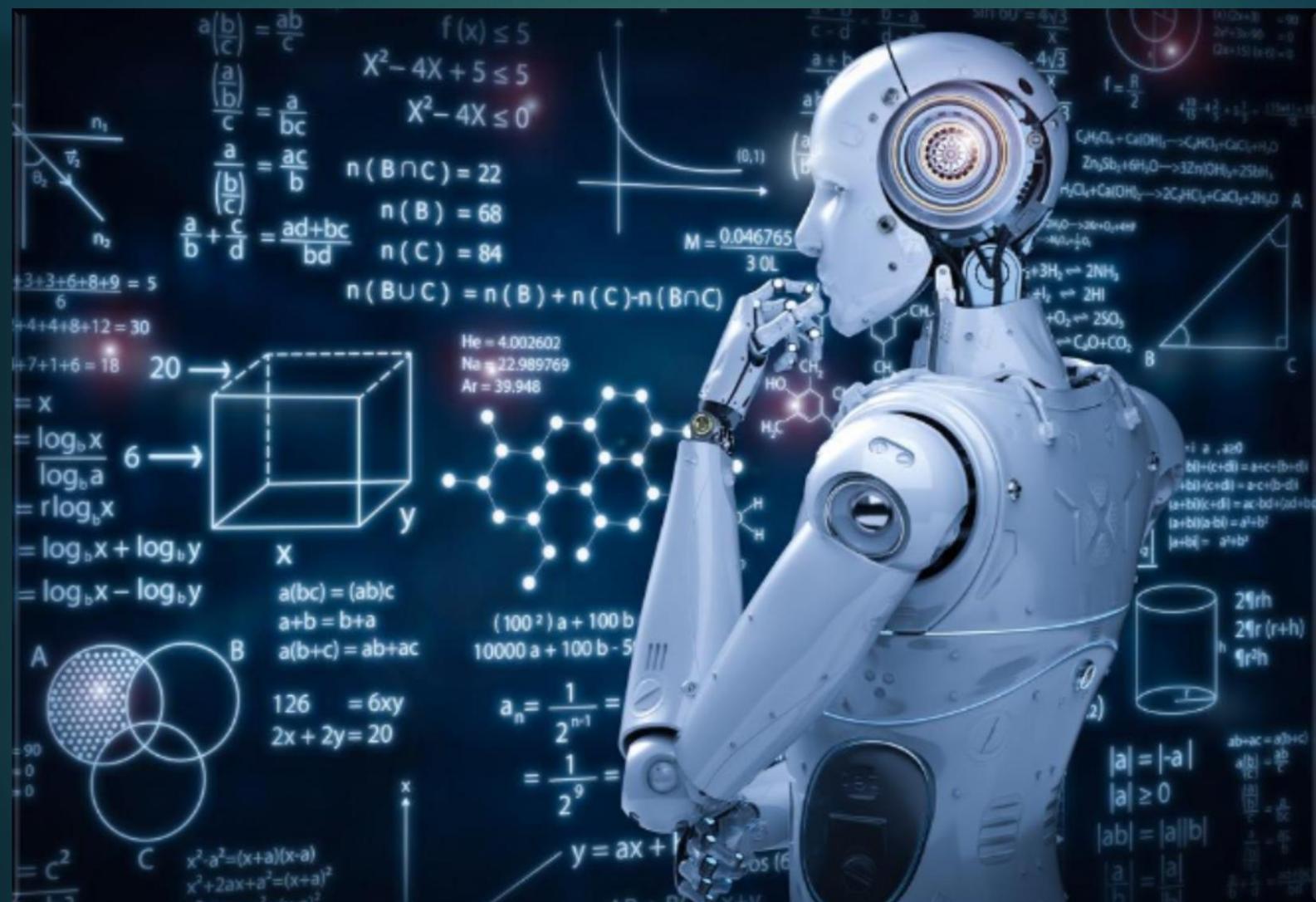
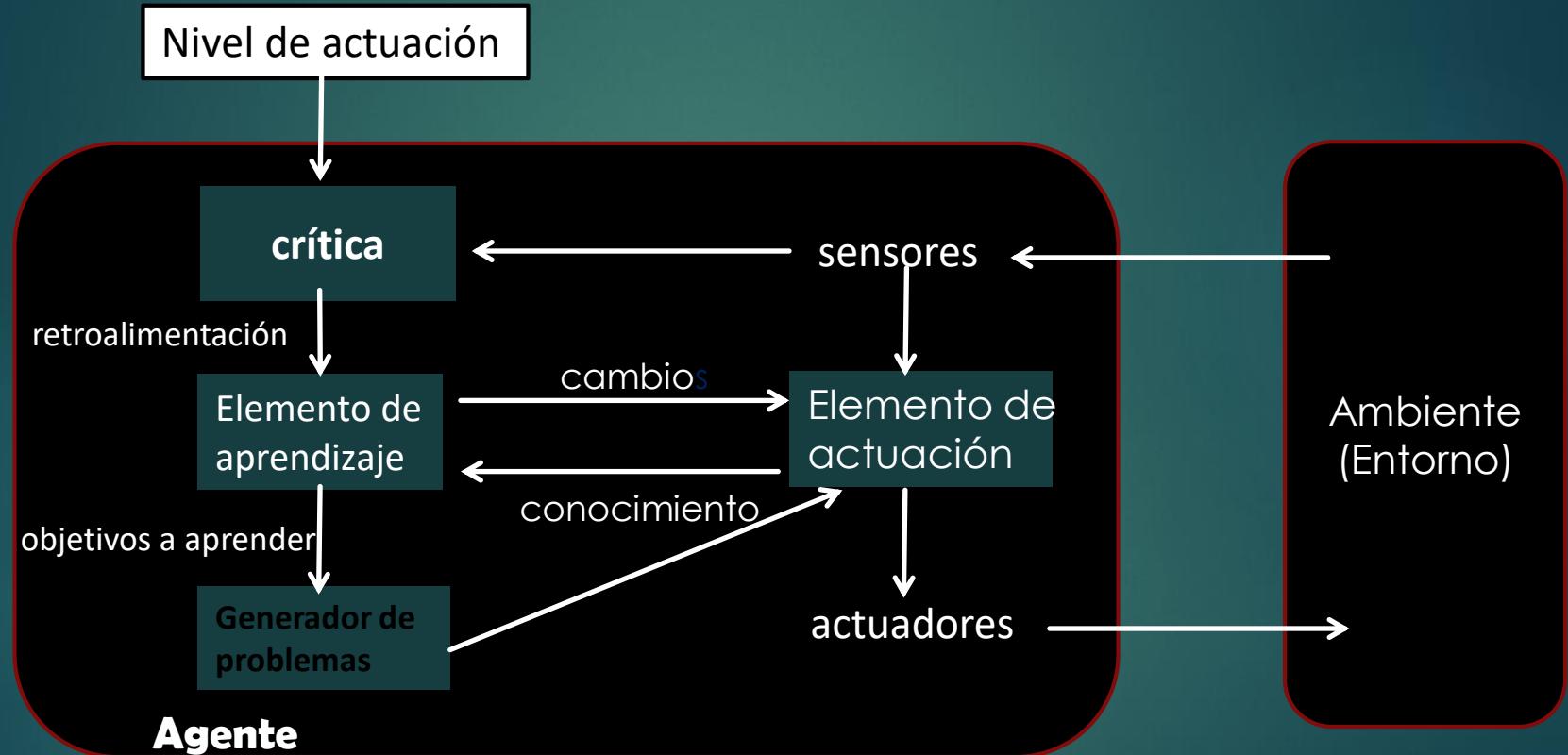


# APRENDIZAJE



Dra. Ing. Selva S. Rivera  
Prof. Titular

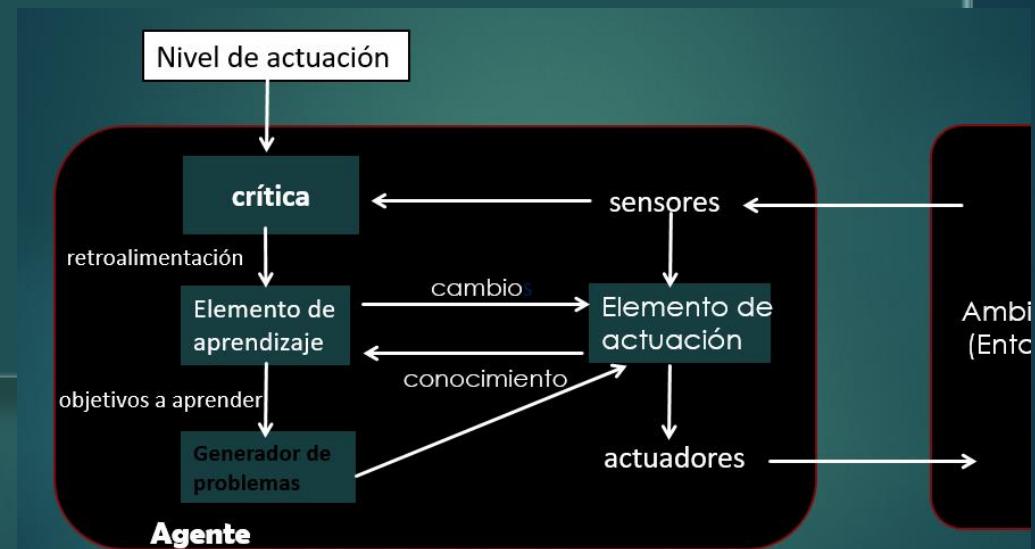
# AGENTES QUE APRENDEN



Utilizar las percepciones no sólo para actuar, sino también para mejorar la habilidad del agente para actuar en el futuro

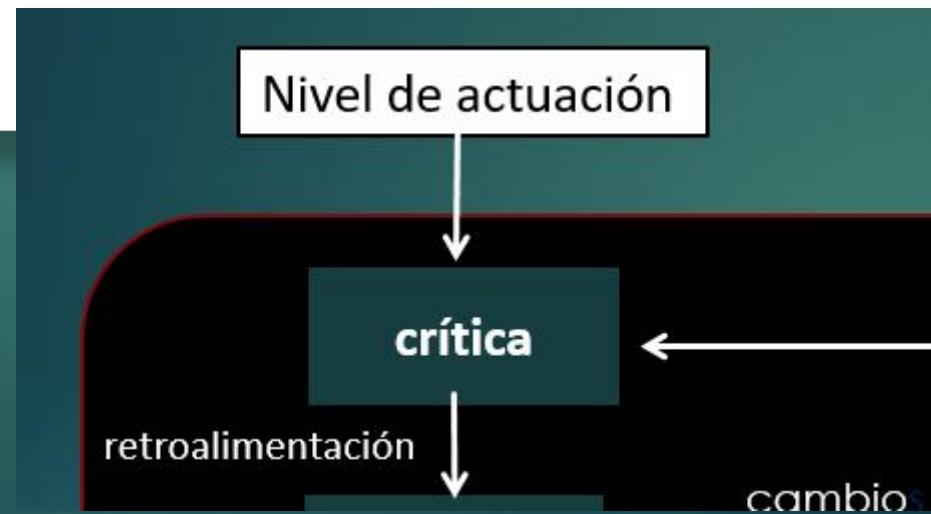
# AGENTES QUE APRENDE

- ▶ La distinción más importante entre el **elemento de aprendizaje** y el **elemento de actuación** es que el primero está responsabilizado de hacer mejoras y el segundo es responsable de la selección de acciones externas.
- ▶ El elemento de aprendizaje se alimenta con las **críticas** sobre la actuación del agente y determina cómo se debe modificar el elemento de actuación para proporcionar mejores resultados en el futuro.



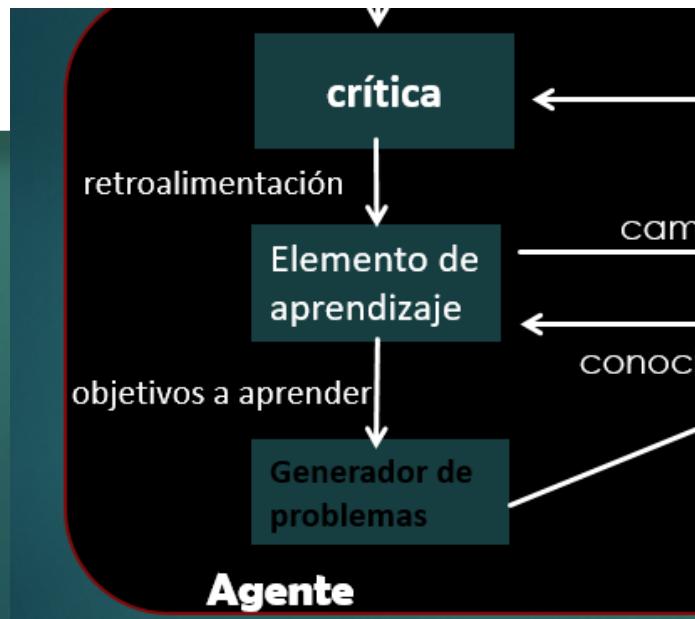
# AGENTES QUE APRENDEN

- ▶ La crítica indica al elemento de aprendizaje qué tal lo está haciendo el agente con respecto a un **nivel** de actuación fijo.
- ▶ La crítica es necesaria porque las percepciones por si mismas no prevén una indicación del éxito del agente.
- ▶ El nivel debe estar fuera del agente, ya que éste no debe modificarlo para satisfacer su propio interés.



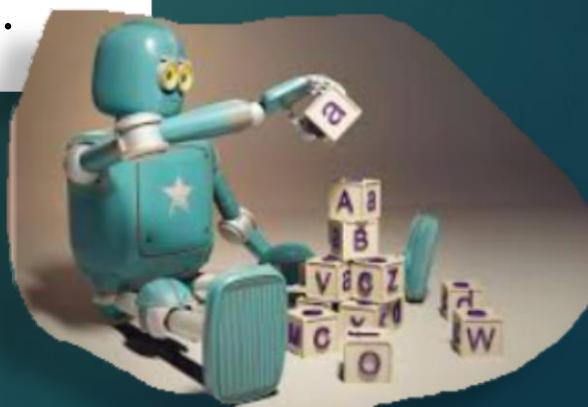
# AGENTES QUE APRENDEN

- ▶ El **generador de problemas** es responsable de sugerir acciones que lo guíen hacia experiencias nuevas e informativas.
- ▶ El generador de problemas le permite explorar al agente y llevar a cabo algunas acciones que no sean totalmente óptimas a corto plazo pero que le permiten descubrir acciones mejores a largo plazo.
- ▶ El trabajo de generador de problemas es sugerir estas acciones exploratorias.

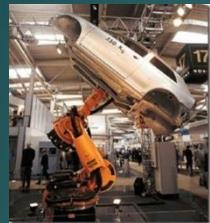


# AGENTES QUE APRENDE

- ▶ Los agentes que aprenden tienen una gran variedad de componentes que se pueden representar de muchas formas en los programas y por lo tanto hay una gran variedad de métodos de aprendizaje.
- ▶ El aprendizaje se puede definir entonces como el proceso de modificación de cada componente del agente, lo cual permite a cada componente comportarse más en consonancia con la información que se recibe y por lo tanto le permite mejorar su nivel medio de actuación.



# FORMAS DE APRENDIZAJE



Acciones

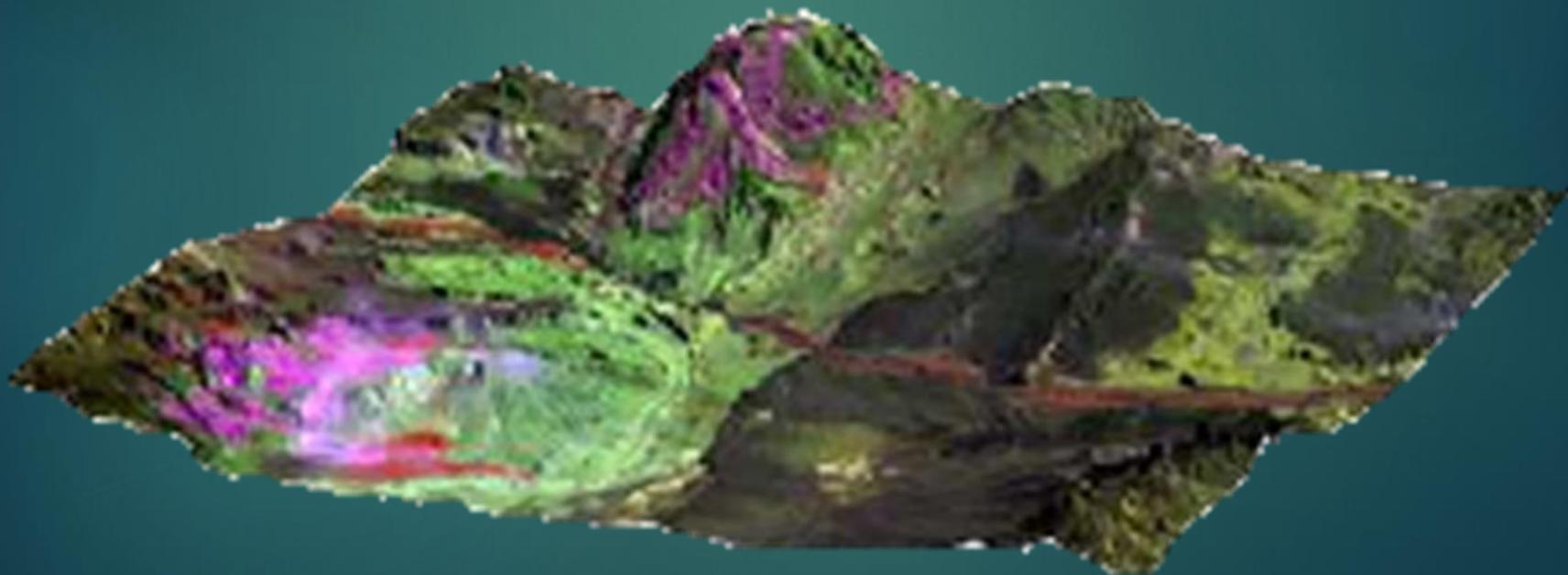
# APLICACIONES

- Sistema para conducir
- Optimización de Centrales Eléctricas
- Control de Trenes de Laminado en la Industria del Acero
- Optimización de Altos Hornos
- Optimización de la Producción de Cartón en la Industria Papelera
- Gestión de Alarmas en Plantas Petroquímicas
- Control de Calidad en la Fabricación de Electrodomésticos
- Optimización del Proceso de Producción de Cemento
- Control de Calidad de Materiales Fabricados Industrialmente
- Control Adaptativo para Optimización de Trayectorias de Robots Industriales
- Control de Calidad en la Fabricación de Cajas de Cambio en la Industria del Automóvil

# APRENDIZAJE NO SUPERVISADO

No hay etiquetas para los ejemplos

**Problema fundamental:** encontrar la estructura subyacente de un conjunto de datos

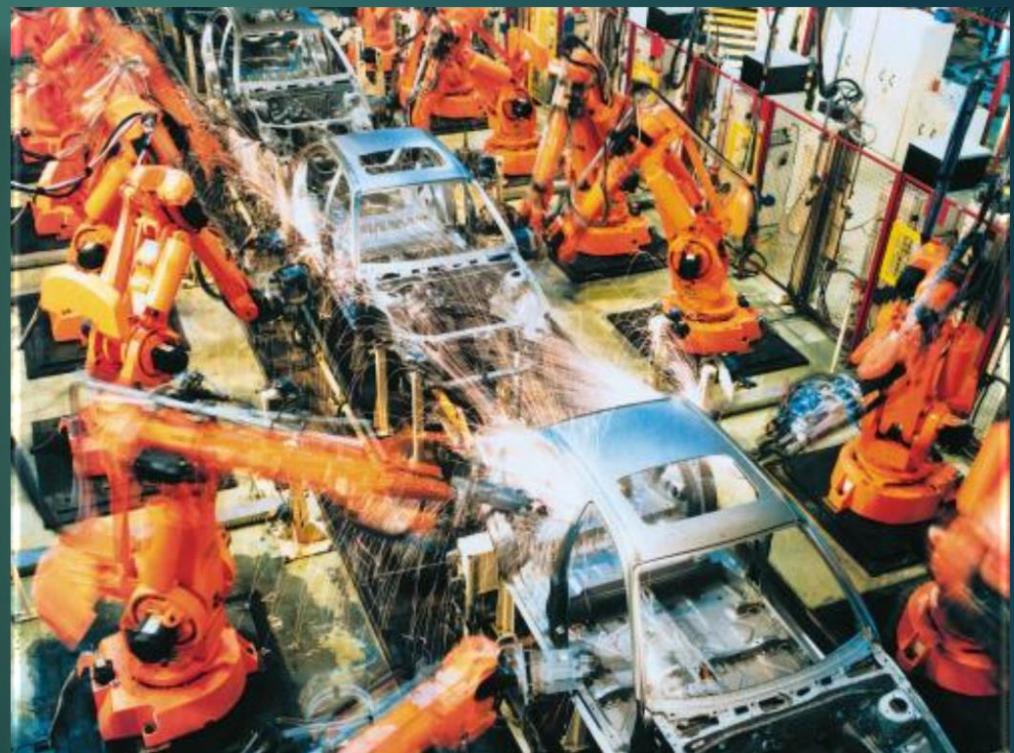


Modelo Digital de Elevación. Límite entre Mendoza y Neuquén (Río Colorado)

# APRENDIZAJE NO SUPERVISADO

## Ejemplos

- ▶ Visión artificial de robots de exploración (espacial, industrial, científica, etc.)
- ▶ Brazos robóticos que desempeñan tareas prefijadas, como por ejemplo, reconocer una pieza y realizar determinadas acciones sobre la misma.
- ▶ Sistemas de guiado para colaboración en la conducción de vehículos



# Aprendizaje no supervisado

- ▶ Consiste en aprender a partir de patrones de entradas para los que no se especifican los valores de sus salidas.
- ▶ En entornos reales con predominio de incertidumbre los agentes utilizan probabilidades y teoría de la decisión, pero primero, deben aprender sus teorías probabilísticas sobre el mundo a partir de la experiencia.

# APRENDIZAJE ESTADÍSTICO

- Los **datos** son evidencias, instanciaciones de todas o algunas de las variables aleatorias que describen el dominio.
- Las **hipótesis** son teorías probabilísticas sobre cómo funciona el dominio.



$h_1$ : 100% cereza

$h_2$ : 75% cereza + 25% limón

$h_3$ : 50% cereza + 50% limón

$h_4$ : 25% cereza + 75% limón

$h_5$ : 100% limón

**H (hipótesis): representa todos los tipos de bolsa**  
**h: cada tipo de bolsa**

**D (datos) : representa todos los datos**  
**d: valor observado**



# APRENDIZAJE bayesiano

**Regla Bayes**

$$P(h_i | \mathbf{d}) = \alpha P(\mathbf{d} | h_i) P(h_i)$$

El aprendizaje bayesiano calcula la probabilidad de cada hipótesis dados los datos y realiza predicciones sobre estas bases.

Las cantidades clave en el enfoque bayesiano son:

- las hipótesis a priori,  $P(h_i)$

**distribución a priori sobre H ej: <0.1, 0.2, 0.4, 0.2, 0.1>**

- y la verosimilitud de los datos dada una de las hipótesis,  $P(\mathbf{d} | h_i)$ .  
(se asume que las observaciones son independientes e idénticamente distribuidas)

$$P(\mathbf{d} | h_i) = \prod_j P(d_j | h_i)$$

j: n° de observación

$h_1$ : 100% cereza

$h_2$ : 75% cereza + 25% limón

$h_3$ : 50% cereza + 50% limón

$h_4$ : 25% cereza + 75% limón

$h_5$ : 100% limón

**Verosimilitud: una función de los parámetros de un modelo estadístico que permite realizar inferencias acerca de su valor a partir de un conjunto de observaciones.**

En base a los datos observados, cada valor de los parámetros nos dará una **puntuación** a la que llamamos **verosimilitud** y, como su propio nombre indica, nos ayuda a entender como de creíbles son dichos valores.

$$P(\mathbf{D} | h_i) = \prod_j P(d_j | h_i)$$

j: n° de observación

- $h_1$ : 100% cereza
- $h_2$ : 75% cereza + 25% limón
- $h_3$ : 50% cereza + 50% limón
- $h_4$ : 25% cereza + 75% limón
- $h_5$ : 100% limón

Si  $d_j$  = caramelo de limón ;  $j=4$  ;  $h_3$

Saqué 4 caramelos y todos eran de limón

$$P(h_3) = 0,4 \quad (\text{Prob. a priori})$$

La verosimilitud de los datos que un caramelo sea de limón dada la hipótesis 3 es  $P(\mathbf{D} | h_3)$

$$P(\mathbf{D} | h_3) = 0,4 \times 0,5 \times 0,5 \times 0,5 \times 0,5 = 0,25$$



# APRENDIZAJE BAYESIANO

distribución a priori sobre  $H$  ej:  $\langle 0.1, 0.2, 0.4, 0.2, 0.1 \rangle$

$h_1$ : 100% cereza

$h_2$ : 75% cereza + 25% limón

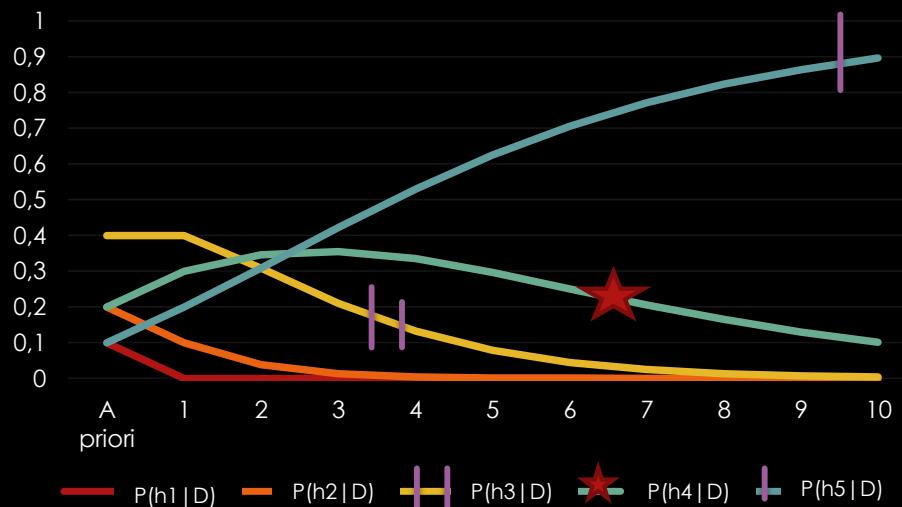
$h_3$ : 50% cereza + 50% limón

$h_4$ : 25% cereza + 75% limón

$h_5$ : 100% limón

Supongamos que  $H=h_5$  y que los 10 primeros caramelos son de limón

Probabilidad de las hipótesis a priori en función del nº de ejemplos



## Regla Bayes

$$P(h_i | d) = \alpha P(d | h_i) P(h_i) \quad ; \quad \alpha = 1 / P(D)$$



# APRENDIZAJE BAYESIANO

## Regla Bayes

$$P(h_i | d_j) = \alpha P(d_j | h_i) P(h_i) \quad ; \alpha = 1 / P(D)$$

distribución a priori sobre H ej: <0.1, 0.2, 0.4, 0.2, 0.1>

|    | d=0             | d=1   | alfa1 =   | 2 |
|----|-----------------|-------|-----------|---|
| hi | P(d=limon   hi) | P(hi) | P(hi   d) |   |
| 1  | 0               | 0,1   | 0         |   |
| 2  | 0,25            | 0,2   | 0,1       |   |
| 3  | 0,5             | 0,4   | 0,4       |   |
| 4  | 0,75            | 0,2   | 0,3       |   |
| 5  | 1               | 0,1   | 0,2       |   |

- $h_1$ :100% cereza  
 $h_2$ :75% cereza + 25% limón  
 $h_3$ :50% cereza + 50% limón  
 $h_4$ :25% cereza + 75% limón  
 $h_5$ :100% limón

$$P(D) = \sum_{i=1}^5 P(dj|h_i).P(h_i|dj)$$

$$d_0=0 \quad ; \quad P(D) = 0 * 0,1 + 0,25 * 0,2 + 0,5 * 0,4 + 0,75 * 0,2 + 1 * 0,1 = 0,5$$

Aplicando la Regla de Bayes:

$$P(h_2 | d=1) = \alpha . P(d|h_2).P(h_2) = 2 . 0,25 . 0,2 = 0,1$$



# APRENDIZAJE BAYESIANO

**Regla Bayes**

$$P(h_i | d) = \alpha P(d | h_i) P(h_i) \quad ; \alpha = 1 / P(d)$$

distribución a priori sobre H ej: <0.1, 0.2, 0.4, 0.2, 0.1>

| hi | P(d=limon   hi) | P(hi   d) | d = 1   | d=2 | alfa2= | 1,5384615<br>4 |
|----|-----------------|-----------|---------|-----|--------|----------------|
| 1  | 0               | 0         | 0,00000 |     |        |                |
| 2  | 0,25            | 0,1       | 0,03846 |     |        |                |
| 3  | 0,5             | 0,4       | 0,30769 |     |        |                |
| 4  | 0,75            | 0,3       | 0,34615 |     |        |                |
| 5  | 1               | 0,2       | 0,30769 |     |        |                |

$h_1$ :100% cereza  
 $h_2$ :75% cereza + 25% limón  
 $h_3$ :50% cereza + 50% limón  
 $h_4$ :25% cereza + 75% limón  
 $h_5$ :100% limón

$$P(\mathbf{D}) = \sum_{i=1}^5 P(dj|h_i).P(h_i|dj)$$

$$d_1=\text{limón} \quad P(\mathbf{D}) = 0 * 0 + 0,25 * 0,1 + 0,5 * 0,4 + 0,75 * 0,3 + 1 * 0,2 = 0,65$$

Aplicando la Regla de Bayes:

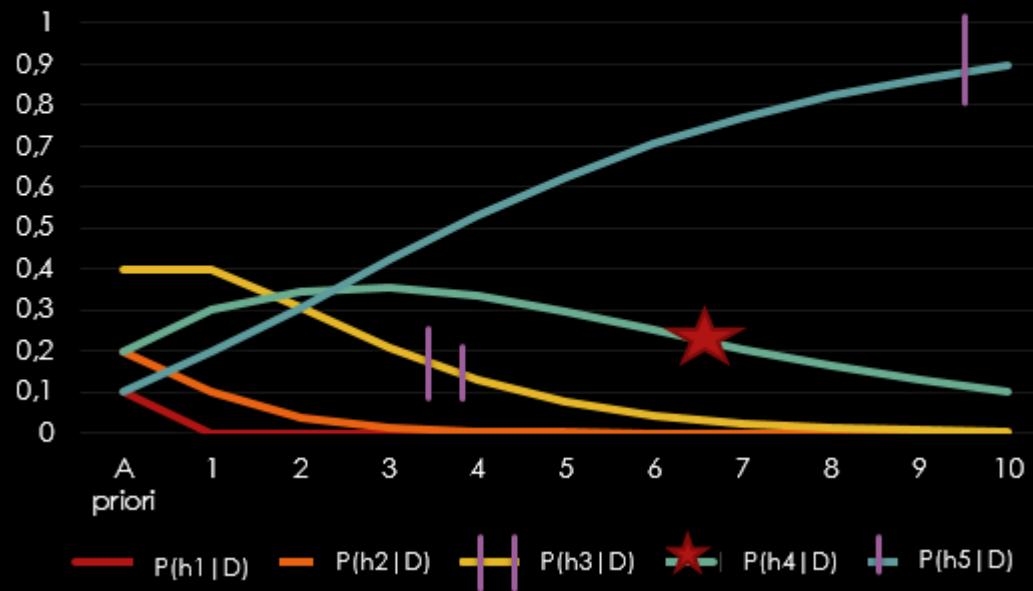
$$P(h_2 | d=2) = \alpha . P(d|h_2).P(h_2) = 1,5384 . 0,25 . 0,1 = 0,03846$$



# APRENDIZAJE BAYESIANO

| d        | P(h1   d) | P(h2   d) | P(h3   d) | P(h4   d) | P(h5   d) |
|----------|-----------|-----------|-----------|-----------|-----------|
| A priori | 0,1       | 0,2       | 0,4       | 0,2       | 0,1       |
| 1        | 0         | 0,1       | 0,4       | 0,3       | 0,2       |
| 2        | 0         | 0,038     | 0,307     | 0,346     | 0,308     |
| 3        | 0         | 0,013     | 0,210     | 0,355     | 0,421     |
| 4        | 0         | 0,004     | 0,132     | 0,334     | 0,529     |
| 5        | 0         | 0,001     | 0,078     | 0,296     | 0,624     |
| 6        | 0         | 0,0003    | 0,044     | 0,251     | 0,705     |
| 7        | 0         | 9,4E-05   | 0,024     | 0,206     | 0,770     |
| 8        | 0         | 2,5E-05   | 0,013     | 0,165     | 0,822     |
| 9        | 0         | 6,6E-06   | 0,007     | 0,129     | 0,863     |
| 10       | 0         | 1,7E-06   | 0,003     | 0,101     | 0,896     |

Probabilidad de las hipótesis a priori en función del n° de ejemplos





# APRENDIZAJE BAYESIANO

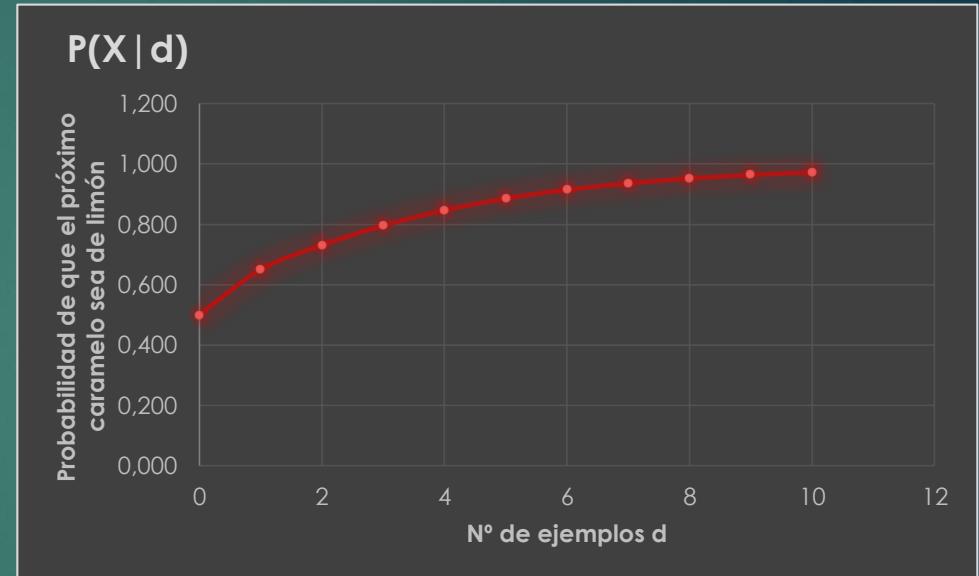
limón

d=0

| hi       | P(d hi) | P(hi d) |      |
|----------|---------|---------|------|
| 1        | 0       | 0,1     | 0    |
| 2        | 0,25    | 0,2     | 0,05 |
| 3        | 0,5     | 0,4     | 0,2  |
| 4        | 0,75    | 0,2     | 0,15 |
| 5        | 1       | 0,1     | 0,1  |
| $P(X d)$ |         |         | 0,5  |

d=1

| hi       | P(d hi) | P(hi d) |       |
|----------|---------|---------|-------|
| 1        | 0       | 0       | 0     |
| 2        | 0,25    | 0,1     | 0,025 |
| 3        | 0,5     | 0,4     | 0,2   |
| 4        | 0,75    | 0,3     | 0,225 |
| 5        | 1       | 0,2     | 0,2   |
| $P(X d)$ |         |         | 0,65  |



Una predicción sobre una cantidad desconocida X se calcula como:

$$P(X|\mathbf{d}) = \sum_i P(d|h_i)P(h_i|\mathbf{d})$$

# APRENDIZAJE BAYESIANO

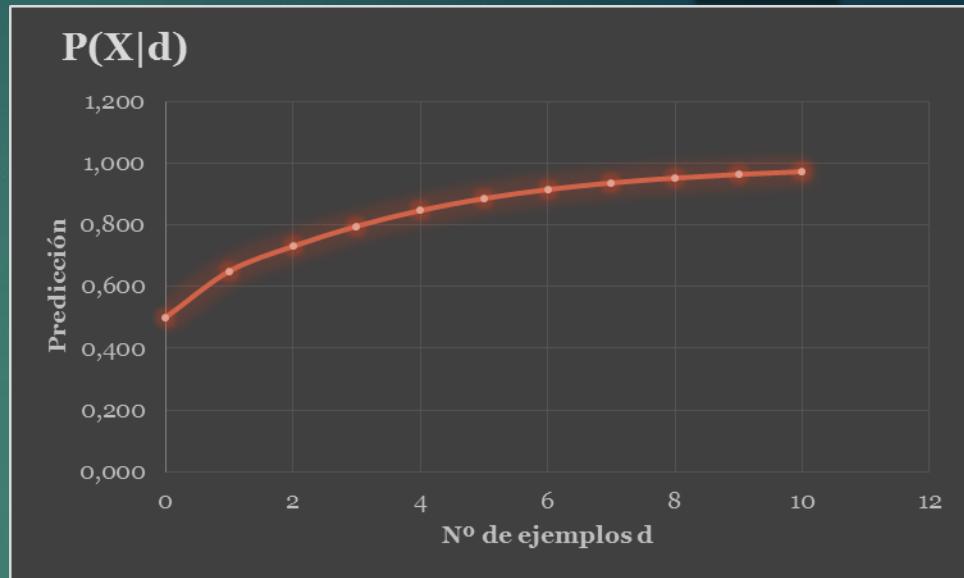
A la larga, la verdadera hipótesis domina la predicción bayesiana.

La probabilidad a posteriori de cualquier hipótesis falsa finalmente desaparecerá.

La predicción bayesiana es **óptima**, tanto si el conjunto de datos es pequeño como si es grande.

Cuando el espacio de hipótesis es grande (o infinito) el cálculo de la predicción es tratable pero en la mayoría de los casos hay que recurrir a métodos aproximados.

Una aproximación muy común, que se adopta normalmente en la ciencia, es hacer predicciones basadas en una única hipótesis, **la más probable**, es decir una  $h_i$  que maximice  $P(h_i | d)$ . (Máximo a posteriori o hipótesis MAP)



# Hipótesis de máxima verosimilitud

El aprendizaje de parámetros con datos completos supone encontrar los parámetros numéricos para un modelo probabilístico cuya estructura está fijada.

Los datos son completos cuando incluyen valores para cada variable del modelo de probabilidad que está siendo aprendido.

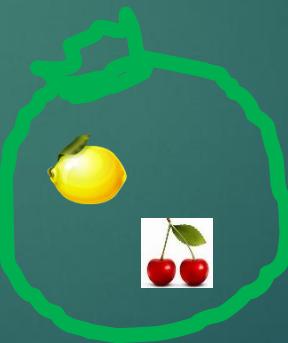
$P(F=cereza)$

$\theta$

Sabor

Desenvolvemos **N** caramelos

**c** son de cereza y **I = N - c** son de limón



parámetro  $\theta$ : proporción de caramelos de cereza;  $h_\theta$  = hipótesis

# Hipótesis de máxima verosimilitud



Desenvolvemos **N** caramelos

**c** son de cereza y **l = N - c** son de limón

La verosimilitud del conjunto es:

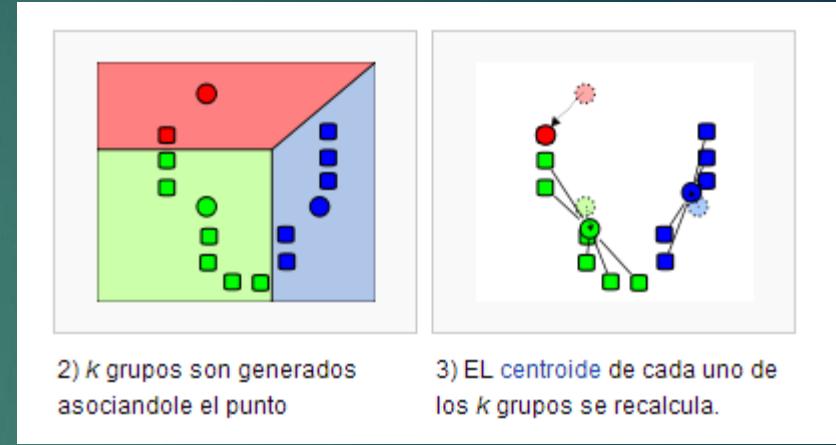
$$P(d|h_\theta) = \prod_{j=1}^N P(d_j|h_\theta) = \theta^c \cdot (1-\theta)^l$$

La hipótesis de máxima verosimilitud viene dada por el valor  $\theta$  que maximiza esta expresión:

$$L(d|h_\theta) = \log P(d|h_\theta) = \sum_{j=1}^N \log P(d_j|h_\theta) = c \log \theta + l \cdot \log(1-\theta)$$
$$\frac{dL(d|h_\theta)}{d\theta} = \frac{c}{\theta} - \frac{l}{1-\theta} = 0 \Rightarrow \theta = \frac{c}{c+l} = \frac{c}{N}$$

El problema de aprendizaje se descompone en problemas separados para cada uno de los parámetros

# Algoritmo no supervisado : agrupamiento



Se utiliza Distancia Euclidiana para el cálculo del centroide.

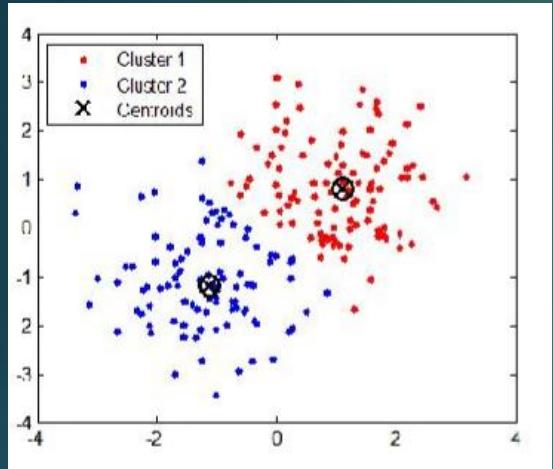
## AGRUPAMIENTO K-means

Como se trata de un algoritmo heurístico, no hay ninguna garantía de que convergen al óptimo global.

El resultado puede depender de los grupos iniciales.

En el peor de los casos, *k*-means puede ser muy lento para converger: en particular, se ha demostrado que existen conjuntos de determinados puntos, incluso en 2 dimensiones, en la que *k*-means toma tiempo exponencial

# Algoritmo K-means

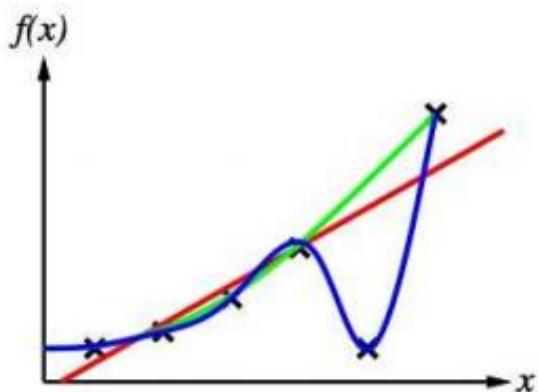
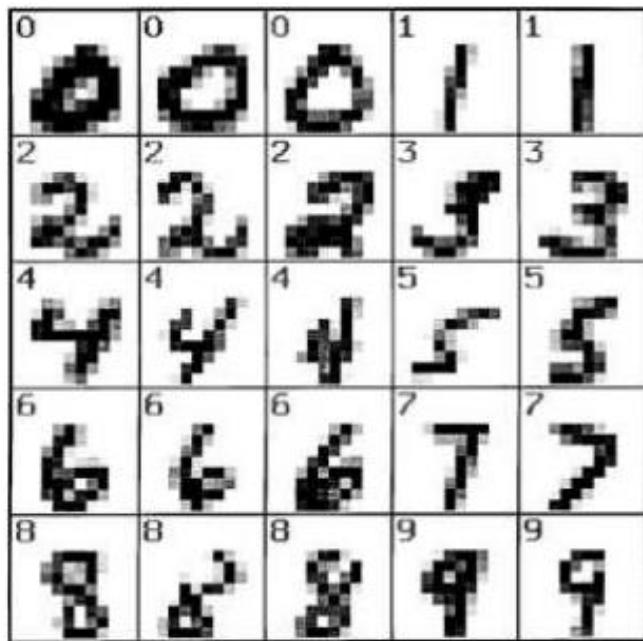


El nombre K-means viene porque representa cada uno de los “clusters” por la media (o media ponderada) de sus puntos, es decir, por su centroide.

- 1- Elegir aleatoriamente “k” objetos que forman así los clusters iniciales. Para cada cluster “k”, el valor inicial del centro es  $=x_i$ ; con los  $x_i$  únicos objetos de  $D_n$  pertenecientes al cluster.
- 2- Para cada objeto  $x$ , el prototipo que se le asigna es el que está más próximo al objeto, según una medida de distancia, (comúnmente la euclídea)
- 3- Una vez que todos los objetos son colocados, se recalculan los centroides “k”.
- 4- Se repiten los puntos 2 y 3 hasta que no se hagan más reasignaciones.

# APRENDIZAJE SUPERVISADO

- **Problema fundamental:** encontrar una función que relacione un conjunto de entradas con un conjunto de salidas
- Tipos principales de problemas:
  - Clasificación
  - Regresión

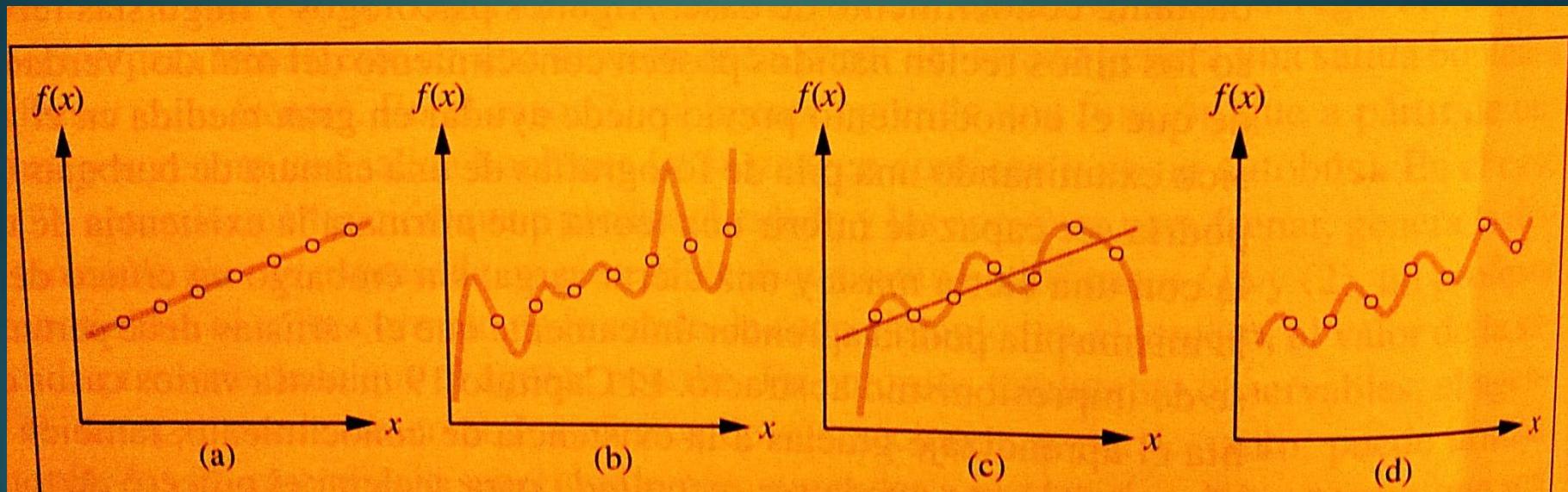


# APRENDIZAJE INDUCTIVO

- ▶ Algoritmo de aprendizaje supervisado determinístico que recibe como entrada el valor correcto para determinados valores de una función desconocida y debe averiguar cuál es la función o aproximarla.
- ▶ La tarea inductiva pura o inducción es: “Dada una colección de ejemplos de  $f$ , devolver una función  $h$  que aproxime a  $f$ .”
- ▶ La función  $h$  se denomina **hipótesis**.
- ▶ No es fácil determinar si  $h$  es una buena aproximación de  $f$ .
- ▶ Una buena  $h$  estará bien generalizada si puede predecir ejemplos que no se conocen.

# ¿Cómo elegir entre múltiples $h$ consistentes?

Respuesta: navaja de Ockham  
es preferible la  $H$  consistente con los datos  
que sea más sencilla.



Espacio de  $H$ : conjunto de hipótesis que se van a considerar (conj. de polinomios)

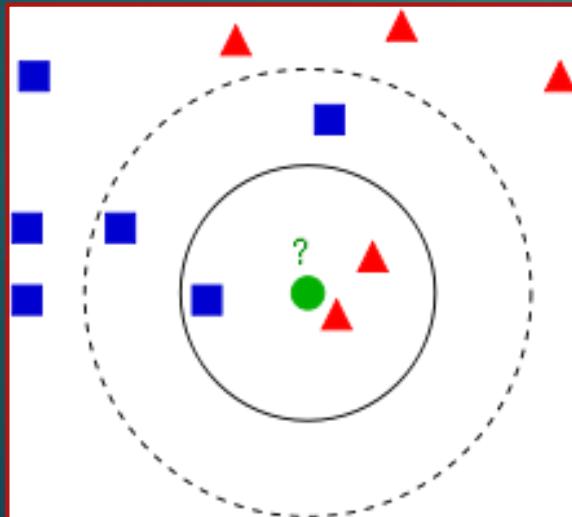
$H$  consistente: verifica todos los datos.

Ockham «afeitaba como una navaja las barbas de [Platón](#)»

# Knn (K nearest neighbours)

Es un método de clasificación supervisada que estima el valor de la probabilidad a posteriori de que un elemento  $x$  pertenezca a la clase  $C_j$  a partir de la información proporcionada por el conjunto de prototipos.

Este algoritmo se utiliza dentro de la teoría de Reconocimiento de patrones para la clasificación de objetos.



**Los ejemplos de entrenamiento son vectores en un espacio multidimensional.**

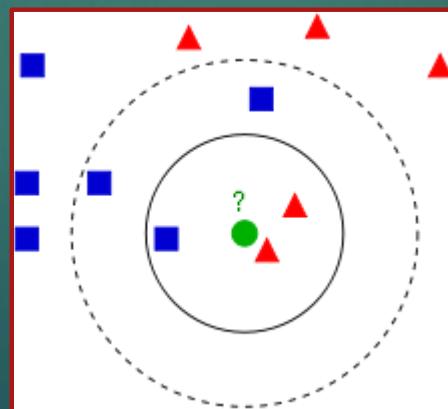
**El espacio es particionado en regiones por localizaciones y etiquetas de los ejemplos de entrenamiento.**

**Un punto en el espacio es asignado a la clase  $C$  si ésta es la clase más frecuente entre los  $k$  ejemplos de entrenamiento más cercano.**

# Algoritmo Knn

Se tienen 3 conjuntos de datos:

- el conjunto de entrenamiento que se utiliza para el aprendizaje del clasificador
- el conjunto de validación
- el conjunto de prueba (datos no empleados durante el entrenamiento)

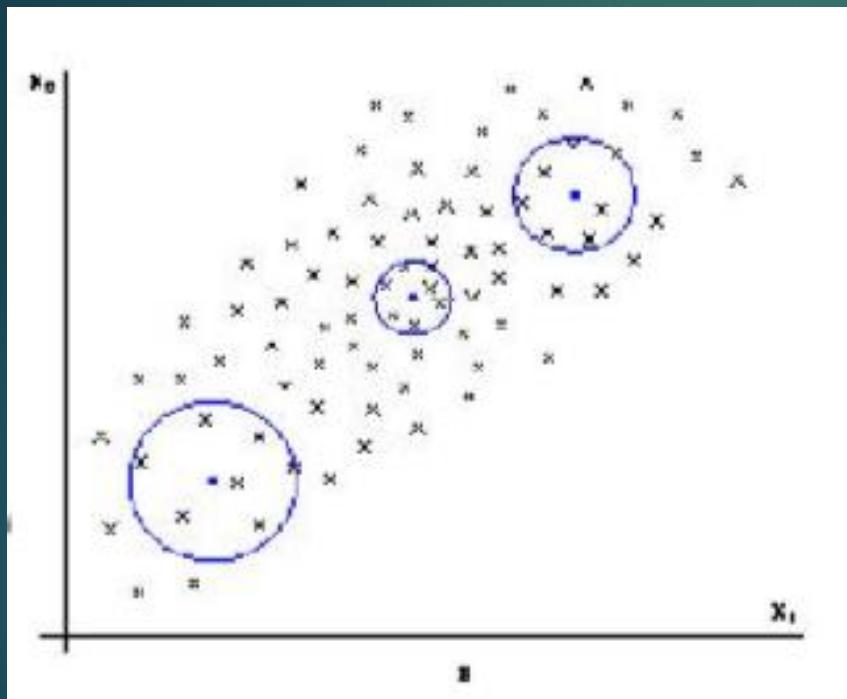


# Algoritmo Knn



- 1- conjunto de entrenamiento, conjunto de vectores con identificación de la clase asociada a la que corresponde.
- 2- K es una constante definida por el usuario. Para clasificar un nuevo vector, se calcula la Distancia Euclídea a todos los vectores entrenamiento y nos quedamos con las “K” muestras mas cercanas, clasificando la muestra de entrenamiento en la clase más frecuente a la que pertenecen los “K” vecinos obtenidos anteriormente.
- 3- Luego se realiza el mismo proceso con los datos de validación. Se calcula el porcentaje de clasificación sobre los ejemplos de este conjunto (desconocidos en la tarea de aprendizaje) para conocer su poder de generalización.

# Elección de k



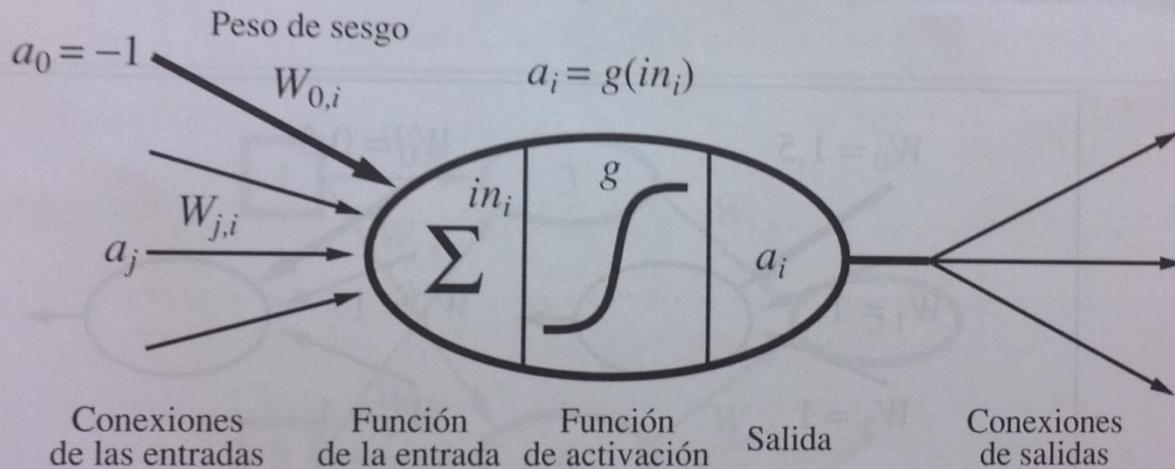
El área del círculo que encierra un nº fijo de puntos, k, es menor en regiones densamente pobladas que en regiones donde los puntos están más dispersos.

La mejor elección de k depende fundamentalmente de los datos.

Generalmente, valores grandes de k reducen el efecto de ruido en la clasificación, pero crean límites entre clases parecidas.

El caso especial en que la clase es predicha para ser la clase más cercana al ejemplo de entrenamiento (cuando k=1) es llamada Nearest Neighbor Algorithm, Algoritmo del vecino más cercano.

# REDES NEURONALES



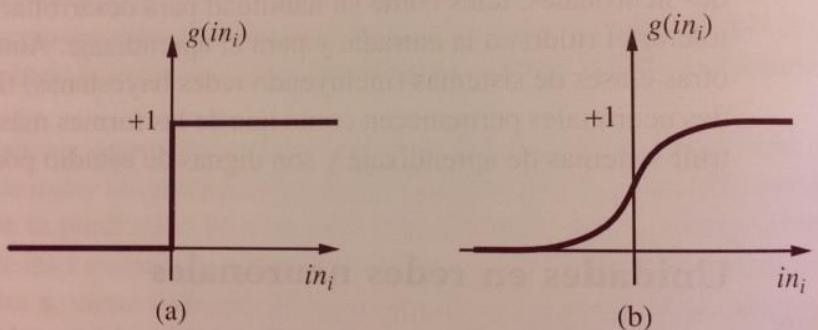
**Figura 20.15** Un modelo matemático sencillo para una neurona. La activación de la salida de la unidad es  $a_i = g(\sum_{j=0}^n W_{j,i} a_j)$ , donde  $a_j$  es la activación de la salida de la unidad  $j$  y  $W_{j,i}$  es el peso de la conexión de la unidad  $j$  a esta unidad.

$$in_i = \sum_{j=0}^n W_{j,i} a_j$$

$$a_i = g(in_i) = g\left(\sum_j^n W_{j,i} a_j\right)$$

# REDES NEURONALES

La función de activación  $g$  se diseña con el objetivo de que la unidad esté “activa” (cercana a +1) cuando se proporcionen las entradas correctas;



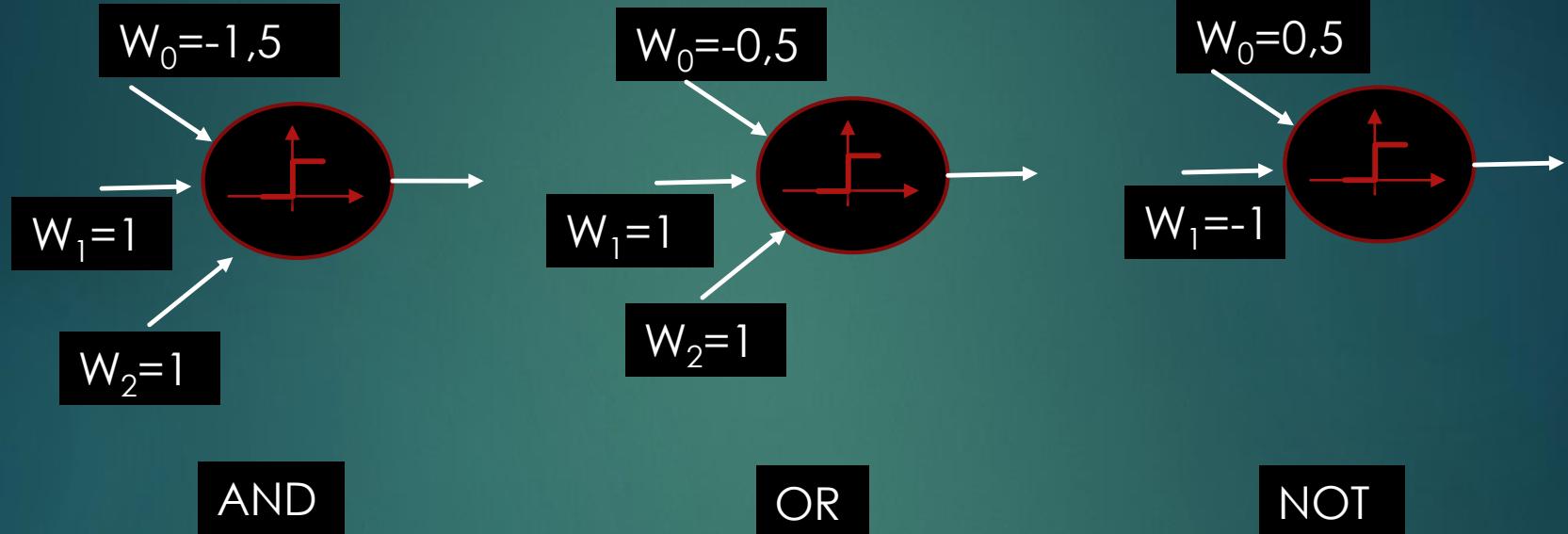
**Figura 20.16** (a) La función de activación umbral, con salida 1 cuando la entrada es positiva y 0 en otro caso. (Algunas veces se usa en lugar de esta la función **signo**, con salida +1 dependiendo de la señal de la entrada.) (b) La función **sigmoide**  $1/(1 + e^{-x})$ .

Ambas funciones tienen un umbral cero.

Los pesos de sesgo constituyen el umbral real de la unidad, en el sentido de que la unidad se activa cuando la suma de los pesos de las entradas “reales” lo excede.

La función Sigmoid tiene la ventaja de ser diferenciable.

# Puertas lógicas con Redes neuronales



AND

$$in = -1,5 + 1 \times 1 + 0 \times 1 = -0,5 \Rightarrow 0$$

$$in = -1,5 + 0 \times 1 + 1 \times 1 = -0,5 \Rightarrow 0$$

$$in = -1,5 + 0 \times 1 + 0 \times 1 = -1,5 \Rightarrow 0$$

$$in = -1,5 + 1 \times 1 + 1 \times 1 = 0,5 \Rightarrow 1$$

OR

$$in_i = \sum_{j=0}^n W_{j,i} a_j$$

$$a_i = g(in_i) = g\left(\sum_j^n W_{j,i} a_j\right)$$

# PERCEPTRÓN

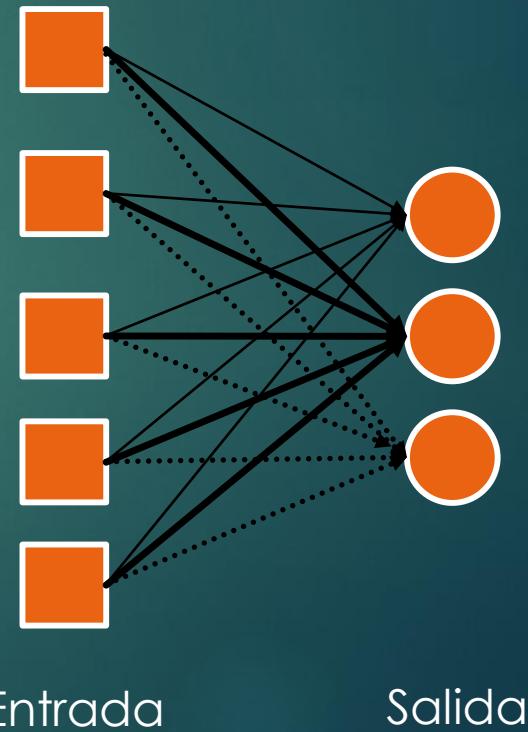
Es una RN de una sola capa con alimentación hacia adelante

Una RN con todas las entradas conectadas directamente a la salida se denomina RN de una sola capa o Red Perceptrón.

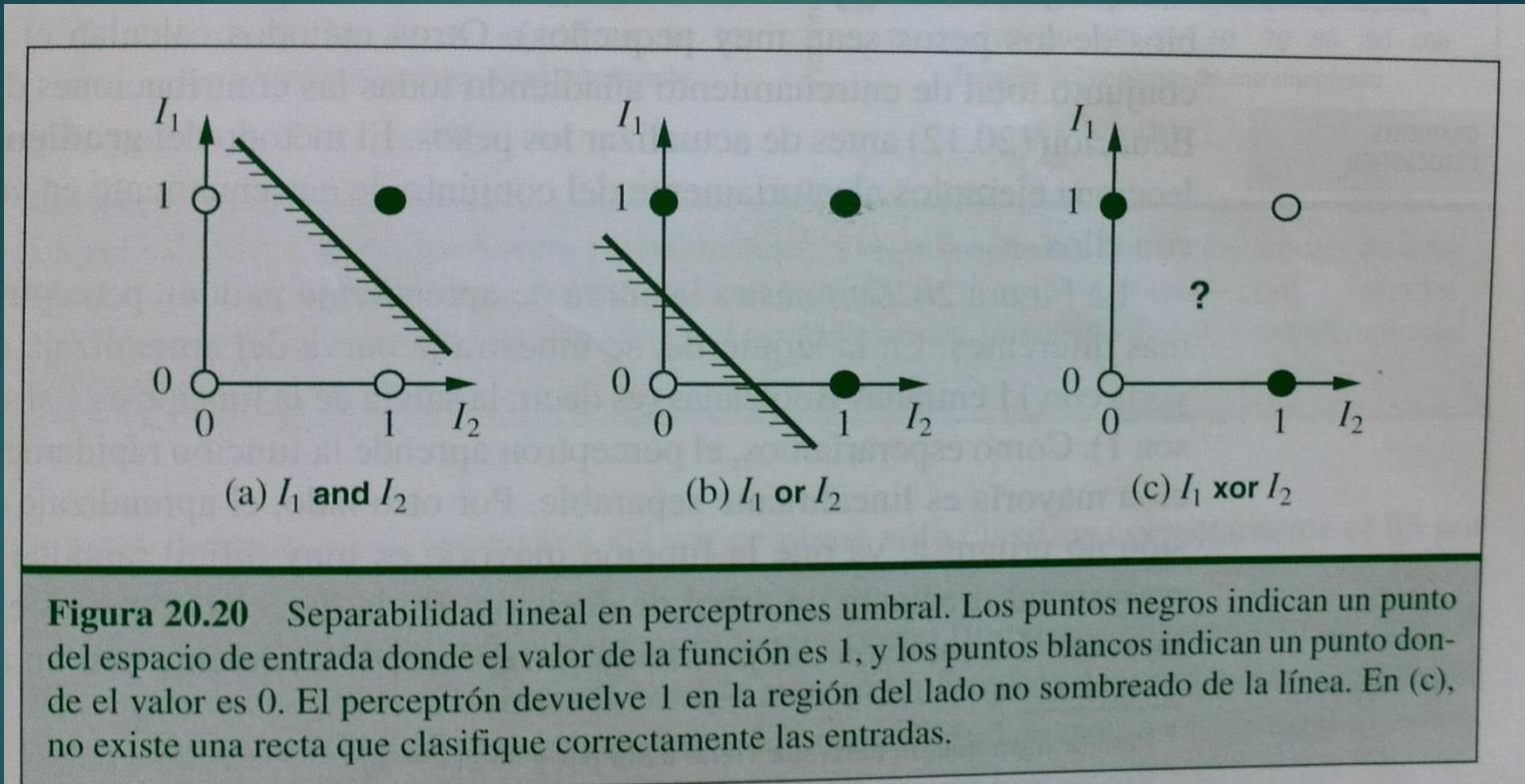
Puede representar algunas funciones Booleanas, pero no todas.

La salida es 1 si y sólo si la suma ponderada de sus entradas (incluyendo los sesgos) es positiva:  $W \cdot x > 0$  que define un hiperplano en el espacio de entrada.

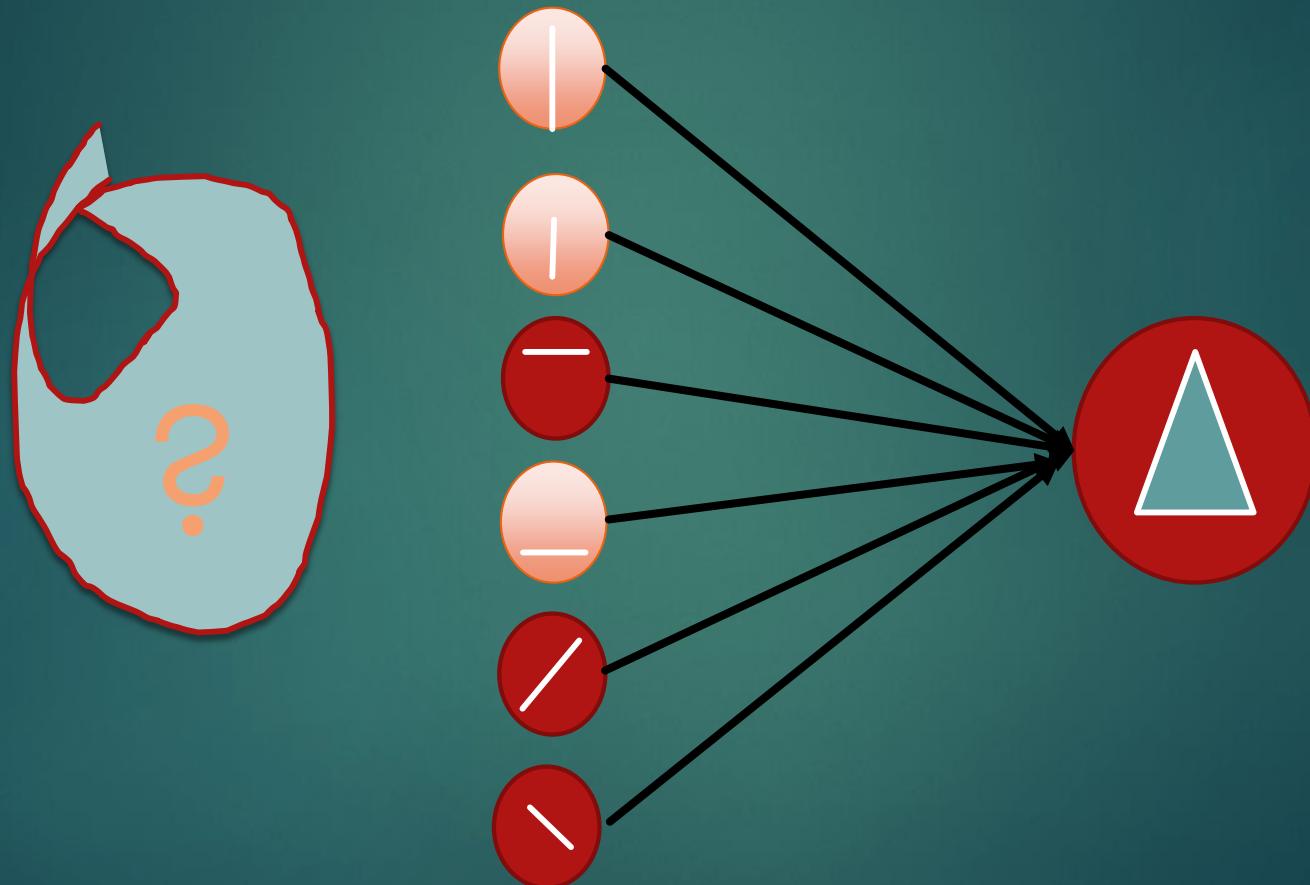
Separador lineal: devuelve 1 si y sólo si la entrada está en un lado de ese hiperplano.



Un algoritmo de aprendizaje sencillo ajusta un perceptrón umbral a cualquier conjunto de entrenamiento que sea linealmente separable.



# Perceptrón: reconocedor de figuras

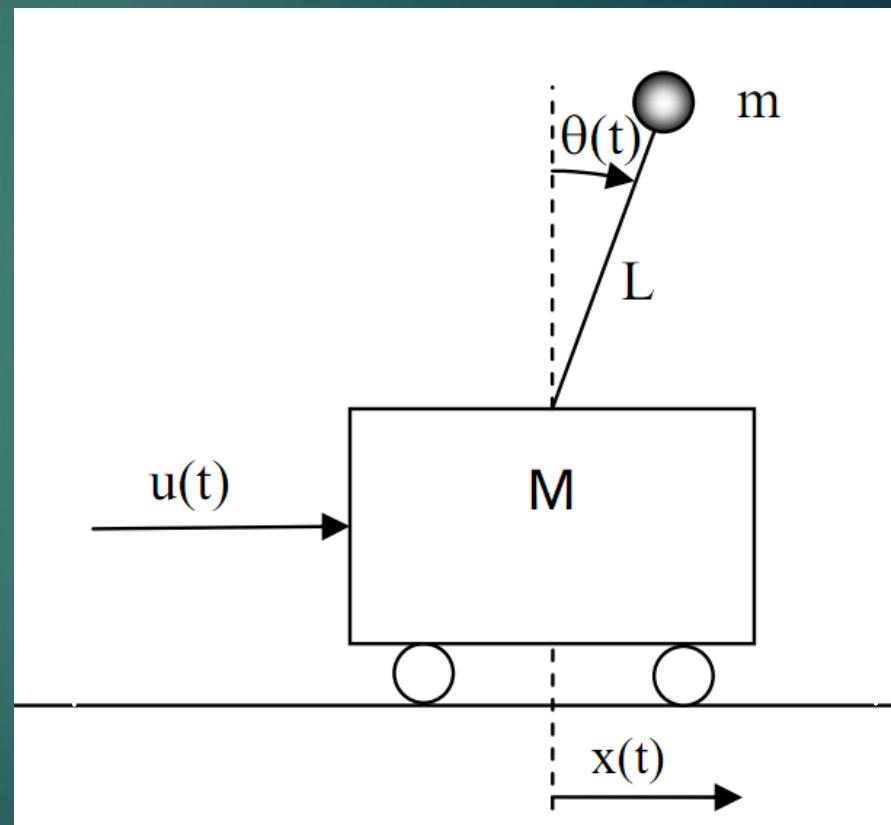


# APRENDIZAJE POR REFUERZO

Un agente puede aprender a partir del éxito y el fracaso, mediante recompensas y castigos y no a partir de ejemplos.

Recompensa o refuerzo es la realimentación con que cuenta el agente para saber que algo bueno ha ocurrido.

Problema fundamental:  
definir una política que permita maximizar el estímulo positivo (premio)

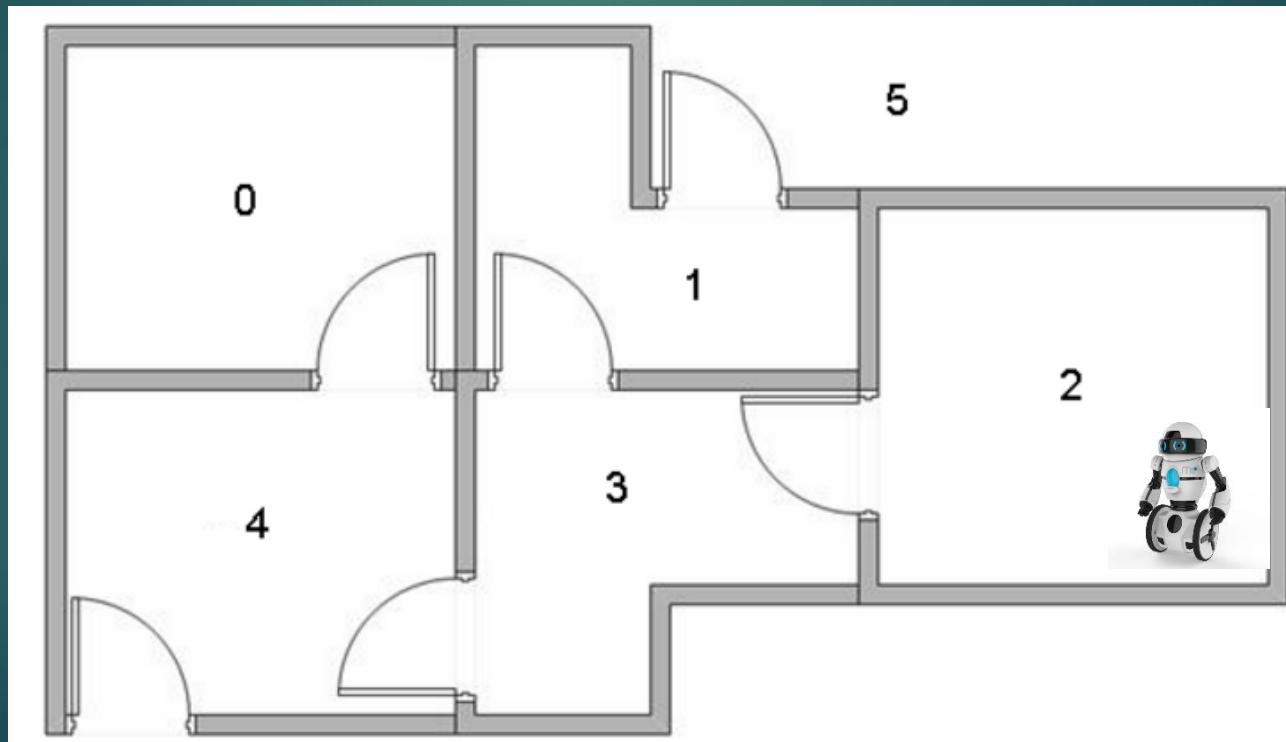


# Aprendizaje por refuerzo

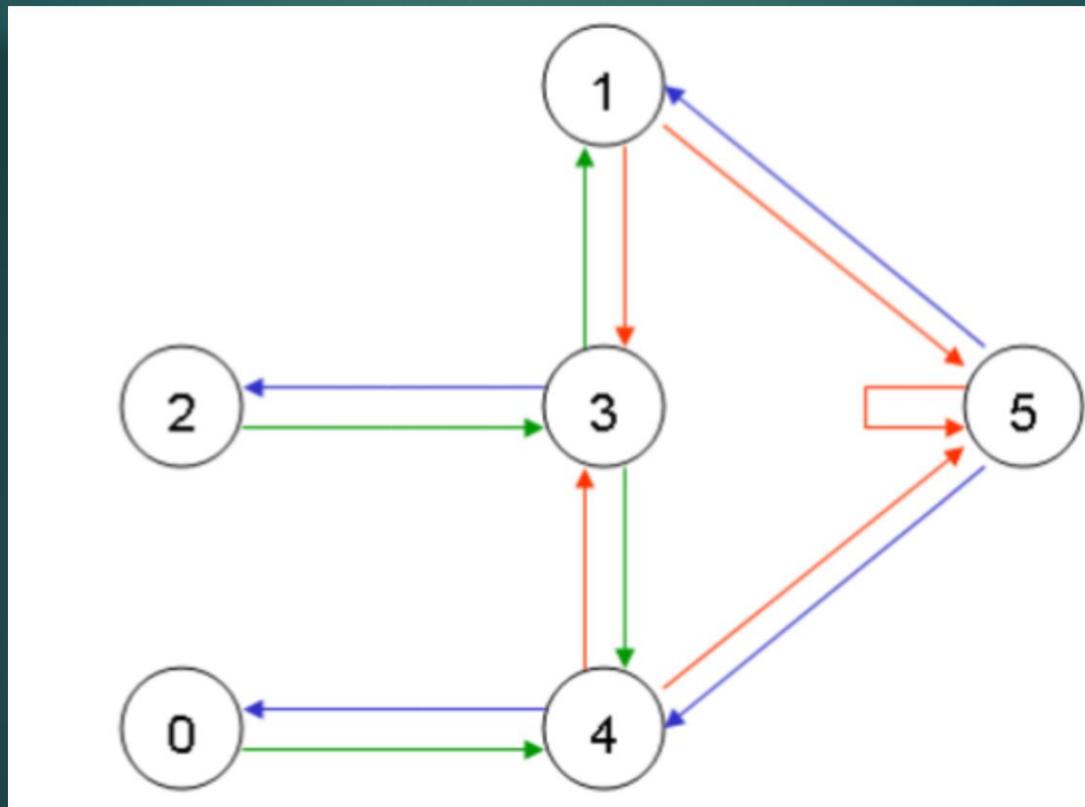
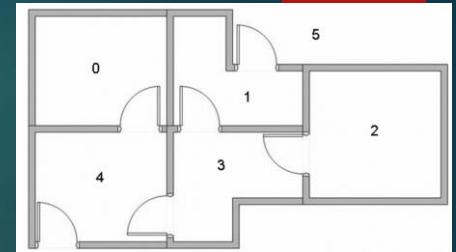
- ▶ Asumiremos un **entorno completamente observable**, por ello por cada percepción se facilita el estado actual.
- ▶ Asumiremos que el agente no conoce cómo funciona el entorno o qué acción llevar a cabo.
- ▶ Permitiremos salidas de acciones probabilísticas.

# Aprendizaje Q

Supongamos 5 habitaciones conectadas por puertas numeradas de 0 a 4. El exterior del edificio se considera una gran habitación numerada con el 5.

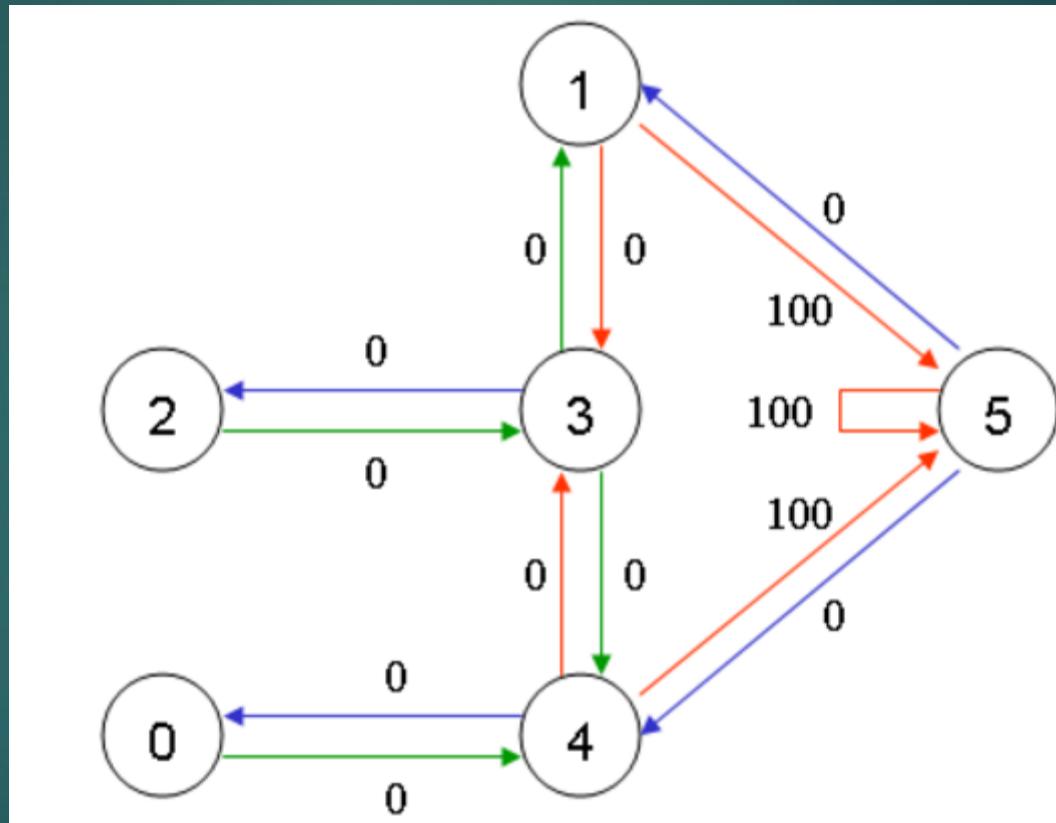


# Aprendizaje Q



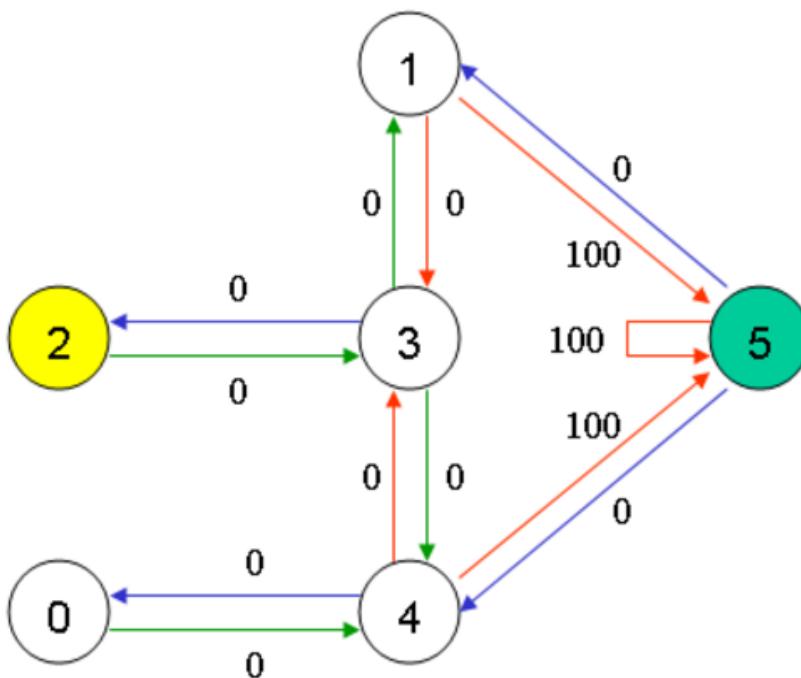
Las habitaciones son los nodos y las puertas los enlaces.

# Aprendizaje Q



Asignación de recompensas

# Aprendizaje Q



| s | a | 0  | 1  | 2  | 3  | 4  | 5   |
|---|---|----|----|----|----|----|-----|
| 0 | 0 | -1 | -1 | -1 | -1 | 0  | -1  |
| 1 | 1 | -1 | -1 | -1 | 0  | -1 | 100 |
| 2 | 2 | -1 | -1 | -1 | 0  | -1 | -1  |
| 3 | 3 | -1 | 0  | 0  | -1 | 0  | -1  |
| 4 | 4 | 0  | -1 | -1 | 0  | -1 | 100 |
| 5 | 5 | -1 | 0  | -1 | -1 | 0  | 100 |

Matriz de  
recompensas

$$R(s,a)$$

# Aprendizaje Q

- ▶ Q: es una matriz de igual dimensión que R que representa la memoria de lo que el agente va aprendiendo por medio de su experiencia.
- ▶ El agente comienza sin saber nada: Q se inicializa a cero.
- ▶ Se calcula  $Q(s,a) = R(s,a) + \gamma \max(Q(s', A))$

$s'$  : próximo estado

A: todas las acciones

$$0 <= \gamma < 1$$

# Aprendizaje Q : algoritmo

- ▶ Inicializamos  $\gamma$  y  $R$
- ▶ Inicializamos  $Q$  a cero
- ▶ Para cada episodio
  - ▶ Seleccionamos un estado inicial aleatorio
  - ▶ Mientras (no se haya alcanzado el objetivo)
    - ▶ Seleccionar las posibles acciones en  $R$
    - ▶ Seleccionar el máximo valor de  $Q$  para esas acciones en  $R$
    - ▶ Calcular  $Q(s,a)$
    - ▶ Actualizar el estado actual
  - ▶ Fin Mientras
- ▶ Fin Para

Cada episodio  
es una sesión  
de  
entrenamiento



Con este algoritmo  
el agente mejora  
su memoria

# Aprendizaje Q : ejemplo

- Inicializamos  $\gamma=0,8$  y

$$\blacktriangleright \quad R = \begin{pmatrix} -1 & -1 & -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & 0 & -1 & 100 \\ -1 & -1 & -1 & 0 & -1 & -1 \\ -1 & 0 & 0 & -1 & 0 & -1 \\ 0 & -1 & -1 & 0 & -1 & 100 \\ -1 & 0 & -1 & -1 & 0 & 100 \end{pmatrix}$$

- ## ► Inicializamos Q a cero

$$Q = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

# Aprendizaje Q : ejemplo episodio 1

- Seleccionamos un estado inicial aleatorio (fila) = 1

- $S=1$

|      |    |    |    |    |    |     |
|------|----|----|----|----|----|-----|
|      | -1 | -1 | -1 | -1 | 0  | -1  |
|      | -1 | -1 | -1 | 0  | -1 | 100 |
| $R=$ | -1 | -1 | -1 | 0  | -1 | -1  |
|      | -1 | 0  | 0  | -1 | 0  | -1  |
|      | 0  | -1 | -1 | 0  | -1 | 100 |
|      | -1 | 0  | -1 | -1 | 0  | 100 |

Hay dos acciones posibles

- Seleccionamos en forma aleatoria ir a 5 en Q

|      |   |   |   |   |   |   |
|------|---|---|---|---|---|---|
|      | 0 | 0 | 0 | 0 | 0 | 0 |
|      | 0 | 0 | 0 | 0 | 0 | 0 |
| $Q=$ | 0 | 0 | 0 | 0 | 0 | 0 |
|      | 0 | 0 | 0 | 0 | 0 | 0 |
|      | 0 | 0 | 0 | 0 | 0 | 0 |

- Calcular:  $Q(1,5)=R(1,5) + \gamma \max[Q(5,1), Q(5,4), Q(5,5)]$

# Aprendizaje Q : ejemplo episodio 1

- Calcular:  $Q(1,5) = R(1,5) + \gamma \max[Q(5,1), Q(5,4), Q(5,5)]$
- $Q(1,5) = 100 + 0,8 \max(0, 0, 0) = 100$

# Aprendizaje Q : ejemplo episodio 1

- $Q(1,5) = 100 + 0,8 \max(0, 0, 0) = 100$
- ▶ Dado que la matriz  $Q=0$ ,  $Q(5,1)=Q(5,4)=Q(5,5)=0$
- ▶ El estado actual se actualiza a 5
- ▶  $Q$  ha sido actualizada a

$$Q = \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

# Aprendizaje Q : ejemplo episodio 2

- Elegimos un nuevo estado inicial aleatorio = 3

$$R = \begin{bmatrix} -1 & -1 & -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & 0 & -1 & 100 \\ -1 & -1 & -1 & 0 & -1 & -1 \\ -1 & 0 & 0 & -1 & 0 & -1 \\ 0 & -1 & -1 & 0 & -1 & 100 \\ -1 & 0 & -1 & -1 & 0 & 100 \end{bmatrix}$$

Seleccionamos  
ir a la 1

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$
$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 80 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

- $Q(3,1) = 0 + 0,8 \max(0, 100) = 80$
- Actualizo

# Aprendizaje Q convergencia

LOS EPISODIOS SE REPITEN HASTA QUE Q ALCANZA LA CONVERGENCIA.

LA MATRIZ Q SE SUELE NORMALIZAR DIVIDIENDO CADA ELEMENTO DISTINTO DE CERO POR EL VALOR DEL ELEMENTO MAYOR.

$$Q = \begin{matrix} 0 & 0 & 0 & 0 & 400 & 0 \\ 0 & 0 & 0 & 320 & 0 & 500 \\ 0 & 0 & 0 & 320 & 0 & 0 \\ 0 & 400 & 256 & 0 & 400 & 0 \\ 320 & 0 & 0 & 320 & 0 & 500 \\ 0 & 400 & 0 & 0 & 400 & 500 \end{matrix}$$

$$Q = \begin{matrix} 0 & 0 & 0 & 0 & 0,8 & 0 \\ 0 & 0 & 0 & 0,6 & 0 & 1 \\ 0 & 0 & 0 & 0,6 & 0 & 0 \\ 0 & 0,8 & 0,5 & 0 & 0,8 & 0 \\ 0,6 & 0 & 0 & 0,6 & 0 & 1 \\ 0 & 0,8 & 0 & 0 & 0,8 & 1 \end{matrix}$$

# Aprendizaje Q política óptima

Se obtiene la mejor secuencia de estados siguiendo los enlaces de mayor valor.

$$Q = \begin{matrix} & 0 & 0 & 0 & 0 & 0,8 & 0 \\ 0 & & 0 & 0 & 0,6 & 0 & 1 \\ 0 & 0 & 0 & 0,6 & 0 & 0 & 0 \\ 0 & 0,8 & 0,5 & 0 & 0,8 & 0 & 0 \\ 0,6 & 0 & 0 & 0,6 & 0 & 0 & 1 \\ 0 & 0,8 & 0 & 0 & 0,8 & 1 & \end{matrix}$$

