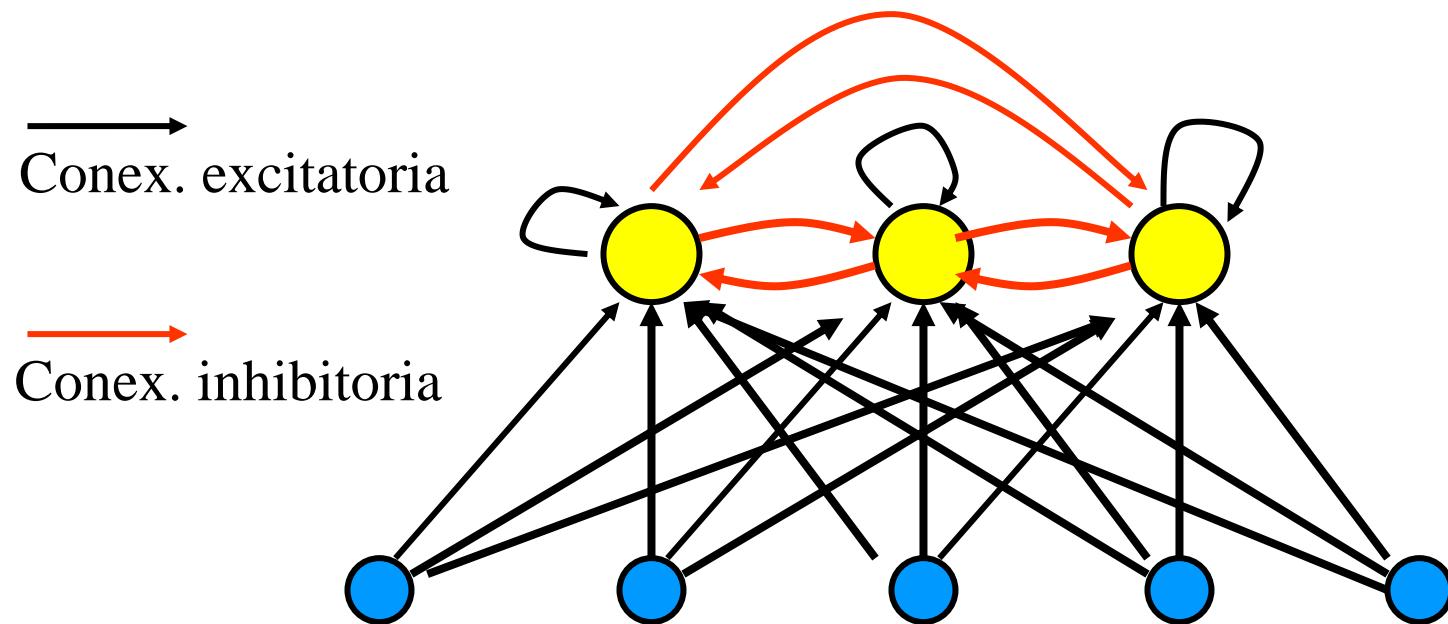
### Aprendizaje Competitivo (SOM y ART)

- Basado en el modelo de competición y selección natural, introducido a comienzo de los 70.
  - Un gran número de neuronas *compiten* entre sí para ser activadas
  - Solo se activa una neurona para un patrón de entrada determinado (modelo *Winner-takes-all*)
    - La salida de la neurona ganadora es 1
    - La salida de las demás neuronas, 0
  - La neurona ganadora es aquella cuyo vector de pesos se asemeja más al patrón de entrada
- La reconstrucción del patrón de entrada consistirá en copiar la distribución de pesos de la unidad ganadora.



## 3.2 Tipos de aprendizaje no supervisado

- El aprendizaje
  - Consiste en reforzar las conexiones de la unidad ganadora y debilitar las otras.
  - No depende del entorno, solo de la competencia entre las neuronas de la red





1. Introducción
2. Clasificación
  1. El problema de clasificar
  2. Tipos de clasificadores
3. Redes que aprenden solas
  1. Aprendizaje NO supervisado
  2. Tipos de aprendizaje NO supervisado
4. Mapas Autoorganizativos
  1. Historia
  2. Arquitectura
  3. Procesamiento
  4. Aprendizaje
  5. Mapas topológicos
  6. Parámetros
  7. Modificaciones al proceso básico
5. Construyendo un clasificador

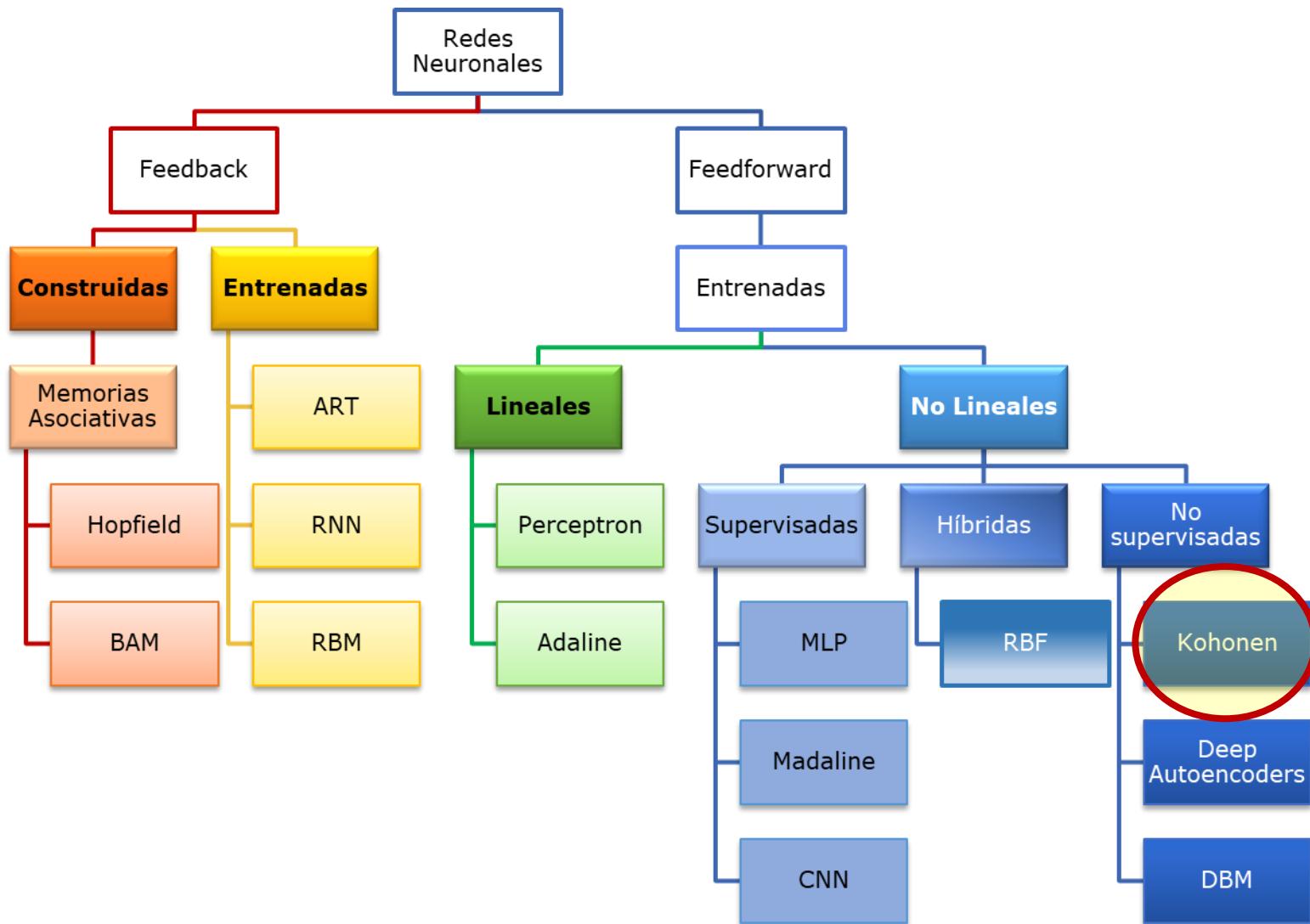
# 4. Mapas Autoorganizativos



- Redes Neuronales...
  - Modelo feedforward
  - No lineales
  - Entrenadas
  - De aprendizaje NO supervisado
    - [Mapas Autoorganizativos de Kohonen](#)
    - Counter-propagation (Kohonen + ART)
    - LVQ (Linear Vector Quantization)



# 4. Mapas Autoorganizativos





## 4.1 Historia

- [1973] Modelo neuronal de aprendizaje no supervisado basado en los trabajos de Willshaw y von der Malsburg.
  - **von der Malsburg** enunció los primeros conceptos relacionados con lo que él llamó *aprendizaje competitivo*
  - Demostró que es adecuado para problemas en los que existen regularidades estadísticamente sobresalientes en un conjunto de patrones de entrada
- [1979 – 1982] **Teuvo Kohonen** de la Universidad Tecnológica de Helsinki (Finlandia) crea:
  - **SOM** (Self Organising Map)
  - Mapa Auto-organizativo
  - Red de Kohonen

Basados en los modelos de aprendizaje competitivo de von der Marlsburg.



## 4.1 Historia

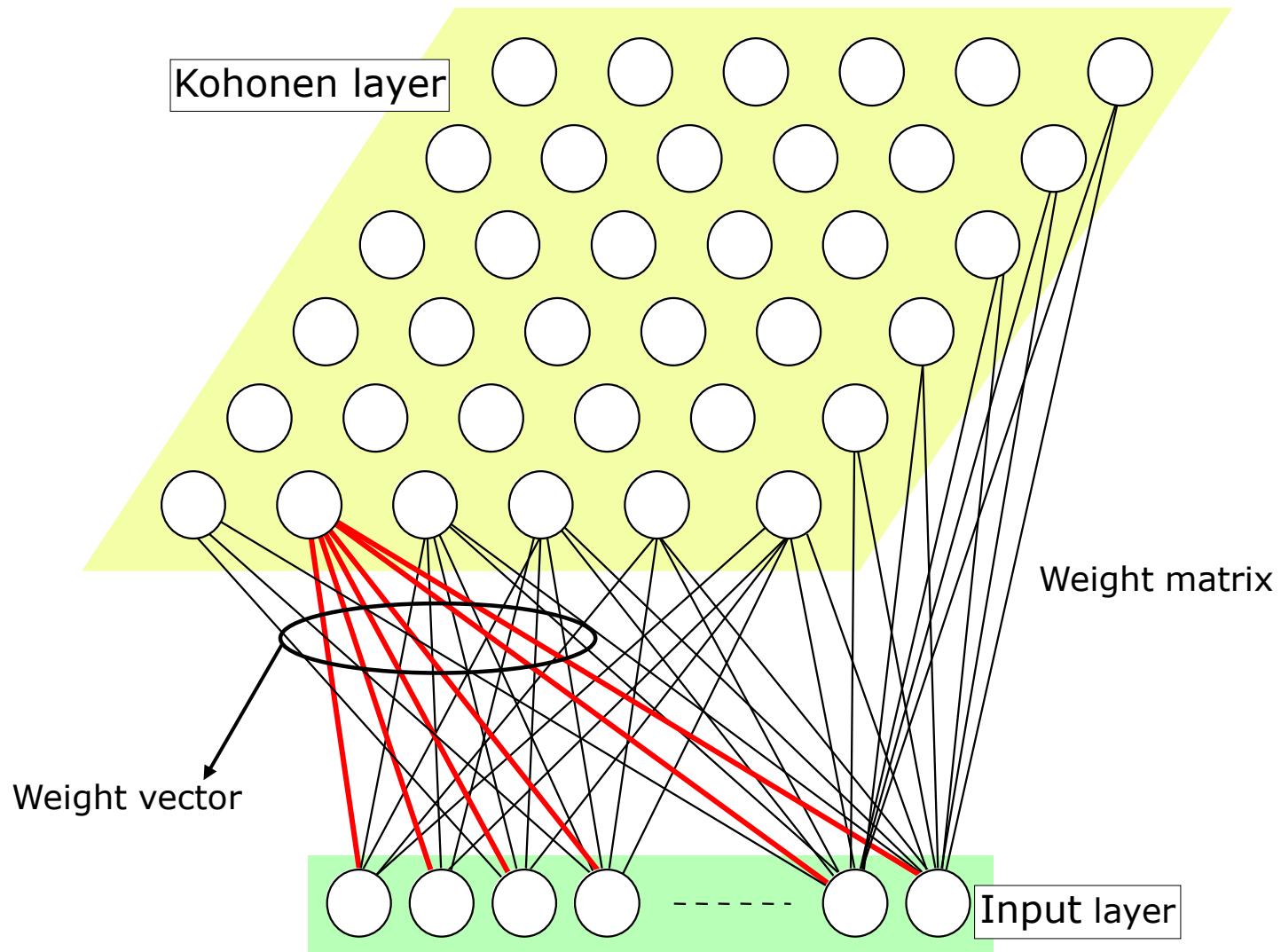


- Aplicaciones
  - **Clustering**, agrupación de los datos de entrada en “cluster” o grupos
  - **Clasificación**, asignación de una clase a nuevos patrones
  - **Reducción de dimensión**, representan estructuras de espacios de dimensión mayor en un mapa 2-D
  - **Extracción de características** significativas de la señal de entrada
- Aplicaciones más destacadas: *dictáfono de Kohonen*
  - Reconocimiento automático del habla
  - Trascripción de habla continua a una secuencia de fonemas.
    - Una vez generada y entrenada, la red es capaz de generar texto escrito a partir de fonemas contenidos en una cinta grabada en voz
    - Por cada fonema emitido, se activa una neurona de la red asociada a ese fonema

## 4.2 Arquitectura



- Capas
  - **NO** es un sistema jerárquico de capas unidimensionales. Solamente hay 2 capas:
    - *Capa de entrada*
      - Capa monodimensional
      - Contiene tantas neuronas como componentes tengan los vectores de los patrones presentados a la red.
      - Sirven de conexión con el exterior
    - *Capa de Kohonen*
      - Capa multidimensional, por lo general bidimensional (malla cuadrada o rectangular)
      - Completamente interconectada con la capa de entrada y entre si
      - Las conexiones determinan la activación de una neurona y la inhibición del resto ante un patrón determinado, permitiendo que cada patrón se asocie a una única neurona de esta capa.



## 4.2 Arquitectura



- Elementos de un SOM
  1. Una *matriz multidimensional* de neuronas que procesa patrones que le llegan desde un array de neuronas de entrada (*tantas como componentes* tienen los patrones).
  2. Una *función discriminante* calculada a partir de un patrón de entrada y los pesos de cada una de las neuronas de la matriz (métrica: *distancia euclídea*).
  3. Un mecanismo de *competición* que *compara las funciones discriminantes* de todas las neuronas y selecciona aquella con el mejor valor de la función (*BMU*).
  4. Un mecanismo de *cooperación por interacción local* que *active* simultáneamente *la neurona seleccionada* y sus vecinas.
  5. Un mecanismo de *adaptación* en virtud del cual los parámetros de las neuronas activas *aumentan su función discriminante* en relación a la entrada actual.

## 4.3 Procesamiento



- Una red ya entrenada, ante un patrón dado, **CLASIFICA**
  - **Busca** la clase a la que pertenece (el vector de pesos más parecido al patrón de entrada)
  - **Activa** la misma neurona de salida
- Sea una Red de Kohonen donde
  - $N$  es el tamaño de la capa de entrada ( $N$  variables de entrada)
  - $K$  es el tamaño del lado de la capa de Kohonen ( $K \times K$  neuronas)
  - $X = (x_1, x_2, \dots, x_N)$  es el vector de entradas o un punto en el espacio de vectores  $N$ -dimensional
  - $W_j = (w_{1j}, w_{2j}, \dots, w_{Nj})$  es el vector de pesos de las conexiones de las neuronas de la capa de entrada con la neurona  $j$  de la capa de Kohonen

## 4.3 Procesamiento



- El procesamiento de la red se realiza en modo “feedforward” y por competición en tres etapas que se describen a continuación:

### 1. Propagación de la entrada

- Capa de Entrada: se asigna el valor de cada una de las componentes del vector de entrada a la neurona correspondiente de esta capa.

### 2. Cálculo de la función discriminante

- Se calcula la distancia euclídea entre el patrón de entrada y el vector de pesos de cada una de las neuronas de la capa de Kohonen de acuerdo a la siguiente fórmula:

$$D_j = \|X - W_j\| = \sqrt{\sum_{i=1}^N (x_i - w_{ij})^2}$$

## 4.3 Procesamiento



- Si los vectores están normalizados con la norma euclídea, esta operación equivale a realizar el producto escalar del vector de entradas por el vector de pesos que une cada neurona con las de la capa de entrada:

$$E = XW_j = \sum_{i=1}^N x_i \cdot w_{ij}$$

### 3. Competición

- Se activa aquella neurona de la capa de Kohonen cuyo vector de pesos esté más próximo al patrón de entrada (distancia euclídea menor o producto escalar mayor), obligando al resto a inhibirse.
- La salida es, por lo tanto, binaria

$$O_j = \begin{cases} 1 & E_j = \max_{i=1..K^2} E_i \\ 0 & E_j \neq \max_{i=1..K^2} E_i \end{cases}$$

## 4.4 Aprendizaje



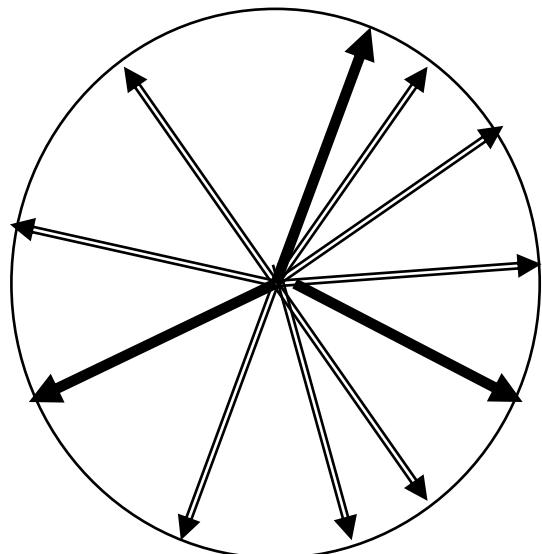
*Ajuste de la matriz de pesos de las neuronas de la capa de Kohonen a medida que se presentan patrones (vectores) de entrada en un proceso iterativo (entrenamiento)*

- Obliga a la red a auto-organizarse de forma que
  - Detecta la forma en que los datos de entrada se distribuyen en el espacio N-dimensional
  - Las neuronas que responden de forma similar a ciertas entradas se agrupan (**compactan**)
- Durante el aprendizaje la red dispone exclusivamente de información sobre los patrones de entrada. No hay valores de “salida”.

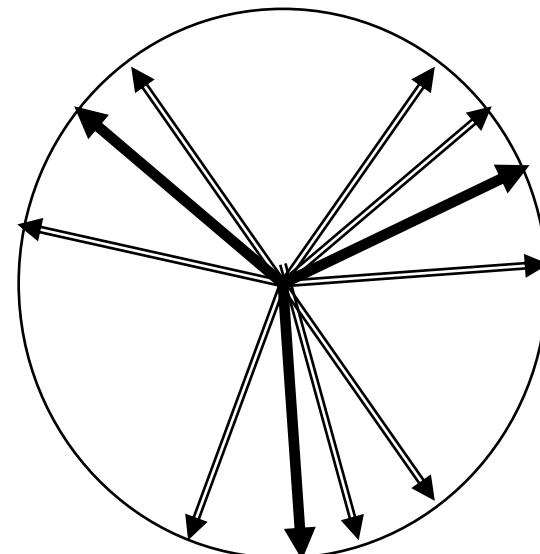
## 4.4 Aprendizaje



- Desde un punto de vista matemático, *una capa de neuronas modifica sus pesos de forma que la distribución de los vectores de pesos tiende a aproximar la función de densidad probabilística de los vectores de entrada*



**Situación inicial aleatoria**



**Red entrenada**



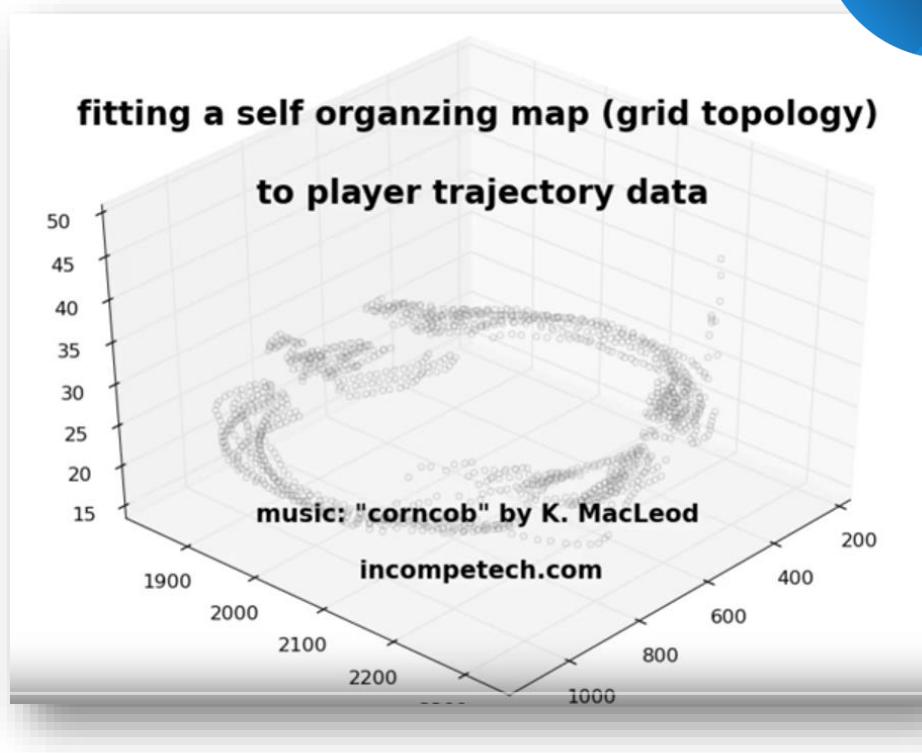
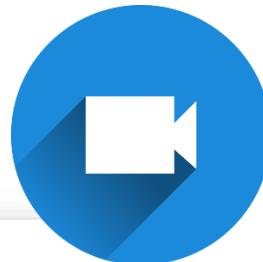
Vectores de pesos



Vectores de entrada



## 4.4 Aprendizaje





### Proceso de aprendizaje (ajuste de la matriz de pesos)

1. Se inicializa la matriz de pesos con valores aleatorios
  - Los vectores de pesos resultantes se pueden normalizar:

$$w_{ij} = \frac{w'_{ij}}{\left[ \sum_{i=0}^N (w'_{ij})^2 \right]^{\frac{1}{2}}}$$

2. Se presentan uno a uno los patrones del conjunto de entrenamiento, también normalizados.
  - “*presentación*”: aplicar un vector patrón a la entrada de la red.
  - el número de presentaciones alcanza un valor prefijado llamado *periodo* (*p*).

## 4.4 Aprendizaje



### 3. Se determina la neurona ganadora (BMU)

- En cada presentación se procesa la entrada según lo visto en el apartado 4.3 obteniendo la activación de una única neurona en la capa de Kohonen (BMU).

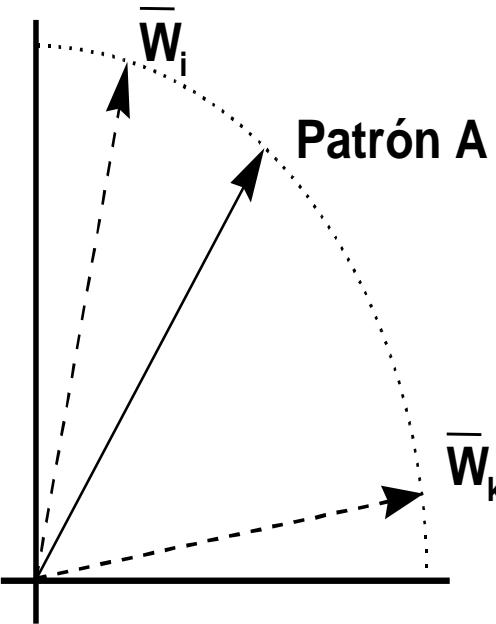
### 4. Se actualizan los pesos de la BMU

- Para esa neurona se aproximan el vector de pesos al vector del patrón de entrada según un *coeficiente de aprendizaje*  $\eta$ .
- La Regla de Aprendizaje Competitivo define el cambio (delta)  $\Delta W_{ij}$  aplicado a la componente del vector de pesos  $W_{ij}$  como:

$$\Delta W_{ij} = \begin{cases} \eta(t)(X_i - W_{ij}) & \text{Si la neurona } j \text{ gana} \\ 0 & \text{Si la neurona } j \text{ NO gana} \end{cases}$$



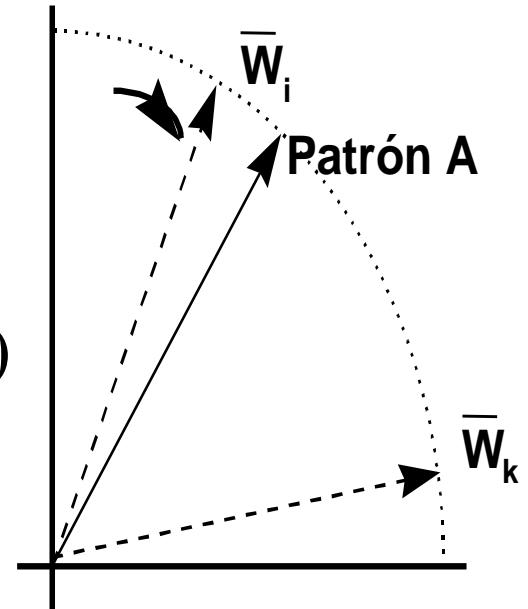
## 4.4 Aprendizaje



Entrenamiento



$$W_{i,j}^{t+1} = W_{i,j}^t + \eta(t)(X_i - W_{i,j}^t)$$





## 4.4 Aprendizaje

### Ejemplo

- Supongamos un patrón 2-D  $\mathbf{X} = \begin{bmatrix} 0.52 \\ 0.12 \end{bmatrix}$
- Presentado a una red de Kohonen de 3 neuronas

$$\mathbf{W}_1 = \begin{bmatrix} 0.27 \\ 0.81 \end{bmatrix}$$

$$\mathbf{W}_2 = \begin{bmatrix} 0.42 \\ 0.70 \end{bmatrix}$$

$$\mathbf{W}_3 = \begin{bmatrix} 0.43 \\ 0.21 \end{bmatrix}$$

- La neurona ganadora se calcula usando el criterio de la mínima distancia euclídea

$$d_1 = \sqrt{(x_1 - w_{11})^2 + (x_2 - w_{21})^2} = \sqrt{(0.52 - 0.27)^2 + (0.12 - 0.81)^2} = 0.73$$

$$d_2 = \sqrt{(x_1 - w_{12})^2 + (x_2 - w_{22})^2} = \sqrt{(0.52 - 0.42)^2 + (0.12 - 0.70)^2} = 0.59$$

$$d_3 = \sqrt{(x_1 - w_{13})^2 + (x_2 - w_{23})^2} = \sqrt{(0.52 - 0.43)^2 + (0.12 - 0.21)^2} = 0.13$$



## 4.4 Aprendizaje

- La ganadora es la neurona 3 y su vector de pesos ( $W_3$ ) se actualiza de acuerdo a la regla de aprendizaje competitivo

$$\Delta w_{13} = \alpha (x_1 - w_{13}) = 0.1 (0.52 - 0.43) = 0.01$$

$$\Delta w_{23} = \alpha (x_2 - w_{23}) = 0.1 (0.12 - 0.21) = -0.01$$

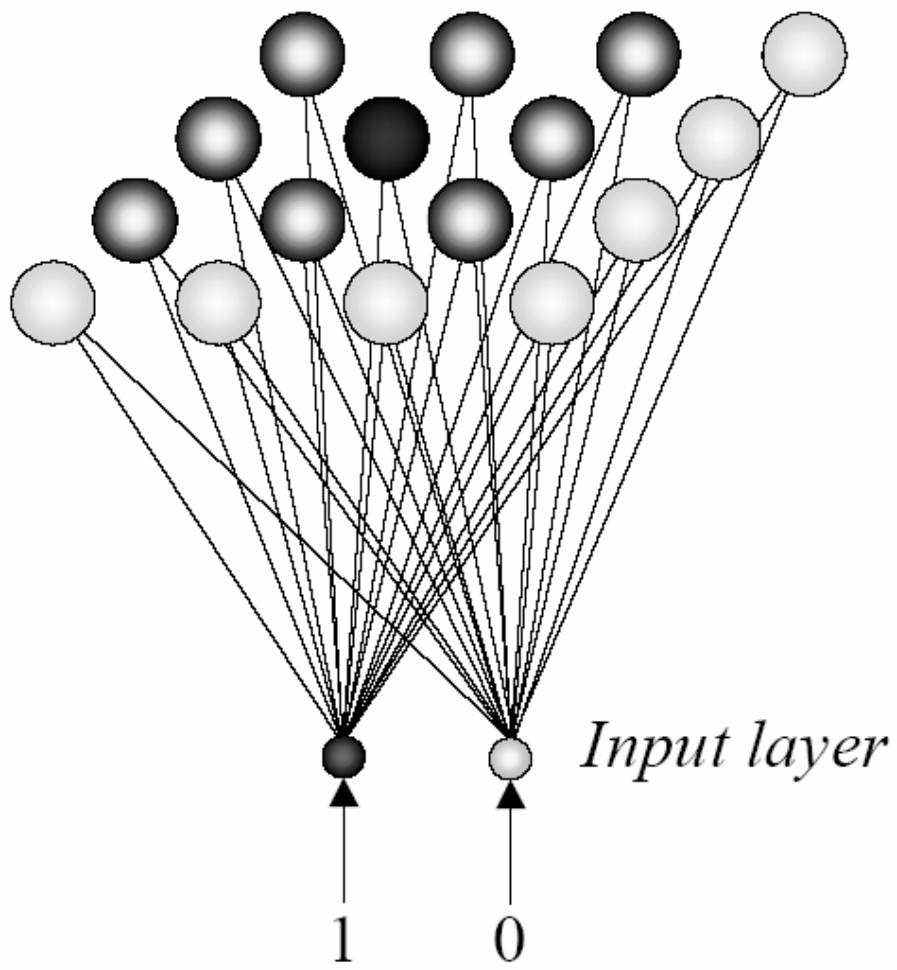


### Vecindario

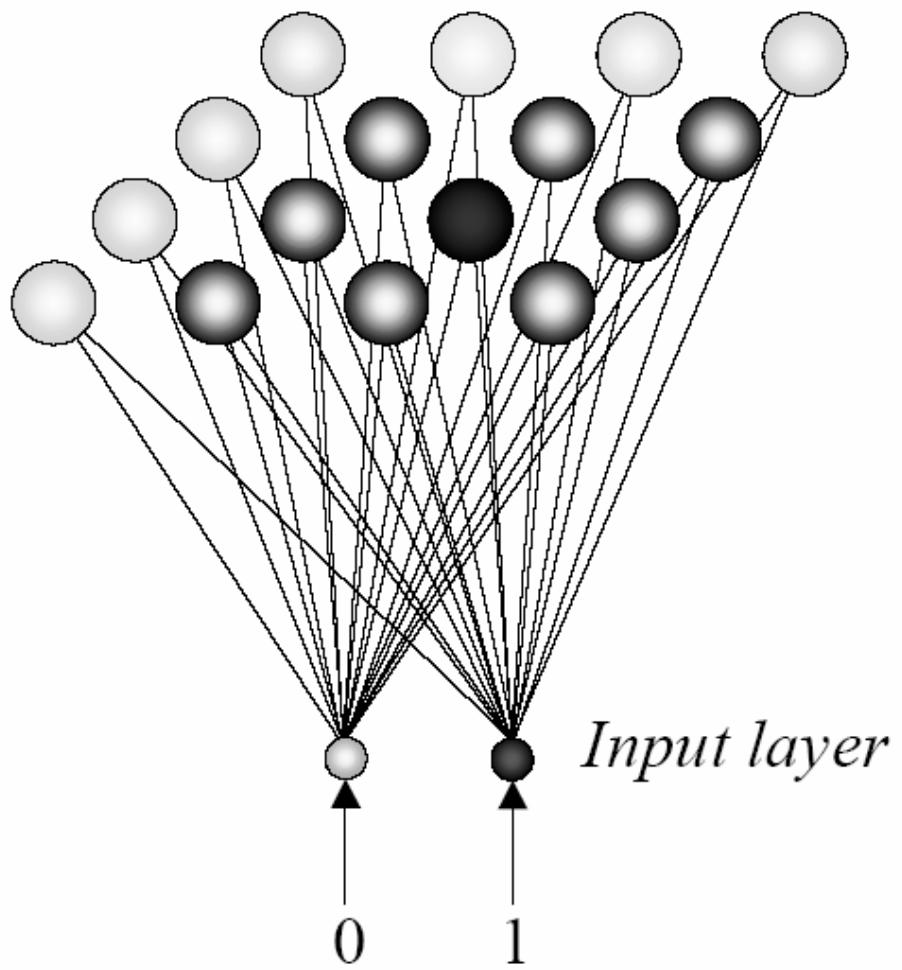
- El aprendizaje no se limita a la BMU sino que se extiende a las neuronas adyacentes:
  - Todas las neuronas que se encuentran en el *vecindario* de la ganadora, incluida ésta, *aprenden* el patrón de entrada
- La neurona ganadora es la que tiene el vector de pesos más parecido al patrón de entrada
  - La diferencia entre estos dos vectores es pequeña
  - Para el vecindario, la diferencia es mayor
    - Al aplicarles el mismo coeficiente que a la ganadora el desplazamiento que sufrirían también es mayor.
    - Por ello se produce una disminución de  $\eta$  (*amortiguación*) según nos alejamos de la BMU



*Kohonen layer*



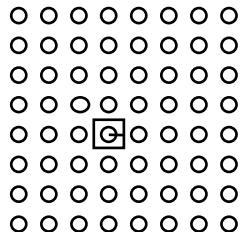
*Kohonen layer*



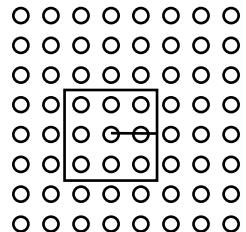
## 4.4 Aprendizaje



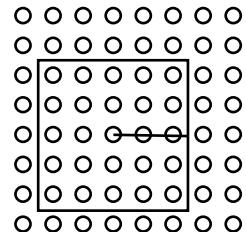
### ■ Tipos de vecindario



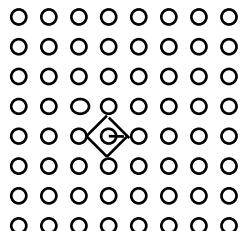
Vecindario=1



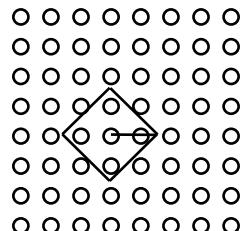
Vecindario=2



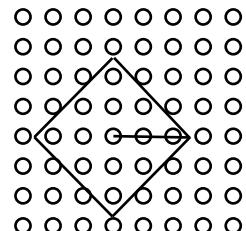
Vecindario=3



Vecindario=1

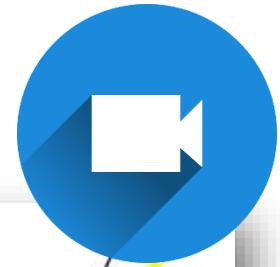


Vecindario=2



Vecindario=3

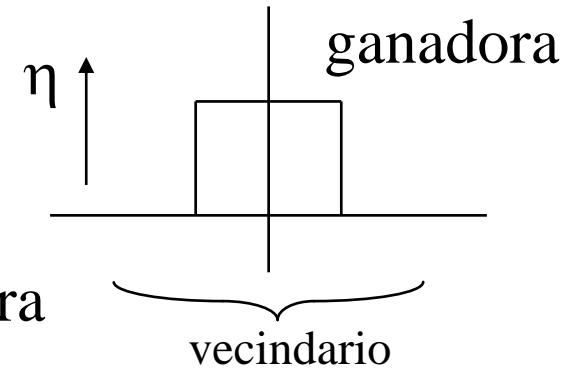
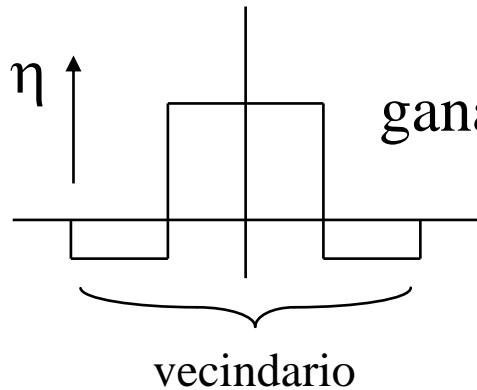
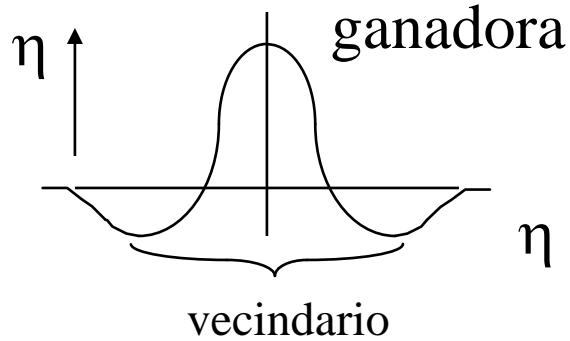
How self  
organizing maps  
algorithm works





## 4.4 Aprendizaje

- Las funciones de aprendizaje y vecindario están ligadas:
  - Se asigna el valor mayor al coeficiente de aprendizaje de la neurona ganadora (**aprenden positivamente**)
  - decreciéndolo con la distancia, dentro del vecindario (**amortiguación de  $\eta$** )
  - hasta llegar a cero o valores negativos cerca de la frontera de éste (**aprenden negativamente**)
  - siendo nulo para el resto de las neuronas de la capa





## 4.5 Mapas topológicos

- La ligazón entre aprendizaje y vecindario hace que la red agrupe sus neuronas en *mapas topológicos*, cada uno de ellos especializado en el reconocimiento de un tipo de patrón.

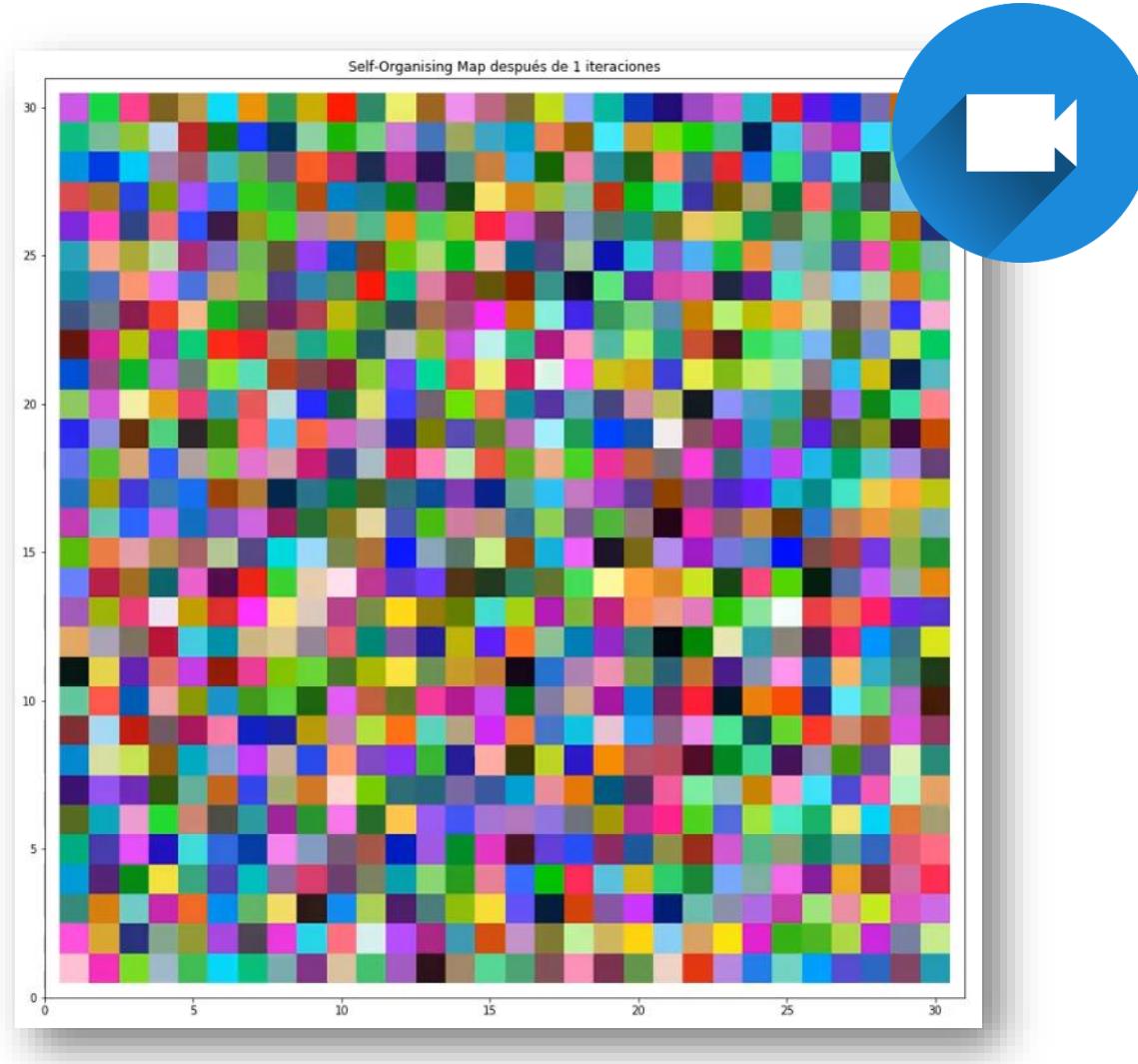
- La capa de Kohonen preserva el orden y compacta la representación de una nube de datos en un *espacio N-dimensional* proyectándola sobre un *mapa bidimensional*.

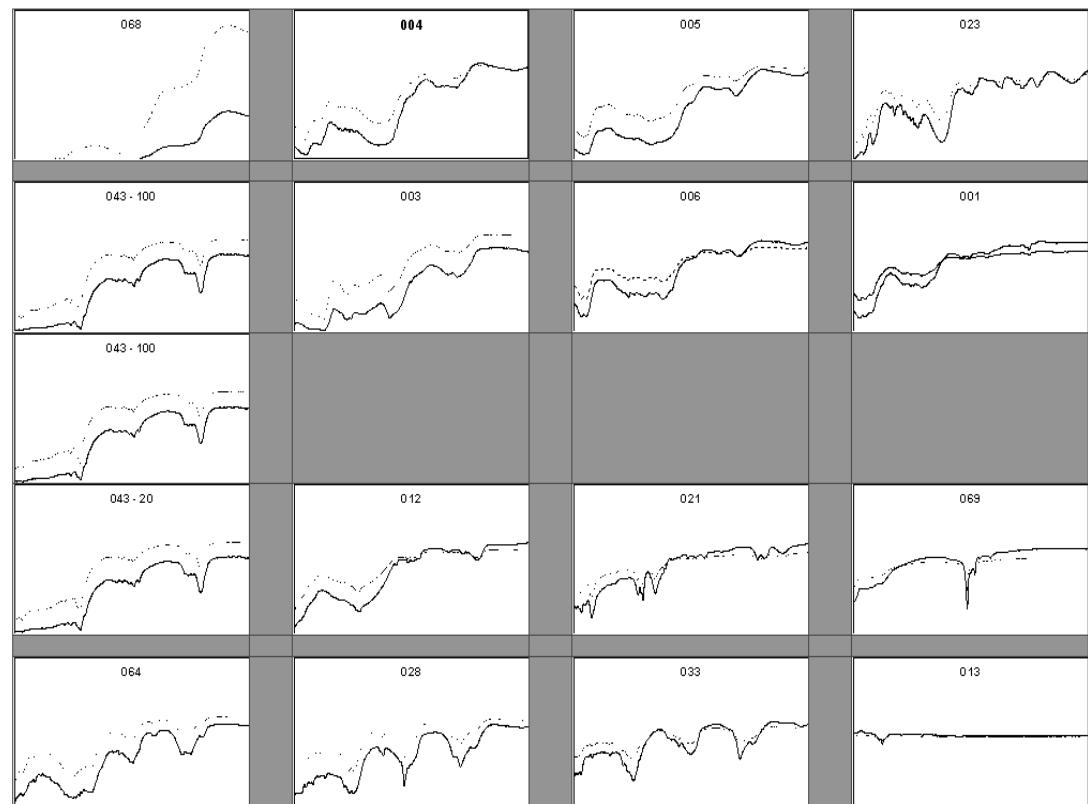
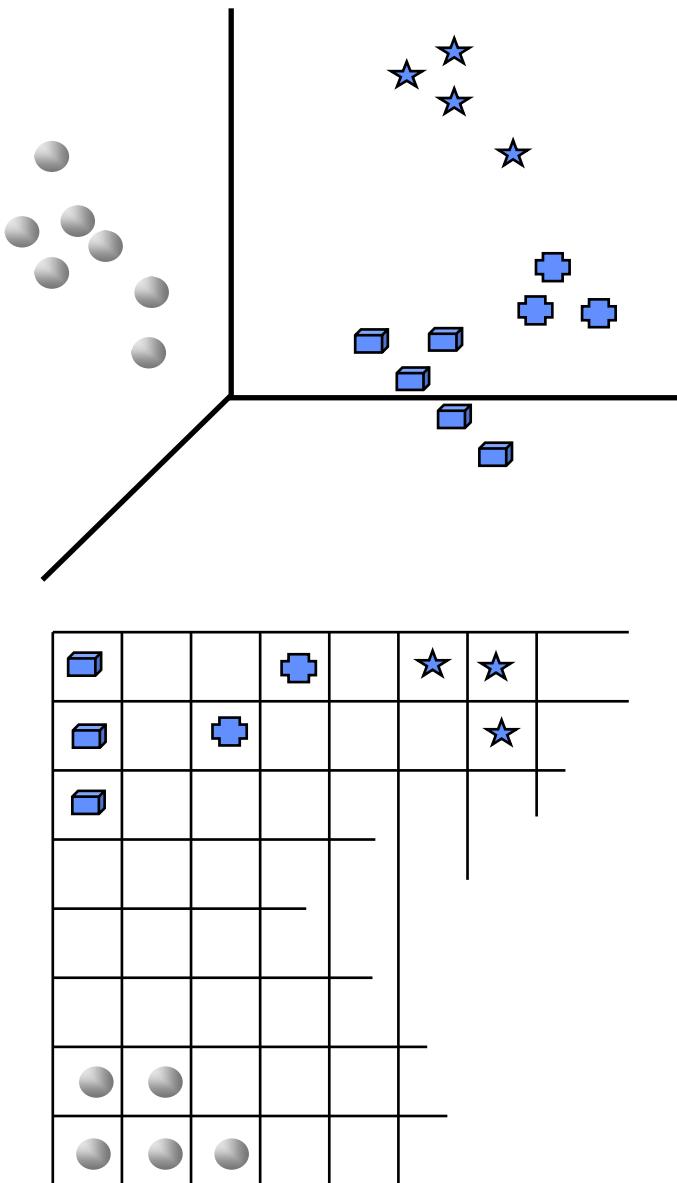
$$f: \mathbb{R}^N \rightarrow \mathbb{R}^2$$

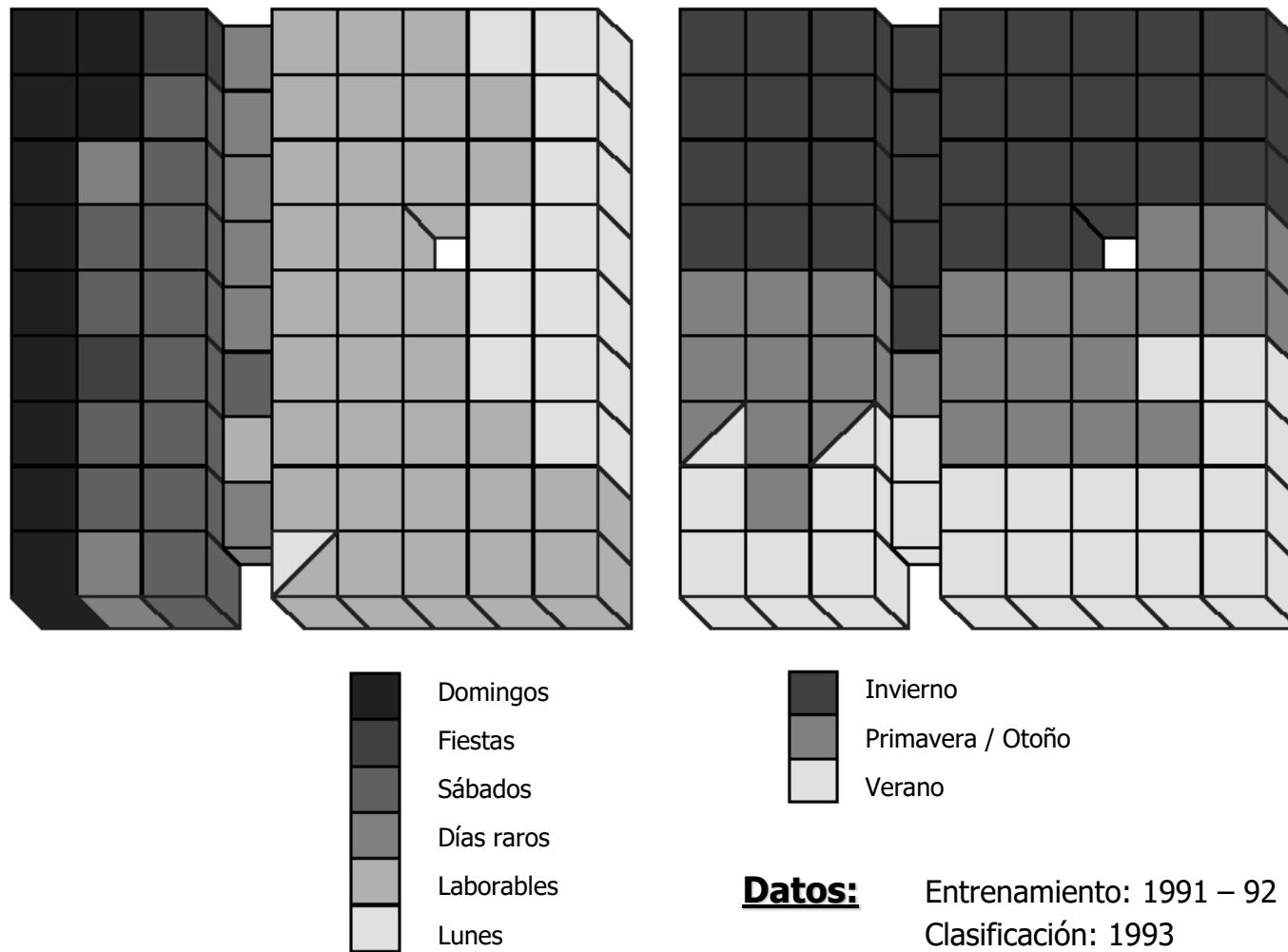
- Se garantiza que
  - Dos patrones cercanos en el espacio inicial producen salidas cercanas en el mapa bidimensional (*mapa topológico*)
  - Neuronas topológicamente cercanas en la capa de Kohonen son sensibles a *entradas físicamente similares* (cercanas en el espacio)

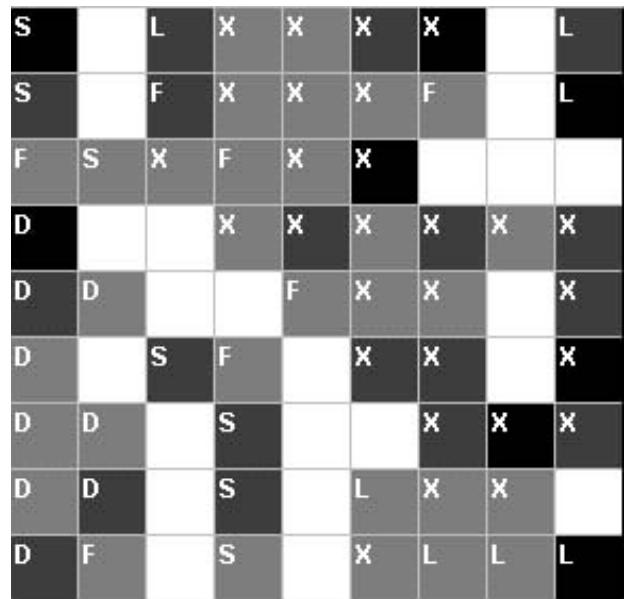


## 4.4 Aprendizaje

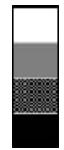








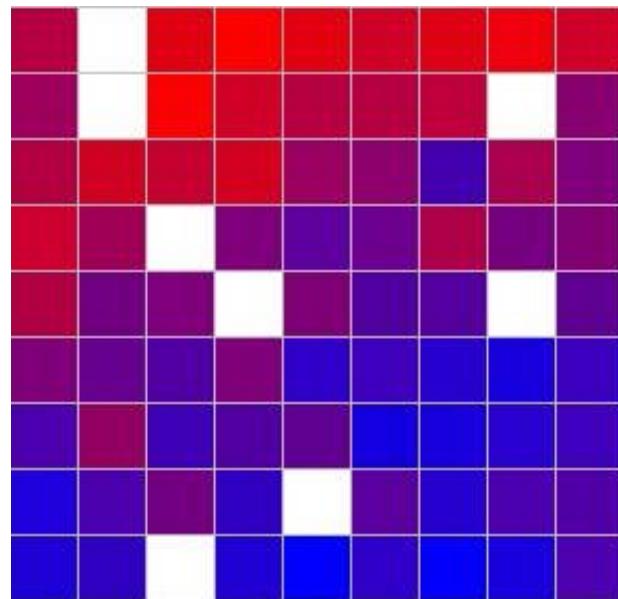
**Activaciones**



0 - 1  
2 - 5  
6 - 9  
10 -

**Tipo de Día**

L: Lunes  
X: Laborables  
S: Sábados  
D: Domingos  
F: Festivos



**Temperaturas**



Más calor  
Más frío

**Datos:**

Entrenamiento: 1993  
Clasificación: 1994



## 4.6 Parámetros

- Los parámetros de un SOM son:
  - Periodo:  $p$  (nº total de presentaciones)
  - Coeficiente de aprendizaje:  $\eta_0, \eta_f$
  - Vecindario:  $V_0, V_f$
  - Amortiguación:  $A_i$
  - Número de neuronas del lado de la capa de Kohonen:  $k$
- Iteración ( $t$ ): Presentación de una muestra a la red. Cada iteración incluye
  - Activación de una neurona por cada vector de entrada: Neurona ganadora o BMU
  - Aprendizaje: El vector de pesos de la BMU y de su vecindario es modificado de acuerdo a la regla de aprendizaje

Según aumentan las iteraciones algunos de estos parámetros van variando

## 4.6 Parámetros

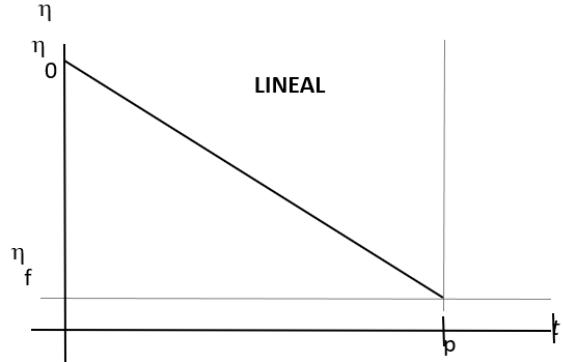


### Número de Presentaciones (t)

- El número de presentaciones o “tiempo” transcurrido  $t$  va aumentando linealmente hasta alcanzar  $p$ .

### Variación del Coeficiente de aprendizaje ( $\eta_t$ )

- Disminuye linealmente de  $\eta_0$  (para  $t=0$ ) a  $\eta_f$  (para  $t=p$ ).



Cuando  $\eta_f = 0$

$$\eta(t) = \eta_0 \left(1 - \frac{t}{p}\right)$$

### Variación del Vecindario ( $v_t$ )

- Disminuye linealmente de  $v_0$  ( $t=0$ ) a  $v_f$  ( $t=p$ ).
- Si  $v_f=0$  debe de incluir una neurona

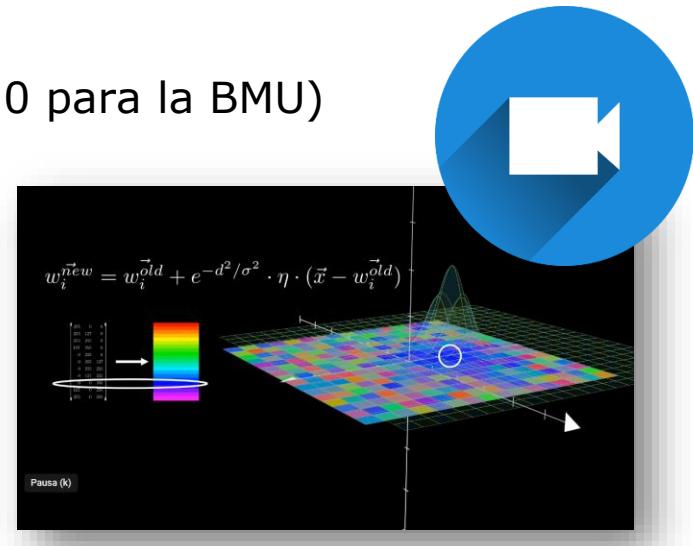
$$v(t) = 1 + v_0 \left(1 - \frac{t}{p}\right)$$



## Amortiguación (a)

- Las neuronas del vecindario de la BMU también actualizan sus pesos de forma proporcional a la distancia a la BMU
- Esta proporcionalidad no es lineal, sino gaussiana (campana centrada alrededor de la BMU) y viene dada por la fórmula
  - $a_j$ : amortiguación para la neurona j
  - $d_j$ : distancia euclídea 2D a la BMU ( $d_j=0$  para la BMU)
  - $v_t$ : vecindario actual

$$a_j = e^{-\frac{d_j^2}{2v_t^2}}$$



## Ecuación de aprendizaje:

$$W_{i,j}^{t+1} = W_{i,j}^t + \eta(t) a_j (X_i - W_{i,j}^t)$$

# 4.7 Modificaciones al proceso básico



## Refuerzo

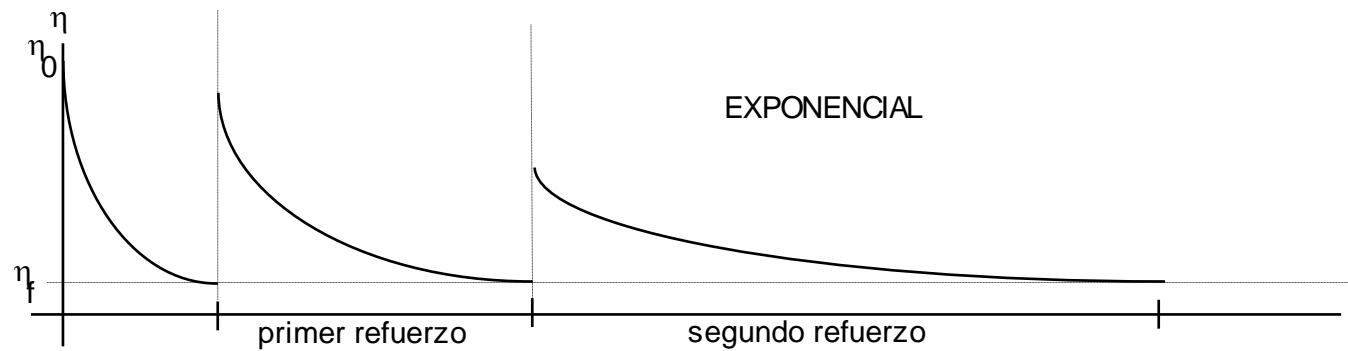
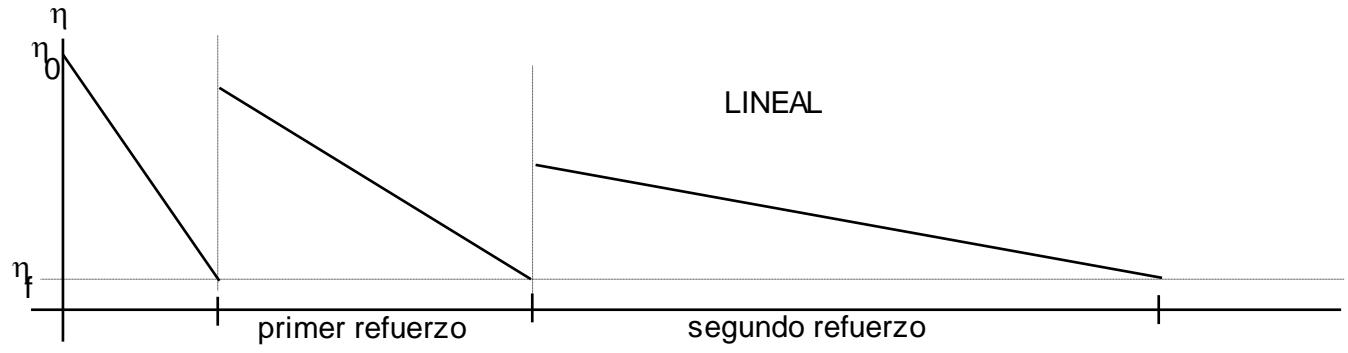
- Despues del primer periodo de aprendizaje, se realizan sucesivos períodos de refuerzo:
  - El número de refuerzos es  $r$
  - Se reduce el valor de  $\eta_0$ , multiplicándolo por un nuevo parámetro llamado  $\eta_{shrink}$
  - Se amplia el valor de  $p$ , multiplicándolo por un nuevo parámetro llamado  $T_{factor}$
  - el vecindario toma el valor constante 1 (lo que quiere decir que en la fase de refuerzos sólo la neurona ganadora ve modificados sus pesos).
- En cada periodo de refuerzo se cumple que:

$$p_r = p \cdot T_{factor}^r \quad \eta_{o,r} = \eta_0 \cdot \eta_{shrink}^r$$



## 4.7 Modificaciones al proceso básico

- Descenso en función del número de iteraciones:



## 4.7 Modificaciones al proceso básico



### Mecanismo de conciencia

- Función de penalización respecto a la frecuencia de activación
  - facilita que aquellas neuronas de la capa de Kohonen que no se activan ante ningún patrón de entrada tengan una mayor probabilidad de activación
  - y que las que se activan con excesiva frecuencia tengan una penalización que dificulte su activación
- Esto se realiza modificando la distancia neurona ↔ patrón a partir de la frecuencia de activación de cada neurona

$$D'_j = D_j - B_j$$

- Parámetros:
  - Factor de actualización de la frecuencia de activación:  $\beta_0$
  - Factor de penalización según la frecuencia de activación:  $\gamma_0$



1. Introducción
2. Clasificación
  1. El problema de clasificar
  2. Tipos de clasificadores
3. Redes que aprenden solas
  1. Aprendizaje NO supervisado
  2. Tipos de aprendizaje NO supervisado
4. Mapas Autoorganizativos
  1. Historia
  2. Arquitectura
  3. Procesamiento
  4. Aprendizaje
  5. Mapas topológicos
  6. Parámetros
  7. Modificaciones al proceso básico
5. Construyendo un clasificador



# 5. Construyendo un clasificador

- Elementos a tener en cuenta:
  1. Condiciones experimentales
  2. Selección de los parámetros
  3. Análisis del clasificador (calidad el mapa)

## 1. Condiciones experimentales

- Presentación random de muestras
- Matriz inicial de pesos random
- Pesos y patrones normalizados con la norma euclídea
- Suficiente número de datos (patrones) para el entrenamiento que cubran todas las posibles categorías



# 5. Construyendo un clasificador

## 2. Selección de parámetros (hiperparámetros)

El mapa se caracteriza por los valores de los hiperparámetros utilizados en el entrenamiento

- $k$ : Tamaño de la capa de Kohonen
- $\eta_0$ : Coeficiente de aprendizaje inicial ( $\eta_f=0$ )
- $v_0$ : Vecindario inicial, normalmente  $k$  o  $k/2$  ( $v_f=0$ )
- $p$ : Número de presentaciones
  - Refuerzo
  - Forma de la función de descenso de  $\eta(t)$

Decreasing Function Shape	Reinforcement	Presentations	Classification
Lineal	No	500 ; ... ; 10000	Bad
	Yes	6500 ; ... ; 19500	Good
Exponential	No	8000	Intermediate
	Yes	20800	Good
	Yes	104500	Good



# 5. Construyendo un clasificador

## 3. Análisis del clasificador (calidad del mapa)

- Métricas para medir la adecuación de cada mapa:

- Error de Cuantificación (*Quantization error*):

Distancia promedio entre cada vector de datos y su BMU (la neurona con el vector de peso más cercano a un vector de datos dado)

Mide la resolución del mapa.

- Distancia media de cada clase
- Distancia media de la red

$$\bar{d} = \frac{1}{N} \sum_{i=1}^N \|x_i - bmu\|$$

- *Error topológico*:

Proporción de todos los vectores de datos para los que la primera y la segunda BMU no son unidades adyacentes.

Mide la preservación de la topología.

- Homogeneidad de la distribución de clases

- $t(x_i) = 0$  si la primer y segunda BMU de un patrón son adyacentes
- $t(x_i) = 1$  en caso contrario

$$\bar{t} = \frac{1}{N} \sum_{i=1}^N t(x_i)$$



## 5. Construyendo un clasificador

- Mapa de clasificación
- Activaciones
  - Nº de clases (neuronas activadas)
  - Mapa de activaciones (histograma)
- Evolución temporal del mapa bidimensional



### Classification Analysis for different Kohonen side sizes

