

# Tema 4: Aprendizaje automático

- ❑ Aspectos fundamentales de la Inteligencia Artificial
  - ❑ Representación del conocimiento
  - ❑ Métodos de búsqueda
  - ❑ Aprendizaje    ←    ←    ←
- ❑ **No se puede hablar de inteligencia auténtica sin aprendizaje**  
... aunque muchos sistemas no tienen aprendizaje

## □ Tema 4: Aprendizaje automático

- Introducción
- Tipos de aprendizaje: Clasificaciones
  - Según el grado de realimentación
  - Según el paradigma utilizado
  - Según lo que se aprende  
(o según el problema que se resuelve)

# Aprendizaje Automático (*machine learning*)

- Aprendizaje
  - “El acto, proceso o experiencia de adquirir conocimiento o aptitudes”
  - ... y poder hacer algún tipo de inferencia o acción con ello
- El aprendizaje es la clave de la inteligencia
- El aprendizaje está relacionado con el conocimiento
  - “El proceso mediante el cual un ente adquiere conocimiento”
  - Este conocimiento puede ser suministrado
    - Por otro ente denominado “profesor”
    - Adquirirse por el propio ente “automáticamente”
- El aprendizaje automático es un área muy activa porque...
  - Hay gran cantidad de información en formato electrónico
  - La mejora constante de las técnicas y los algoritmos
  - La potencia de procesamiento cada vez es más barata

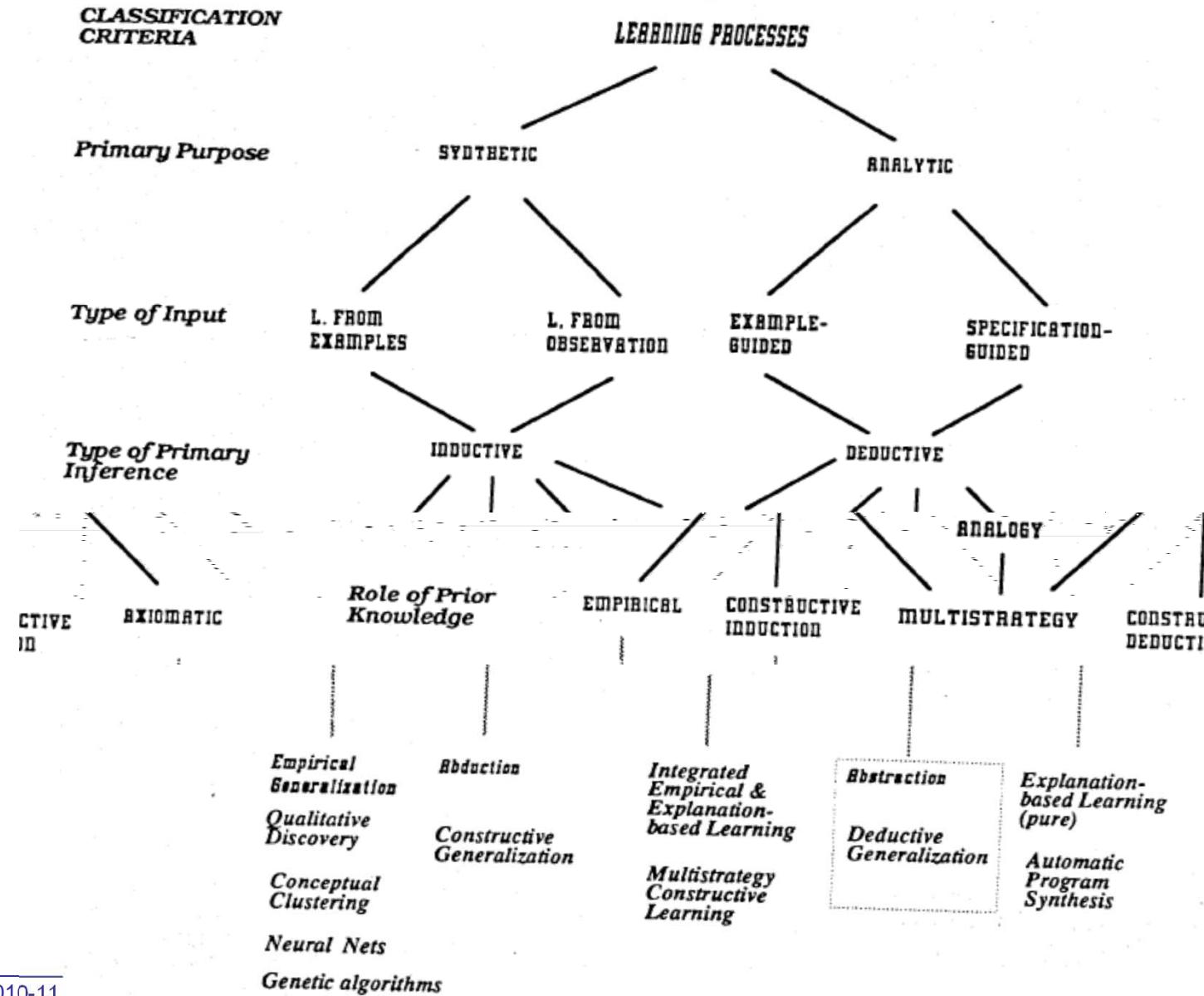
# Aprendizaje de... conocimiento

- Sistema Inteligente
  - ¿qué conocimiento? ¿cómo adquirirlo?
- Ingeniería de conocimiento – expertos humanos
  - El cuello de botella de la adquisición de conocimiento
  - Mucho tiempo y mala gestión de errores
  - ¡¡¡ A veces ni siquiera está disponible !!!
  - Ejemplo: todos somos expertos en reconocimiento de imágenes
    - Podemos ver una foto y reconocer gente envejecida, con poca luz, bronceada, girada,....
    - ¿Podemos escribir un programa que lo haga? ¿Y ayudar a explicitar el conocimiento que utilizamos para realizar este proceso?
- A menudo, las personas aprenden de ejemplos
  - → Muchos sistemas imitan este modo de aprendizaje

# ¿Qué entendemos por aprendizaje?

- Cambios en el sistema que son adaptativos:
  - Ejecutar la misma tarea de un modo más eficiente y eficaz
  - Resolver problemas nuevos
- Mejorar una tarea con la **experiencia** mediante
  - Refinamiento de habilidades
  - Adquisición de conocimiento
- El aprendizaje no puede añadirse a posteriori a un programa
- Aprender significa varias cosas. Entre otras:
  - Memorizar
  - Observar y explorar
  - Practicar para desarrollar las capacidades
  - Organizar el nuevo conocimiento de una forma general y efectiva

# Clasificaciones de Aprendizaje



# Clasificaciones de Aprendizaje

- ❑ Segundo el grado de realimentación
  - ❑ Cómo se observa, explora y memoriza
  - ❑ Supervisado, no supervisado (por descubrimiento) y por refuerzo
- ❑ Segundo el paradigma utilizado
  - ❑ Aprendizaje inductivo: aprende descripción (con muchos ejemplos “+” y “-”)
  - ❑ Aprendizaje analítico/deductivo: mejor resolver problemas (1 ejemplo+teoría)
  - ❑ Algoritmos genéticos: crea generación de soluciones y evalúa su calidad
  - ❑ Método conexionista-red neuronal: reconocimiento de patrones (letras, voz)
- ❑ Segundo lo que se aprende (usan los paradigmas anteriores)
  - ❑ Aprendizaje de resolución de problemas
    - ❑ Memorístico o rutinario ; Por ajuste de parámetros ;
    - ❑ Macro-operadores ; Macro-reglas y meta-reglas ;
    - ❑ Por analogía ; Razonamiento basado en casos (CBR)
  - ❑ Aprendizaje de conceptos
    - ❑ Inductivo (Winston,Mitchell, Quinlian ID3). Analítico (explicaciones EBL)

# Tipos de aprendizaje: según el grado de realimentación

## Aprendizaje supervisado

- Los ejemplos de entrenamiento son comprobables... Ya sea porque
  - “el profesor” le suministre el valor real al sistema
  - o porque pueda percibirlo (en entornos observables)
- Redes neuronales supervisadas y los demás que veremos (no clustering)

## Aprendizaje no supervisado o por descubrimiento

- No hay “profesor” para verificar los ejemplos
- Objetivo: descubrir patrones en el conjunto de entrenamiento que...
- Permitan agrupar y diferenciar unas observaciones (ejemplos) de otras:
  - Formación de taxonomías y clustering conceptual
  - Descubrimiento de leyes cualitativas (AM) y de leyes cuantitativas (BACON)

## Aprendizaje por refuerzo

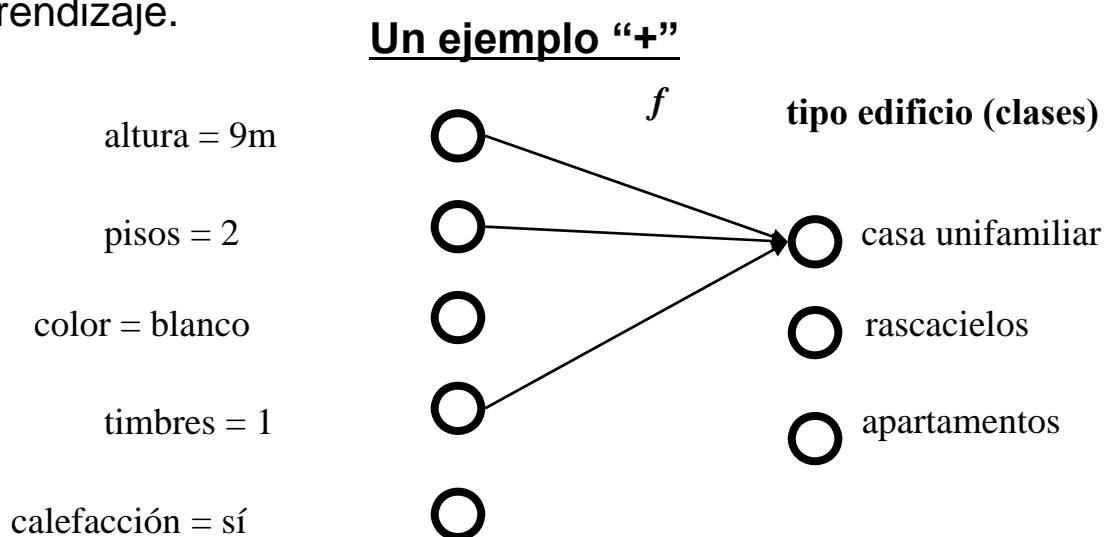
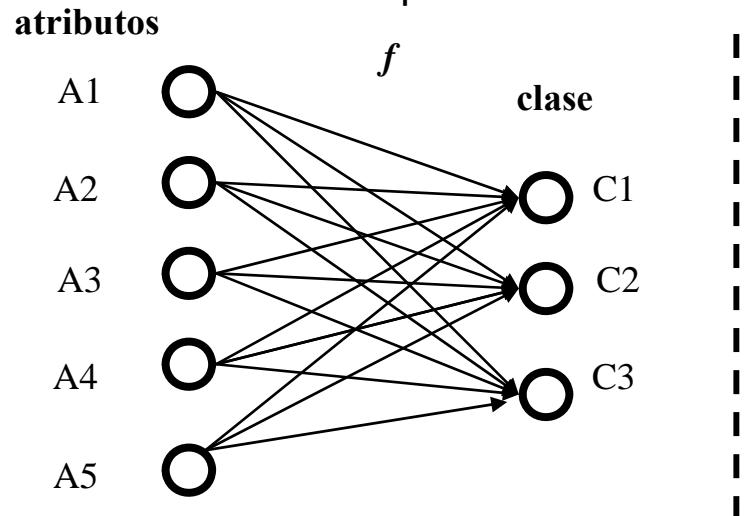
- El sistema recibe algún tipo de recompensa (positiva o negativa)
- Cada vez que produce una respuesta,
- Ajustando su comportamiento en función de dicha recompensa

## ---Aprendizaje inductivo ---

- Clasificaciones del aprendizaje:
  - Grado de realimentación: supervisado
  - Paradigma: inductivo
  - Qué aprende: descripción de Conceptos (y también reglas)

# Aprendizaje inductivo: Definiciones previas

- Estrategia de aprendizaje: algoritmo que genera la descripción concepto/clase
- Ejemplos de entrenamiento
  - Describen situaciones reales mediante un conjunto de atributos
  - Se usan para entrenar el aprendizaje.



- Ejemplos positivos “+” (ejemplos que sí son de ese concepto o clase) y negativos “-“ (que no son del concepto, contraejemplos)
- Clasificar los ejemplos: asignarle etiqueta de + o –
- Ruido: si se ha dado una etiqueta + a un ejemplo negativo o viceversa
- Hipótesis: conjeturas posibles sin probar (solución: la descripción adecuada).

## Aprendizaje inductivo: *aprendizaje a partir de ejemplos*

- ❑ Objetivo: construir la descripción de un concepto (conjunto condiciones)
  - ❑ Así establecer rasgos comunes de una serie de ejemplos
  - ❑ En la que encajen todos los ejemplos “+” y ningún ejemplo “-”
  - ❑ Que permita decidir si un ejemplo sin etiquetar es “+” ó “-”
  - ❑ Y así diferenciarlo de otro concepto : Clasifica en Categorías
- ❑ Estrategia: Proceso de búsqueda en el espacio de hipótesis
  - ❑ Hipótesis: cada posible descripción del concepto
- ❑ Un ejercicio: Dada una lista de ejemplos (previamente clasificadas)
  - ❑ pares ordenados de números:  $\{(3,4), E+\}, \{(4,5), E-\}, \{(9,5), E-\}, \{(1,6), E+\}\}$
- ❑ Una solución (concepto obtenido):
  - ❑ “par ordenado de números en el que el primero es menor que el segundo”  
(una hipótesis de todas las posibles)

# Aprendizaje inductivo : Dificultades

- Disponemos de un número suficiente de ejemplos?
- Problemas de ruido (ejemplos mal clasificados)
  - Un coche de juguete (no es vehículo)
- Ejemplos malos/sesgados/poco representativos
  - Una carroza / nave espacial / platillo volante
- Información irrelevante → dificulta la inducción
  - El color , hélices ?
- Definir un vehículo:

Etiqueta	ruedas	motor	volante	manillar	hélices
Bici	2	-	-	si	-
Coche	4	1	si	-	-
Barco	-	1	-	-	si (timón ?)
Moto	2	1	-	si	-
Carrito de la compra	2	-	-	-	- (no vehículo)

# --- Sistema de aprendizaje de Winston ---

- Es inductivo supervisado
- Qué aprende?
  - Construir definiciones de conceptos del dominio “mundo de bloques”
- Proceso:
  - Recibe **incrementalmente ejemplos** y contraejemplos de un concepto/clase
  - Genera incrementalmente una **descripción** estructural de dicho **concepto**
  - Esta descripción sirve para clasificar correctamente ejemplos desconocidos
- Se emplea en tareas de reconocimiento:
  - de caracteres, de piezas defectuosas, etc.
- Es **sensible al orden** de los ejemplos
- Realiza un recorrido primero en profundidad
  - en el espacio de posibles descripciones de conceptos

# Aprendizaje de Winston: Análisis situación

- ❑ Se empieza por analizar la situación, determinando
  - ❑ El problema: crear una descripción textual (regla a cumplir)  
Distinguir lo que es un “arco” de otra cosa
  - ❑ Elementos relevantes: objetos, relaciones o propiedades  
Objetos: bloques B, C, D  
Relaciones: vertical, horizontal, separados, es\_un, tiene\_parte
  - ❑ Determinar la representación formal de la descripción textual  
apoyado\_en(a1 a2) no\_unión(a2 a3)  
Si hecho1(a1 a4) & ... hecho8(a2) Entonces es\_un(a1, arco)
  - ❑ Construir ejemplos “+” y “-” (50% cada) en el lenguaje formal
    - ❑ con la ayuda de expertos del dominio (el “profesor”)
- ❑ Algoritmo: Encontrar una descripción que sea un patrón (regla)
  - ❑ Donde encajen los ejemplos adecuados (+) y no los contraejemplos (-)
  - ❑ Mediante un proceso de refinamiento (generaliza o especifica la regla)

# Aprendizaje de Winston: Algoritmo

Pasos:

- Crear una descripción estructural (regla) usando el primer ejemplo +
- Generalizar la regla para que incluya todos los ejemplos +
- Especificar la regla para que excluya todos los ejemplos -

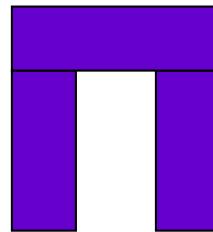
Operaciones sobre la regla a partir de cada ejemplo

- Variabilización: Añadir variables (generalizar)
- Jerarquizar las clases o conceptos (generalizar)
- Debilitar añadiendo disyunciones de condiciones (generalizar)
- Fortalecer añadiendo conjunciones de condiciones (especificar)
- Cuantificar para todos los ejemplos (generalizar). Así se termina.

# Aprendizaje de Winston : Ejemplo en texto

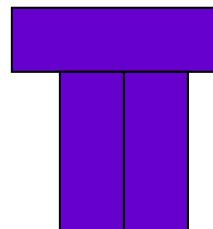
## Ejemplos recibidos

EJ 1



(+)

EJ 2



(-)

## Descripción estructural generada

**(regla a cumplir)**

**Dos bloques verticales  
y un bloque horizontal**

Se empieza analizando una instancia conocida del concepto que quiere describirse (*arco*, en este caso)

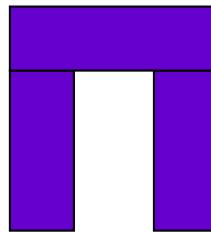
**Dos bloques verticales *separados*  
y un bloque horizontal**

Cuando un ejemplo (–) encaja es porque la descripción es demasiado general → **especializar**

# Aprendizaje de Winston : Ejemplo en texto

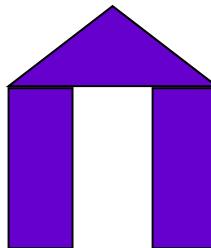
## Ejemplos recibidos

EJ 3



(+)

EJ 4



(+)

## Descripción estructural generada

Dos bloques verticales separados  
y un bloque horizontal

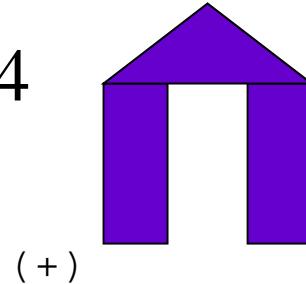
Dos bloques verticales separados  
y un *objeto* horizontal

Cuando un ejemplo (+) no encaja es porque la descripción es demasiado específica → **generalizar**

# Aprendizaje de Winston : Ejemplo en texto

## Ejemplos recibidos

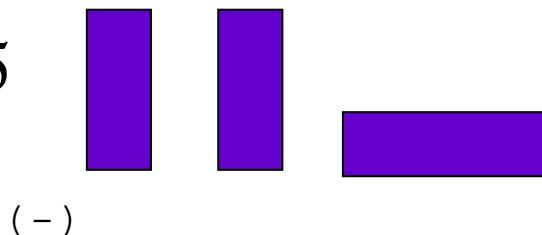
EJ 4



## Descripción estructural generada

Dos bloques verticales separados  
y un objeto horizontal  
(repetido para la explicación)

EJ 5

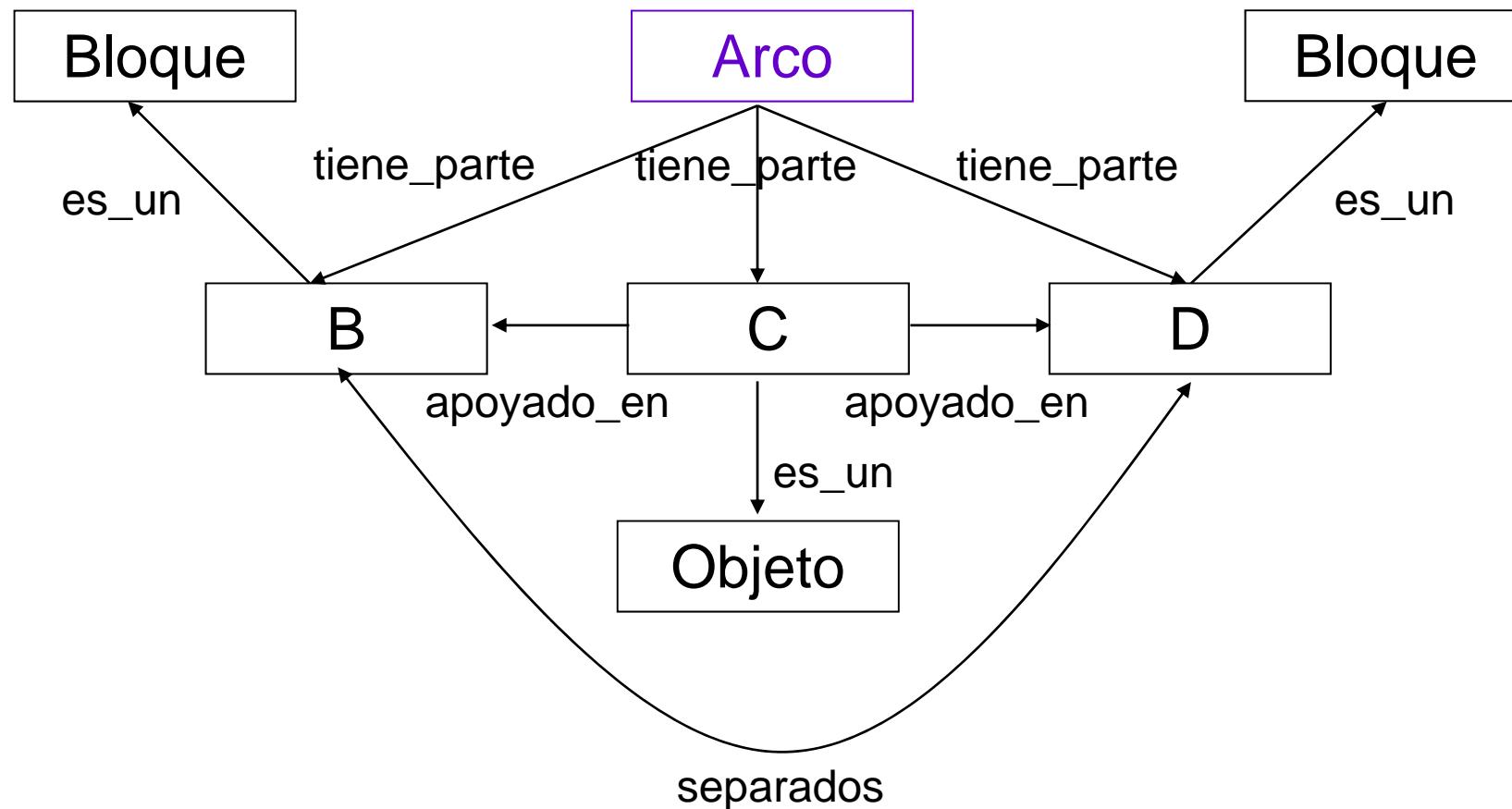


Dos bloques verticales separados  
y un objeto horizontal  
*apoyado en ellos*

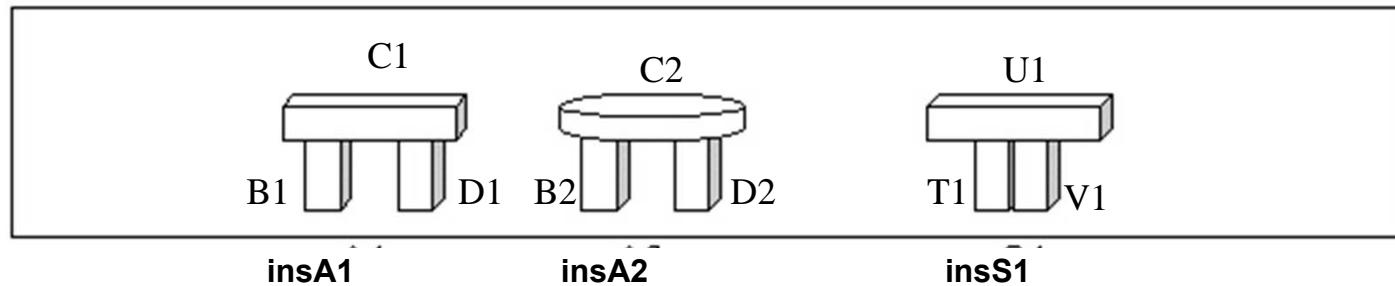
Cuando un ejemplo (-) encaja es porque la descripción  
es demasiado genérica → fortalecer (especificar)

# Aprendizaje de Winston : Ejemplo con redes

- Las descripciones se representan con redes semánticas
  - Facilitan el proceso de generalización y particularización



# Aprendizaje de Winston : Ejemplo formalizado



- Paso 1: análisis de la instancia insA1 (ejemplo positivo de arco)

```
tiene_parte(insA1, B1) & tiene_parte(insA1, C1) &  
tiene_parte(insA1, D1) & apoyado_en(C1, B1) &  
apoyado_en(C1, D1) & es_un(B1, ladrillo) & es_un(C1,  
ladrillo) & es_un(D1, ladrillo) & no_union(B1, D1)
```

- Paso 2: análisis de la instancia insA2 (ejemplo positivo de arco)

```
tiene_parte(insA2, B2) & tiene_parte(insA2, C2) &  
tiene_parte(insA2, D2) & apoyado_en(C2, B2) &  
apoyado_en(C2, D2) & es_un(B2, ladrillo) & es_un(C2,  
cilindro) & es_un(D2, ladrillo) & no_union(B2, D2)
```

- Luego, la descripción se **generaliza** para incluir a ambas instancias

```
tiene_parte(A, B) & tiene_parte(A, C) & tiene_parte(A, D) &  
apoyado_en(C, B) & apoyado_en(C, D) & es_un(B, ladrillo) &  
es_un(C, objeto) & es_un(D, ladrillo) & no_union(B, D)
```

donde tanto ladrillo (bloque) como cilindro son instancias de una clase superior objeto

# Aprendizaje de Winston : Ejemplo formalizado

- Paso 3: se considera la instancia insS1 (semejante, ejemplo negativo)

```
tiene_parte(insS1, T1) & tiene_parte(insS1, U1) &  
tiene_parte(insS1, V1) & apoyado_en(U1, T1) & apoyado_en(U1,  
V1) & es_un(T1, ladrillo) & es_un(U1, ladrillo) & es_un(V1,  
ladrillo)
```

- La definición del concepto debe **restringirse** para excluir el semejante
  - La definición debe excluir de manera explícita al semejante fortaleciendo la relación con “no\_debe\_union”
- La descripción queda de la siguiente manera:

```
tiene_parte(A, B) & tiene_parte(A, C) & tiene_parte(A, D) &  
apoyado_en(C, B) & apoyado_en(C,D) & es_un(B, ladrillo) &  
es_un(C, objeto) & es_un(D, ladrillo) & no_debe_union(B, D)
```

## ---- Espacio de versiones de Mitchell -----

- Es inductivo y supervisado
- Qué aprende?
  - Una descripción de un concepto
  - Que sea consistente con los ejemplos positivos y negativos
- No le afecta el orden de los ejemplos
- Se mantiene un conjunto de descripciones posibles hasta el final
  - Existen descripciones más generales que otras (orden parcial)
  - La jerarquía completa forma el espacio de descripciones de conceptos
- Dos límites de abstracción de descripciones que se acercan
  - G: Las más generales consistentes con los ejemplos (límite superior)
  - E: La más específica consistente con los ejemplos (límite inferior)
- El espacio de versiones son descripciones entre G y E
- En E solo hay una descripción

# Espacio de versiones de Mitchell

- ❑ Sesgo del sistema (elementos necesarios para que pueda aprender):
  - ❑ Conjunción de literales positivos
- ❑ Desarrolla una búsqueda en anchura
  - ❑ Sobre el conjunto de descripciones posibles
  - ❑ En el espacio de versiones (franja contenida entre G y E)
- ❑ Tiene baja tolerancia al ruido:
  - ❑ un ejemplo mal etiquetado puede hacer que no converja
- ❑ Puede no obtener solución (si no converge G y E)

# Espacio de versiones de Mitchell: algoritmo

$G :=$  hipótesis vacía (*variables*)

$E :=$  primer ejemplo positivo (puede haber antes ejem. negativos)

**mientras** quedan ejemplos **y** no parar

- Coger siguiente ejemplo SE
- Si SE es positivo => eliminar de  $G$  cualquier descripción que no sea consistente con SE.  
*Generalizar E lo imprescindible para que sea consistente con SE*
- Si SE es negativo => eliminar de  $E$  descripciones consistentes con SE. *Especializar G lo imprescindible para que no sea consistente con SE*
- Si  $E$  y  $G$  son unitarios e iguales => imprimir contenido y parar
- Si  $E$  y  $G$  son unitarios e incompatibles => error de inconsistencia y parar

# Espacio de versiones de Mitchell: ejemplo 1

- Objetivo:

- Aprender el “perfil característico” de pacientes con enfermedad X

- Cada paciente se describe con 5 características:

- Edad (<20, 20-50, >50)
  - Lugar de residencia (ciudad, campo, montaña, costa)
  - Actividad física (baja, media, alta)
  - Consumo de tabaco (nulo, bajo, medio, alto)
  - Tipo de trabajo (sedentario, físico, paro)

- Ejemplos de entrenamiento

1. (20-50, ciudad, baja, nulo, sedentario) +
2. (>50, campo, baja, medio, físico) -
3. (>50, ciudad, baja, alto, sedentario) +
4. (20-50, ciudad, media, bajo, sedentario) -
5. (20-50, ciudad, baja, medio, paro) +

# Espacio de versiones de Mitchell: ejemplo 1

## Proceso de aprendizaje *(Xi son variables)*

### Ejemplo 1: (20-50, ciudad, baja, nulo, sedentario) +

$G = \{(X_1, X_2, X_3, X_4, X_5)\}$  *hipótesis vacía*

$E = \{(20-50, \text{ciudad}, \text{baja}, \text{nulo}, \text{sedentario})\}$  *1º ej. positivo*

### Ejemplo 2: (>50, campo, baja, medio, físico) -

$E = \{(20-50, \text{ciudad}, \text{baja}, \text{nulo}, \text{sedentario})\}$  *nada a eliminar*

$G = \{(20-50, X_2, X_3, X_4, X_5), (X_1, \text{ciudad}, X_3, X_4, X_5), (X_1, X_2, X_3, \text{nulo}, X_5), (X_1, X_2, X_3, X_4, \text{sedentario})\}$

*especializar lo imprescindible para que no sea consistente G con EJ2*

### Ejemplo 3: (>50, ciudad, baja, alto, sedentario) +

$G = \{(X_1, \text{ciudad}, X_3, X_4, X_5), (X_1, X_2, X_3, X_4, \text{sedentario})\}$

*eliminar lo no consistente*

$E = \{(X_1, \text{ciudad}, \text{baja}, X_4, \text{sedentario})\}$  *generalizar*

# Espacio de versiones de Mitchell: ejemplo 1

## Proceso de aprendizaje (continuación)

- Ejemplo 3: ( $>50$ , ciudad, baja, alto, sedentario) + (repetimos)
  - $G = \{(X_1, \text{ciudad}, X_3, X_4, X_5), (X_1, X_2, X_3, X_4, \text{sedentario})\}$  por claridad)
  - $E = \{(X_1, \text{ciudad}, \text{baja}, X_4, \text{sedentario})\}$

## Ejemplo 4: (20-50, ciudad, media, bajo, sedentario) -

- $E = \{(X_1, \text{ciudad}, \text{baja}, X_4, \text{sedentario})\}$  *nada a eliminar*
- $G = \{(X_1, \text{ciudad}, \text{baja}, X_4, X_5), (X_1, X_2, \text{baja}, X_4, \text{sedentario})\}$   
*especializar lo imprescindible para que no sea consistente*

## Ejemplo 5: (20-50, ciudad, baja, medio, paro) +

- $G = \{(X_1, \text{ciudad}, \text{baja}, X_4, X_5)\}$   
*eliminar lo no consistente*
  - $E = \{(X_1, \text{ciudad}, \text{baja}, X_4, X_5)\}$  *generalizar*
- FIN : G = E* luego es correcta esa descripción aprendida

# Espacio de versiones de Mitchell: ejemplo 2

- Representación del concepto coche como marco:

```
(origen = x1;  
marca = x2;  
color = x3;  
década = x4;  
tipo = x5)
```

$x1 \in \{\text{Japón, EEUU, UK, Italia, ...}\}$   
 $x2 \in \{\text{Honda, Toyota, Chrysler, Fiat, ...}\}$   
 $x3 \in \{\text{azul, blanco, amarillo, verde, ...}\}$   
 $x4 \in \{\text{1950, 1960, 1970, 1980, 1990, 2000, ...}\}$   
 $x5 \in \{\text{económico, lujo, deportivo, ...}\}$

- Un ejemplo o instancia del concepto *coche* en particular:

```
(origen = Japón; marca = Honda; color = azul;  
década = 1970; tipo = económico)
```

- Descripción de conceptos en términos de ranuras y valores.

- Por ejemplo, el concepto *coche económico japonés*

```
(origen = Japón; marca = x2; color = x3;  
década = x4; tipo = económico)
```

- Objetivo: como deducir el concepto a partir de ejemplos?

## Espacio de versiones de Mitchell: ejemplo 2

- Objetivo: inducir el concepto *coche económico japonés*
  - Se inicializa  $G = \{(x_1, x_2, x_3, x_4, x_5)\}$  *hipótesis vacía*
  - Ejemplo 1 (positivo)
    - (origen = Japón; marca = Honda; color = azul; década = 1970; tipo = económico)
    - $E = \{(Japón, Honda, azul, 1970, económico)\}$  *1º ej. positivo*
  - Ejemplo 2 (negativo)
    - (origen = Japón; marca = Toyota; color = verde; década = 1970; tipo = deportivo)
    - $E = \{\}$  igual (porque excluye Ejemplo 2)
    - $G = \{(x_1, Honda, x_3, x_4, x_5), (x_1, x_2, azul, x_4, x_5), (x_1, x_2, x_3, x_4, económico)\}$  *especializar para no consistencia*
  - Ejemplo 3 (positivo)
    - (origen = Japón; marca = Toyota; color = azul; década = 1990; tipo = económico)
    - $G = \{(x_1, x_2, azul, x_4, x_5), (x_1, x_2, x_3, x_4, económico)\}$  *eliminar*
    - $E = \{(Japón, x_2, azul, x_4, económico)\}$  *generalizar*

# Espacio de versiones de Mitchell: ejemplo 2

## Ejemplo 3 (positivo) → repetido para explicación

- (origen = Japón; marca = Toyota; color = azul; década = 1990; tipo = económico)
- $G = \{(x_1, x_2, \text{azul}, x_4, x_5), (x_1, x_2, x_3, x_4, \text{económico})\}$
- $E = \{\text{(Japón, } x_2, \text{azul, } x_4, \text{económico)}\}$

## Ejemplo 4 (negativo)

- (origen = EEUU; marca = Chrysler; color = azul; década = 1980; tipo = económico)
- $E$  igual (porque excluye a ejemplo 4)
- $G = \{\text{(Japón, } x_2, \text{azul, } x_4, x_5), \text{(Japón, } x_2, x_3, x_4, \text{económico)}\}$

## Ejemplo 5 (positivo)

- (origen = Japón; marca = Honda; color = blanco; década = 1980; tipo = económico)
- $G = \{\text{(Japón, } x_2, x_3, x_4, \text{económico)}\}$  *eliminar*
- $E = \{\text{(Japón, } x_2, x_3, x_4, \text{económico)}\}$  *generalizar*

## $G = E$ convergen: es la descripción del concepto buscada

# Espacio de versiones : El problema del ruido

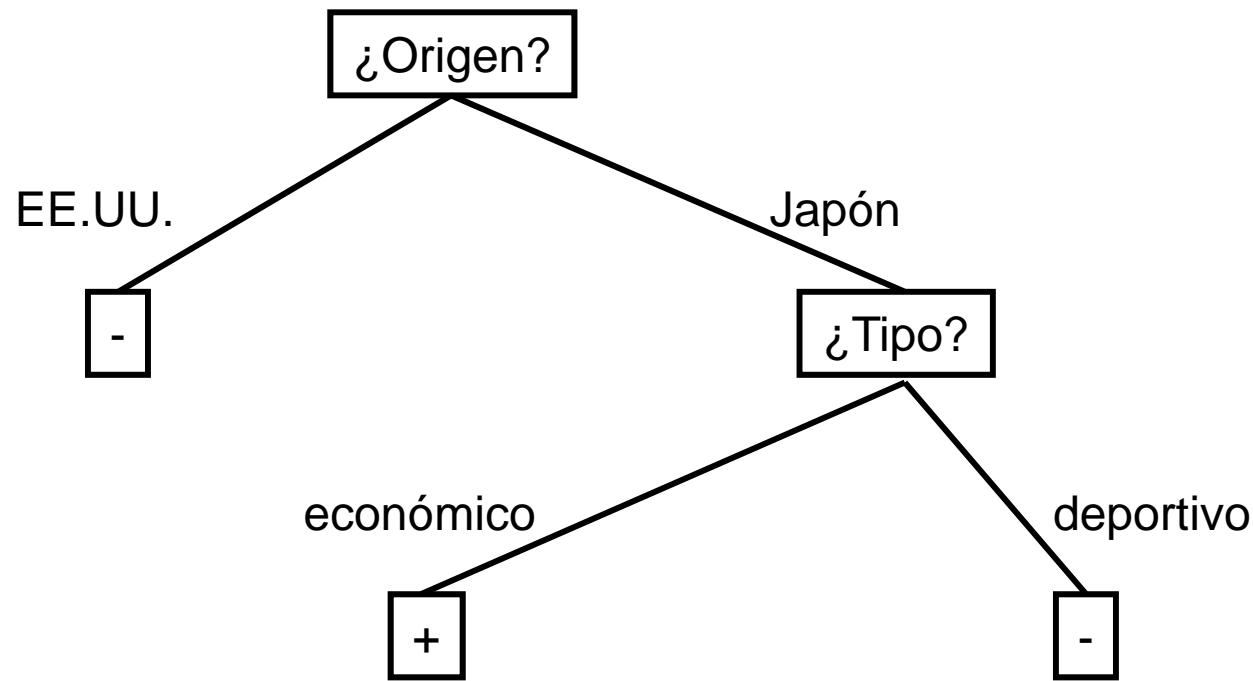
- Problema: un error en el etiquetado de un ejemplo
  - Puede causar: la poda del concepto objetivo en el espacio de versiones
  - P.E.: si el 3º coche ejemplo se etiqueta como negativo por error,
    - no se llega al concepto de coche económico japonés
- Posible solución: mantener varios conjuntos  $G$  y  $E$ 
  - Un conjunto  $G$  es consistente con todas los ejemplos de entrenamiento,
    - otro lo será con todas menos una,
    - otro con todas menos dos, etc. (ídem para  $E$ )
  - Cuando se presenta una inconsistencia
    - el algoritmo cambia de conjuntos de entrenamiento
  - Mantener múltiples espacios de versiones puede resultar muy costoso

## ----- ID3 (Quinlan) -----

- ❑ Inductivo, supervisado y aprendizaje de conceptos
  - ❑ A partir de un conjunto de ejemplos,
  - ❑ Construye un **árbol de decisión** (no necesariamente binario)
  - ❑ Que le permite clasificar casos nuevos (sin etiquetar)
    - ❑ Asignando un valor entre varios posibles
  - ❑ Determina qué atributo es más importante (raíz)
- ❑ Objetivo (qué aprende): construir el árbol de decisión más simple
  - ❑ Que clasifique bien los ejemplos de entrenamiento
- ❑ Sesgo: descripciones con
  - ❑ Conjunciones, disyunciones y negaciones
- ❑ Ventaja: permite aprender varios conceptos simultáneamente
  - ❑ No sólo un concepto y su negación
- ❑ **No es incremental:** usa todos los ejemplos simultáneamente

# Árboles de decisión: ejemplo 1

- Especificación con un árbol de decisión de un concepto
  - ¿Cuál?
    - “Coche económico japonés”
      - **Origen**, marca, color, década, **tipo**



- **Clasificación booleana:** ejemplos positivos o negativos

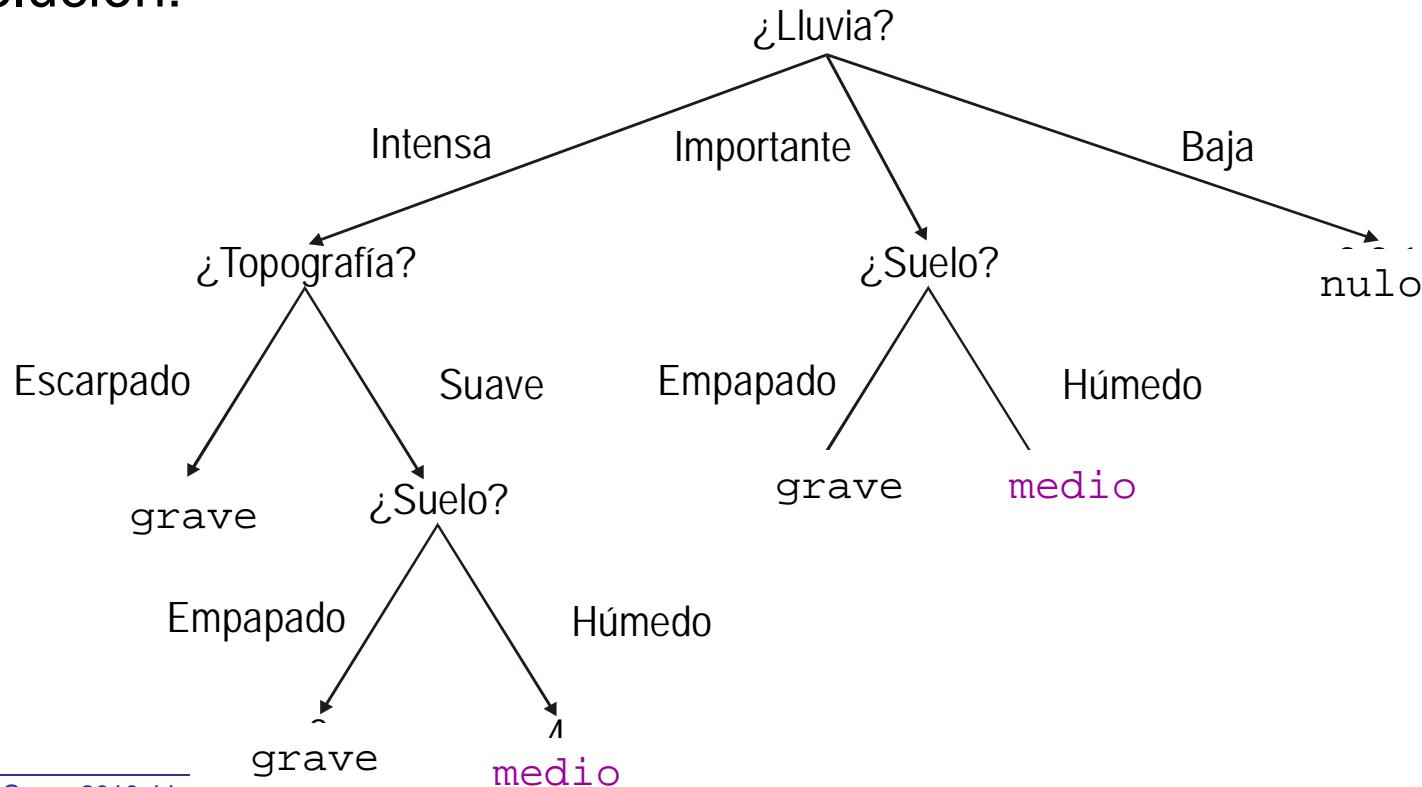
# Árboles de decisión: ejemplo 2

- Aprender a clasificar situaciones de riesgo de inundación
  - En función de los atributos *lluvia, suelo y topografía*
- Ejemplos de entrenamiento:

Caso	Lluvia	Suelo	Topografía	Problema
1	intensa	empapado	escarpada	grave
2	intensa	empapado	suave	grave
3	intensa	húmedo	escarpada	grave
4	intensa	húmedo	suave	medio
5	importante	empapado	escarpada	grave
6	importante	húmedo	escarpada	medio
7	importante	húmedo	suave	medio
8	baja	empapado	escarpada	nulo
9	baja	húmedo	escarpada	nulo
10	baja	húmedo	suave	nulo

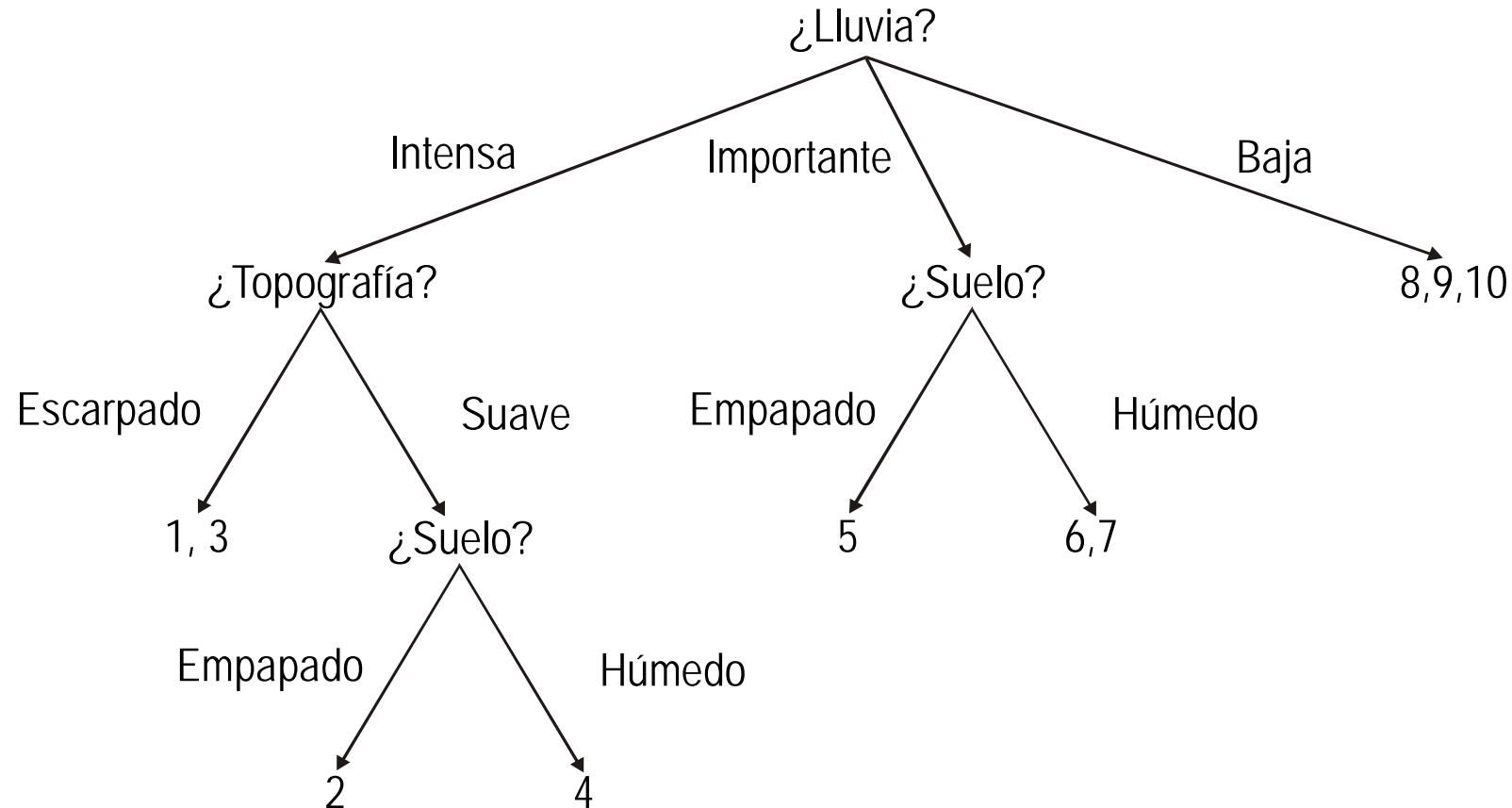
# Árboles de decisión

- Representación natural de criterios de decisión de las personas
  - Nodos: preguntas (atributos o características)
  - Arcos: posibles respuestas (valores posibles en los ejemplos)
  - Hojas están etiquetadas con la predicción (clase de ese ejemplo)
- Solución:



# Árboles de decisión

- Criterio principal en una inundación? : Lluvia ← la raíz



# Construcción de árboles de decisión con ID3

## A) Árbol de decisión trivial

- Un camino de la raíz a una hoja por cada ejemplo
- Memoriza exactamente los ejemplos de entrenamiento, no extrae ningún patrón, luego **no** sirve para extrapolar a ejemplos nuevos

## B) Buscamos el árbol de decisión *más pequeño* que sea consistente con los ejemplos: ¡Problema intratable...!

- El orden en el que se eligen los atributos para construir un árbol de decisión afecta al tamaño del árbol (*el 1º + importante*)
- Usando heurísticas, puede encontrarse uno “más bien” pequeño

## C) Solución: ID3 (Quinlan, 83) o su sucesor C4.5 (Quinlan, 93)

- Algoritmo de búsqueda heurística
- Optimización local: escalada
- Teoría de la información* para estimar el mejor candidato
- Se consigue complejidad lineal

- ❑ Datos sobre los que opera el algoritmo:
  - ❑ Conjunto de ejemplos de entrenamiento
    - ❑ EJ
  - ❑ Conjunto de las clases a las que pueden pertenecer los ejemplos
    - ❑  $Cl_1, Cl_2, \dots, Cl_N$
  - ❑ Conjunto de atributos definidos sobre los ejemplos
    - ❑ A, B, C, ...
  - ❑ Conjuntos con los valores posibles para cada atributo
    - ❑  $A_1, \dots, A_K, B_1, \dots, B_L, C_1, \dots, C_M, \dots$

# ID3

- ❑ El árbol se construye de arriba a abajo, trabajando por niveles
- ❑ En cada iteración del algoritmo se pretende:
  - ❑ Obtener el atributo en base al cual ramificar el nodo problema
  - ❑ Seleccionar el que mejor discrimine entre el conjunto de ejemplos
    - ❑ Heurística para obtener árboles pequeños (en profundidad)
    - ❑ El atributo más discriminante será aquél que conduzca a un estado
      - ❑ con menor **entropía** o menor desorden (mayor información)
- ❑ La entropía (Shannon, 1948) mide la ausencia de homogeneidad de un conjunto de ejemplos con respecto a su clase
  - ❑ Es una medida estándar del desorden (*0 es homogeneidad total*)
    - ❑ *utilizada en física y en la teoría de la información*
- ❑ Ganancia de información es la diferencia entre
  - ❑ la entropía del conjunto original y la de los subconjuntos obtenidos

## ID3

- La entropía se define utilizando resultados de la teoría de la información
- La entropía inicial de un nodo  $X$  (antes de clasificar los ejemplos que contiene en base a alguno de los atributos) es:

$$E(X) = - \sum_{j=1}^N P_X(Cl_j) \cdot \log_2 P_X(Cl_j)$$

donde

$$\log_2 x = \begin{cases} 0 & \text{si } x = 0 \\ \log_2 x & \text{en otro caso} \end{cases}$$

y la probabilidad de una clase  $Cl_j$  en el nodo  $X$  es

$$P_X(Cl_j) = \frac{|n^o \text{ de ejemplos correspondientes a } Cl_j \text{ en } X|}{|n^o \text{ total de ejemplos en } X|}$$

para  $j \in [1..N]$

## ID3

- La **entropía final** del nodo  $X$  al ramificar utilizando el atributo  $A$ , es igual a la suma de las entropías de los nodos resultantes de fijar el valor del atributo multiplicadas por la probabilidad de cada valor

$$E_A(X) = \sum_{i=1}^k P_X(A_i) \cdot E(A_i)$$

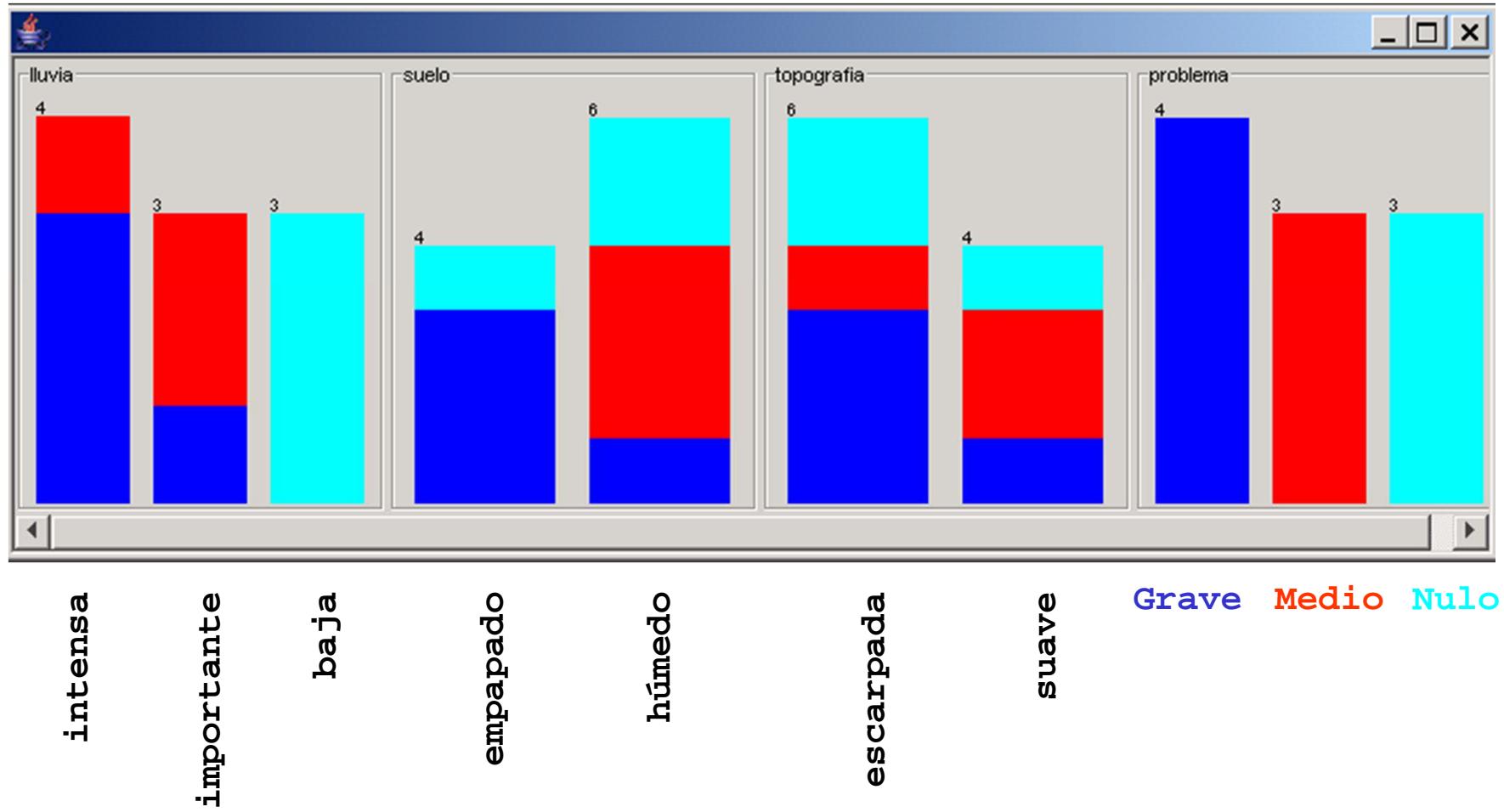
donde la probabilidad  $P_X(A_i)$  es

$$P_X(A_i) = \frac{|n^o \text{ de ejemplos en } X \text{ con atributo } A = A_i|}{|n^o \text{ total de ejemplos en } X|}$$

## ID3

- ❑ Para cada atributo se calcula la disminución de entropía causada por su utilización
  - ❑ Disminución de entropía<sub>A</sub>(X)=  $E(X) - E_A(X)$
  - ❑ Disminución de entropía<sub>B</sub>(X)=  $E(X) - E_B(X)$
  - ❑ Disminución de entropía<sub>C</sub>(X)=  $E(X) - E_C(X) \dots$
- ❑ En cada nodo, se selecciona aquel atributo que mayor disminución de entropía provoca
- ❑ Aplicado al ejemplo
  - ❑ Hay 3 clases
    - ❑ Cl<sub>1</sub> problema grave, Cl<sub>2</sub> problema medio, Cl<sub>3</sub> problema nulo
  - ❑ Y 3 atributos
    - ❑ A lluvia, B suelo, C topografía

# ID3: ejemplo



# ID3: ejemplo

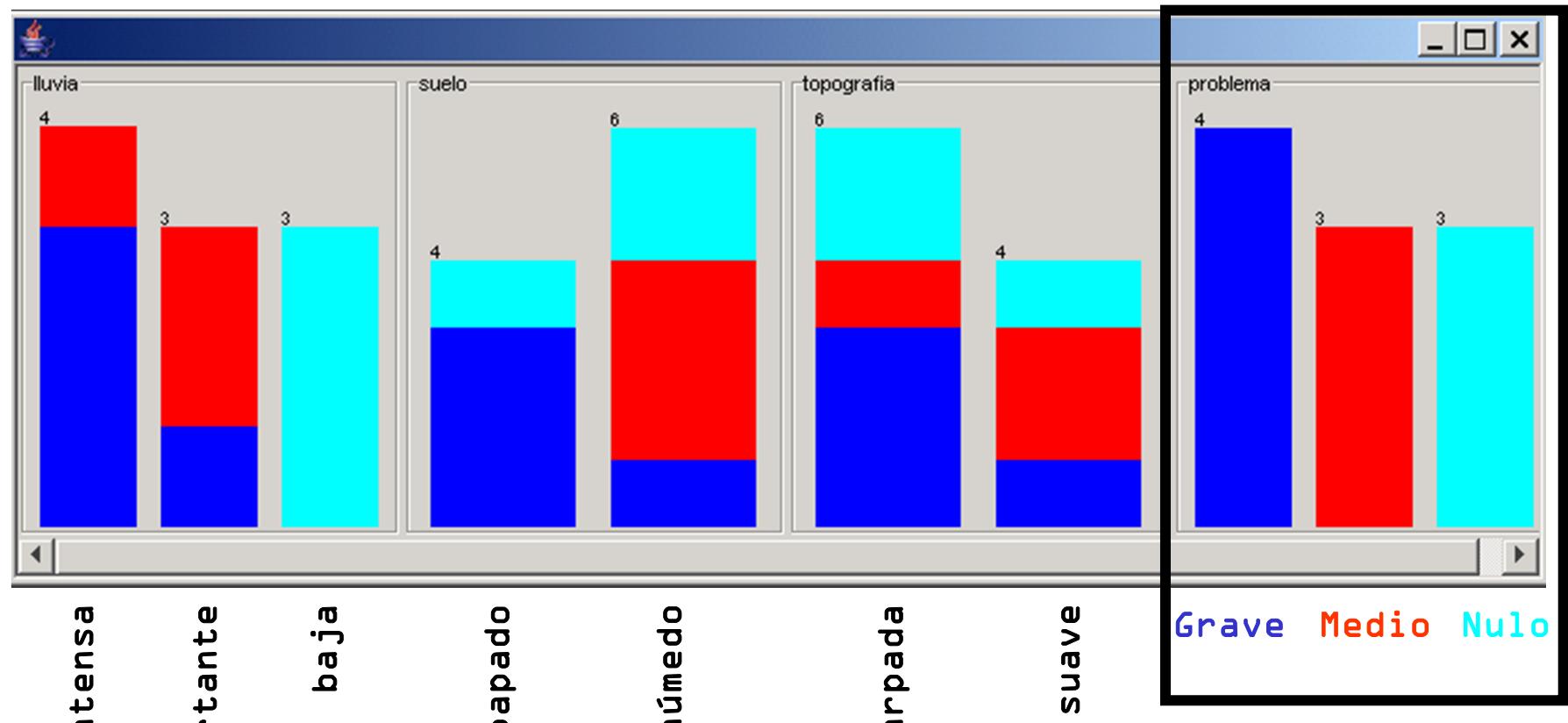
- Entropía inicial en la raíz del árbol: (del problema global)

$$P(\text{grave}) = 0,4$$

$$P(\text{medio}) = 0,3$$

$$P(\text{nulo}) = 0,3$$

$$E(\text{raíz}) = -0,4 \log_2 0,4 - 0,3 \log_2 0,3 - 0,3 \log_2 0,3 = 1,571$$



# ID3: ejemplo

## □ Entropía final clasificando según lluvia (A):

$A_1$ : lluvia intensa,

$A_2$ : lluvia importante,

$A_3$ : lluvia baja

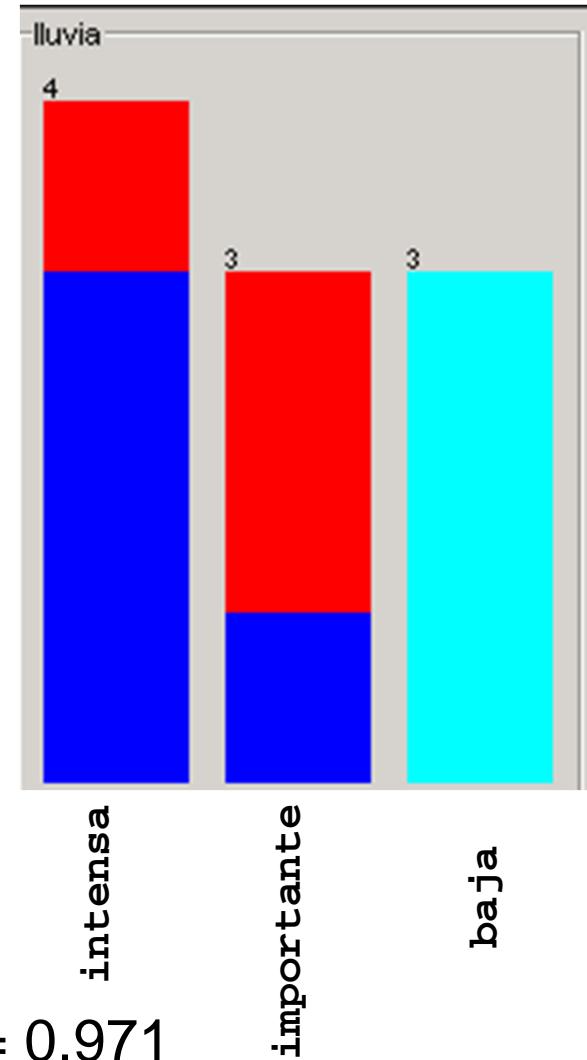
$$E(A_1) = -0,75 \log_2 0,75 - 0,25 \log_2 0,25 = 0,811$$

$$E(A_2) = 0,918$$

$$E(A_3) = 0$$

$$E_A(\text{raíz}) = 0,4 * 0,811 + 0,3 * 0,918 = 0,6$$

Probabilidad de "intensa"

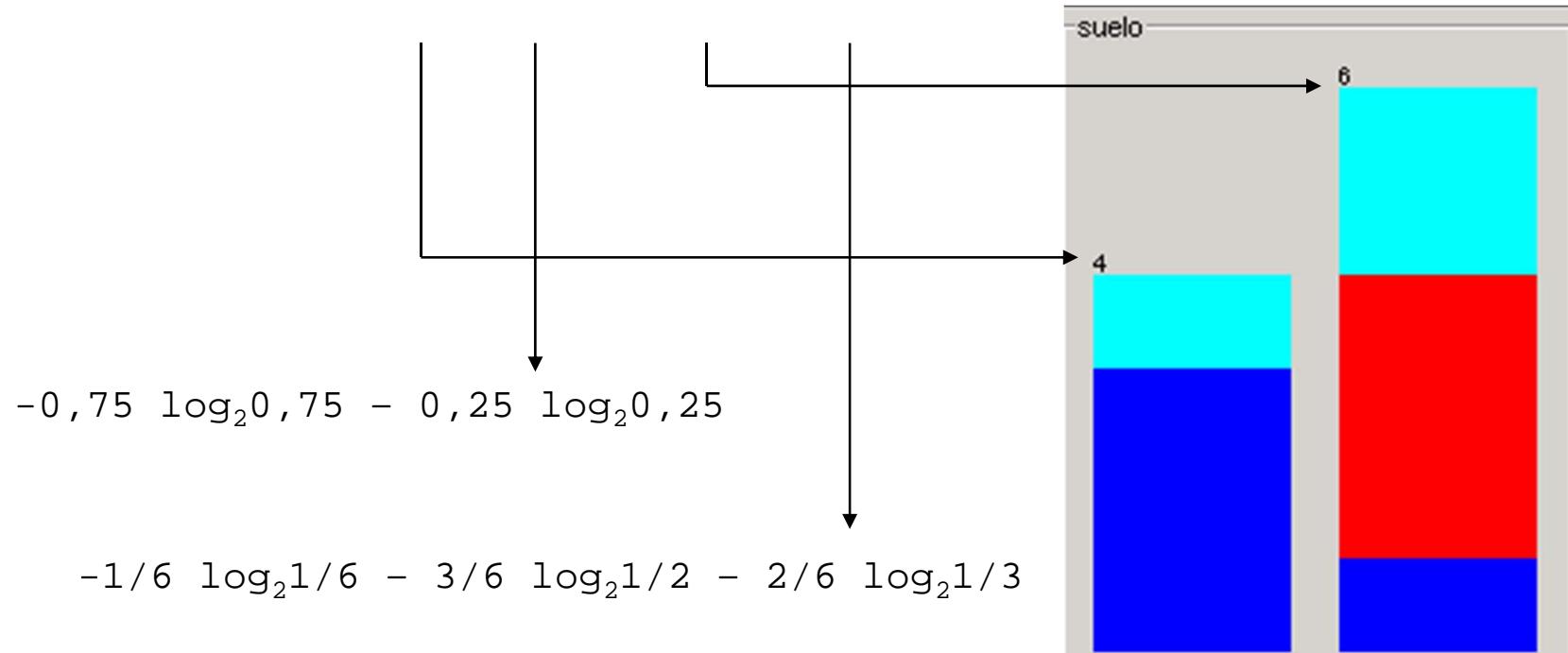


$$\text{Disminución de entropía}_A(\text{raíz}) = 1,571 - 0,60 = 0,971$$

## ID3: ejemplo

- ## □ Entropía final clasificando según suelo(B):

$$E_B(\text{raíz}) = 0,4 * 0,811 + 0,6 * 1,459 = 1,20$$



$$\text{Disminución de entropía}_{\text{B}}(\text{raíz}) = 1,571 - 1,20 = 0,371$$

## ID3: ejemplo

- Entropía final clasificando según topografía (C):

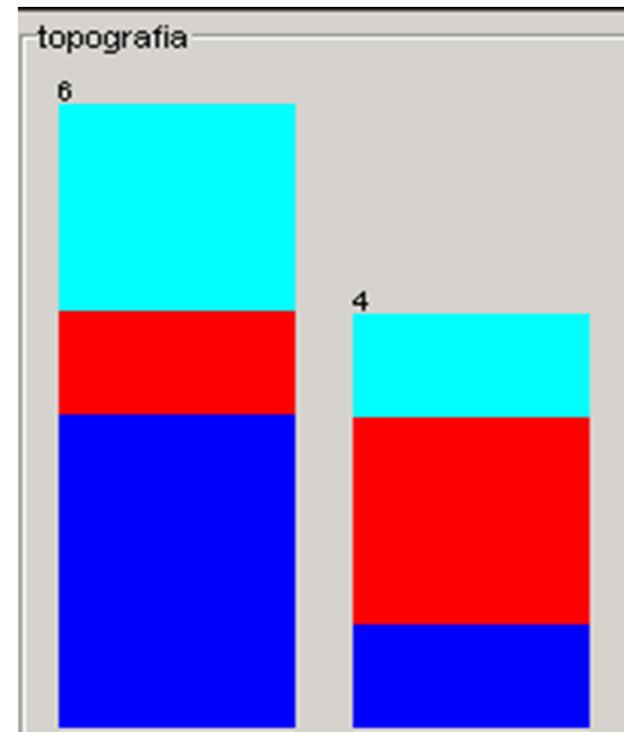
$$-1/4 \log_2 1/4 - 2/4 \log_2 2/4 - 1/4 \log_2 1/4$$



$$E_C(\text{raíz}) = 0,6 \cdot 1,459 + 0,4 \cdot 1,50 = 1,475$$



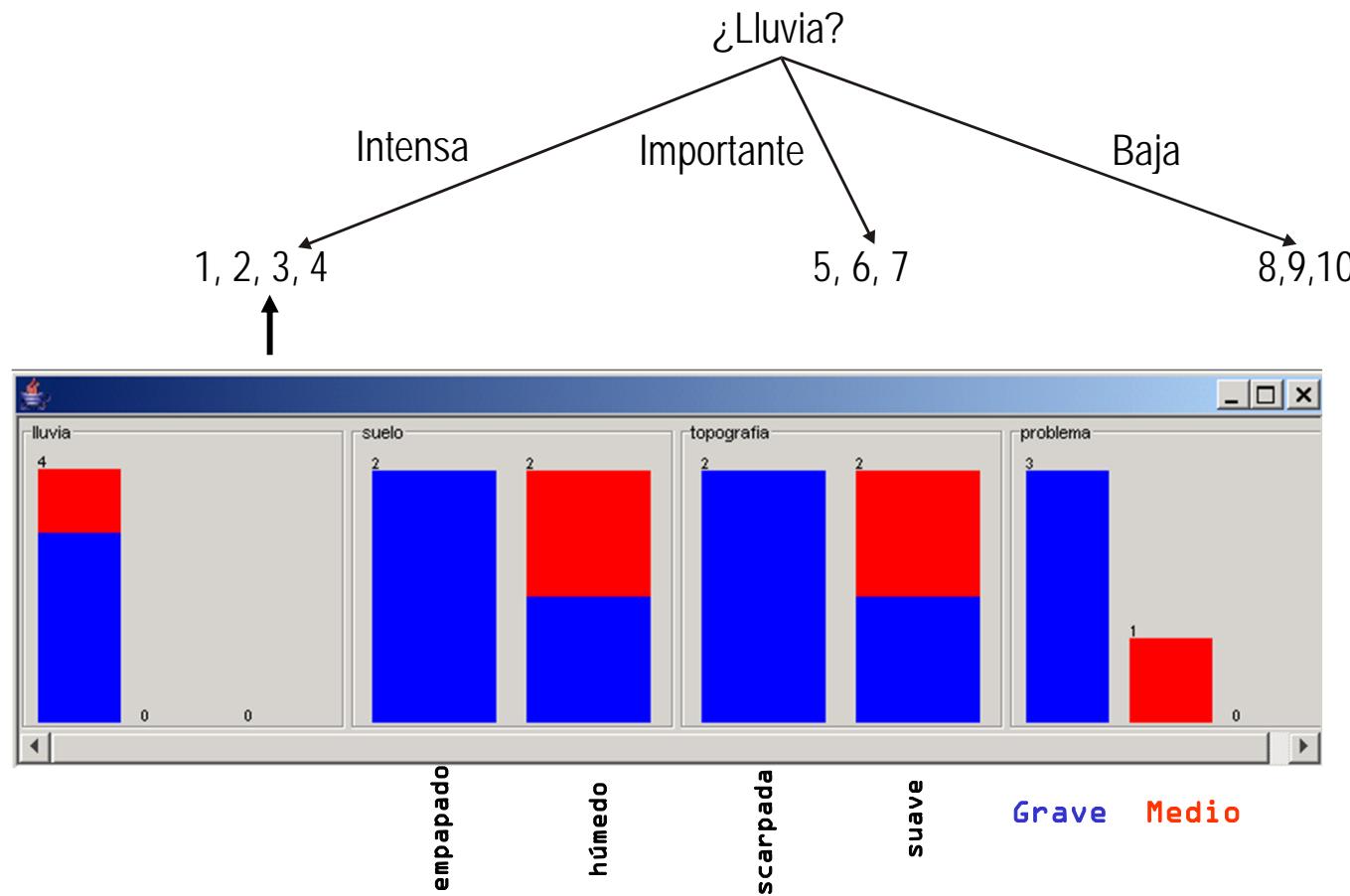
$$-1/6 \log_2 1/6 - 3/6 \log_2 1/2 - 2/6 \log_2 1/3$$



$$\text{Disminución de entropía}_C(\text{raíz}) = 1,571 - 1,475 = 0,096$$

- La mayor disminución de entropía se consigue con el atributo A y por ello éste es el seleccionado para el primer nivel del árbol

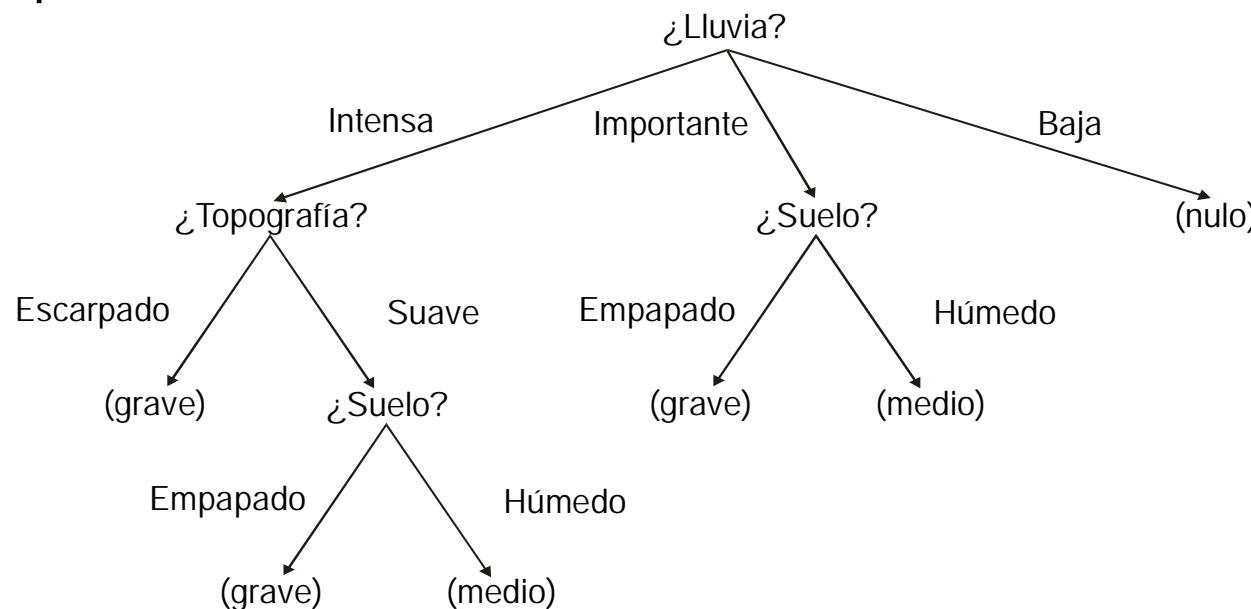
# ID3: ejemplo



- En la siguiente iteración se vuelve a aplicar el algoritmo sobre cada uno de los tres nuevos nodos, considerando en cada uno el subconjunto de ejemplos obtenido y habiendo eliminado el atributo lluvia del conjunto de atributos

## □ Terminación:

- La expansión de un nodo se detiene cuando todos sus ejemplos pertenecen a la misma clase ( $\equiv$  entropía nula)
- El proceso se detiene cuando no se puede seguir expandiendo ningún nodo
- A las hojas se les asigna la clase a la que pertenecen todos sus ejemplos



# ID3: refinamientos

- ❑ Cuando es muy grande, del conjunto de ejemplos disponibles se escoge una **ventana** para construir el árbol (*incremental*)
  - ❑ Si el resto de ejemplos es clasificado correctamente, se detiene
  - ❑ Si no, se añaden elementos a la ventana y se repite el proceso
  - ❑ Análisis empíricos demuestran que esta estrategia es más eficiente que considerar todos los ejemplos desde el principio (*no claro*)
- ❑ **Ruido** en los datos: para mejorar la tolerancia
  - ❑ Sólo se ramifica un nodo cuando la disminución de entropía está por encima de un determinado umbral. A cada nodo terminal se le asigna la clase de la mayoría de sus ejemplos
- ❑ Ha evolucionado dando lugar al algoritmo **C4.5** (Quinlan 93)
  - ❑ Soluciona un pequeño problema de ID3: tiene una cierta tendencia a favorecer la elección de atributos con muchos valores posibles, lo que redunda en una peor generalización de las observaciones
  - ❑ Última versión: C5.0 comercial (*RuleQuest Research, de Quinlan*)

# ----- Aprendizaje por Descubrimiento -----

- ❑ Clasificaciones del aprendizaje:
  - ❑ Grado de realimentación: no supervisado
  - ❑ Paradigma: inductivo
  - ❑ Qué aprende: agrupa en clases
- ❑ Aprendizajes NO supervisados (sin “profesor”)
  - ❑ Formación de taxonomías y clustering conceptual      [← veremos](#)
  - ❑ Descubrimiento de leyes cualitativas
  - ❑ Descubrimiento de leyes cuantitativas
- ❑ Aprendizaje por descubrimiento:
  - ❑ Transformar o reorganizar las experiencias
  - ❑ De manera que se puedan extraer nuevas conclusiones

# Formación de taxonomías

- Escenario (situación):

- Conjunto de objetos / observaciones reales
    - No se sabe a qué clase pertenece cada objeto ni qué clases hay
    - Ej: Conjunto de coches averiados: descubrir qué tienen en común

- Descripción del objeto:

- N pares <atributo, valor> (espacio N dimensiones)

- Similitud entre objetos: distancia entre los puntos que los representan

$$d_E(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

$$d_{ABS}(x, y) = \sqrt{\sum_{i=1}^n |x_i - y_i|}$$

Dist. Euclídea

Dist. Valor absoluto

(pag.264)

- Objetivos:

- Agrupar ejemplos con características similares para formar categorías
  - Crear árbol de clasificación con grupos a diferentes alturas de abstracción
  - maximizar la similitud “intra-cluster”
  - minimizar la similitud entre “clusters”

- REF.: Aprendizaje Automático (B. Sierra Araujo) Pearson

# Formación de taxonomías : Algoritmo

- ❑ FASE 1: Crear Matriz de Distancias, usando  $d_E(x, y)$  ó  $d_{ABS}(x, y)$
- ❑ FASE 2: Agrupación de Individuos (técnica jerárquica aglomerativa)
  1. Partición inicial  $P_0$ : Cada objeto es un cluster  $P_0 = \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}\}$
  2. Partición siguiente, usando M. Distancias:
    - ❑ Elegir los dos objetos más cercanos (parecidos)  $\{\{i\}, \{j\}\} \Rightarrow \{\{i, j\}\}$ 
      - ❑ Mínimo en la matriz D
    - ❑ Agrupar los dos en un cluster (objeto nuevo)
      - ❑ Definir la distancia de cualquier objeto  $\{l\}$  al nuevo:
$$d'(l, \{i, j\}) = \min(d(l, i), d(l, j)), l \neq i, j$$
        - ❑ Eliminar los dos originales
        - ❑ Recalcular la Matriz de Distancias con los objetos actuales
      - ❑ Índice heterogeneidad  $\alpha(C_k \cup C_h) = d'(C_k, C_h)$ 
        - ❑ Distancia entre  $C_k$  y  $C_h$  cuando se unen  $C_k$  y  $C_h$
- 3. Repetir paso 2 hasta tener solo un cluster con todos los individuos
  - ❑ Formar clases a diferentes niveles: Dendograma o Árbol de Clasificación

## Formación de taxonomías : Ejemplo del Algoritmo

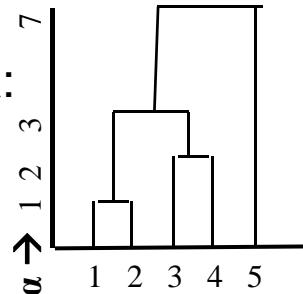
- Ejemplo (ej.: p.275): Dadas las observaciones  $\Omega = \{1,2,3,4,5\}$

y su matriz de distancias:

$$D = \begin{pmatrix} 0 & 1 & 3 & 4 & 7 \\ 1 & 0 & 4 & 4 & 8 \\ 3 & 4 & 0 & 2 & 8 \\ 4 & 4 & 2 & 0 & 7 \\ 7 & 8 & 8 & 7 & 0 \end{pmatrix}$$

El algoritmo crea esta jerarquía aglomerativa:

- Dendograma de la jerarquía:



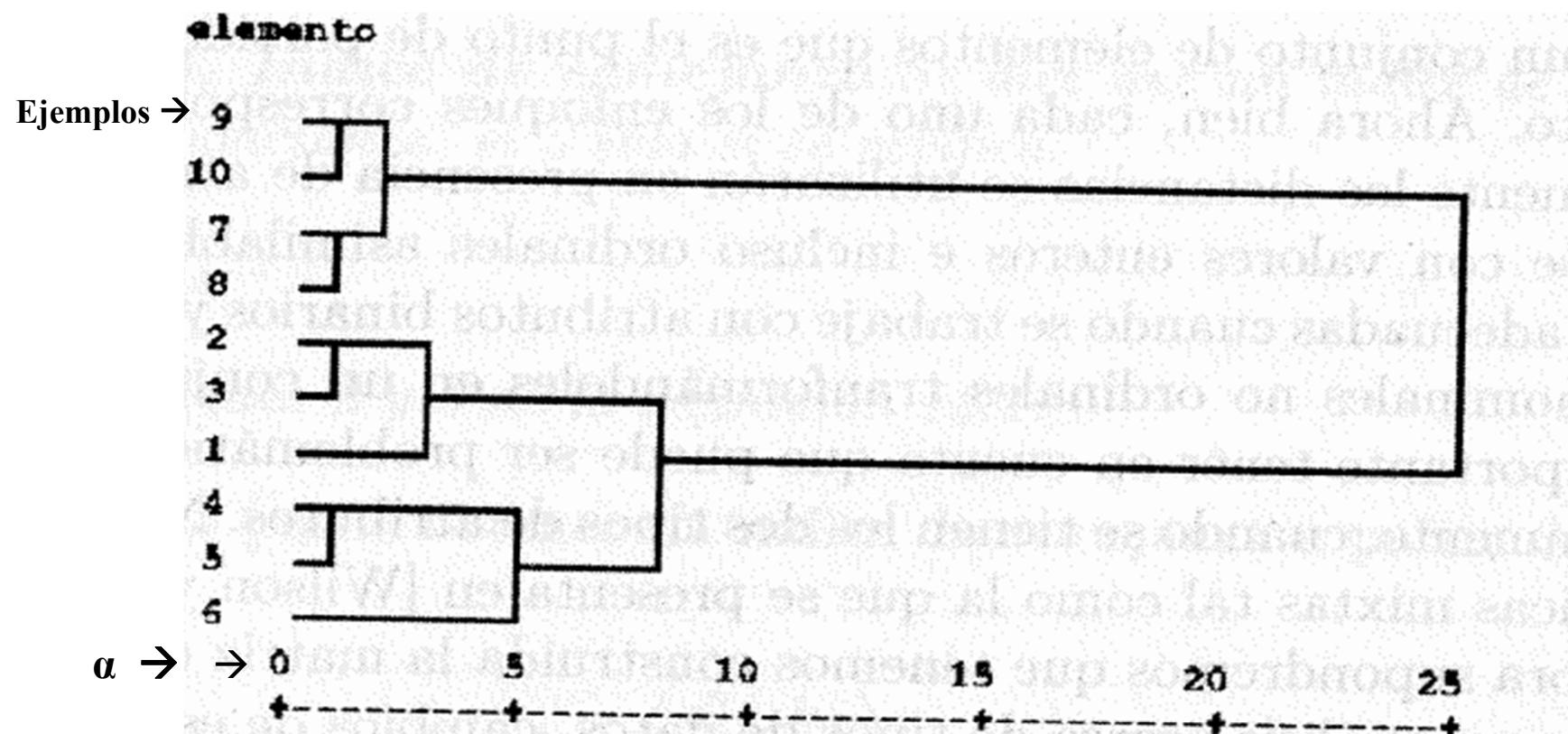
$$\begin{aligned} P_0 &= \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}\} \\ P_1 &= \{\{1, 2\}, \{3\}, \{4\}, \{5\}\} \\ P_2 &= \{\{1, 2\}, \{3, 4\}, \{5\}\} \\ P_3 &= \{\{1, 2, 3, 4\}, \{5\}\} \\ P_4 &= \{\{1, 2, 3, 4, 5\}\} = \{\Omega\} \end{aligned}$$

- Según a qué altura se corte en horizontal se producen clases diferentes
- Algunos ejemplos más
  - P. 700 Inteligencia Artificial (J.T.Palma, R. Martín)
  - <http://anthropologynet.files.wordpress.com/2008/02/population-dendogram.jpg>

## Formación de taxonomías: Otro ejemplo, matriz dist.

	1	2	3	4	5	6	7	8	9	10
1	,000	1,490	5,440	2,440	8,290	14,690	22,690	21,640	38,090	36,040
2	1,490	,000	1,250	4,250	8,000	9,000	25,000	21,250	41,000	36,250
3	5,440	1,250	,000	9,000	11,250	7,250	31,250	25,000	48,250	41,000
4	2,440	4,250	9,000	,000	2,250	10,250	10,250	10,000	21,250	20,000
5	8,290	8,000	11,250	2,250	,000	5,000	5,000	3,250	13,000	10,250
6	14,690	9,000	7,250	10,250	5,000	,000	16,000	9,250	26,000	18,250
7	22,690	25,000	31,250	10,250	5,000	16,000	,000	1,250	2,000	2,250
8	21,640	21,250	25,000	10,000	3,250	9,250	1,250	,000	4,250	2,000
9	38,090	41,000	48,250	21,250	13,000	26,000	2,000	4,250	,000	1,250
10	36,040	36,250	41,000	20,000	10,250	18,250	2,250	2,000	1,250	,000

# Formación de taxonomías: Otro ejemplo, Agrupamiento



1. Partición  $\{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}, \{8\}, \{9\}, \{10\}\}$
2. Partición  $\{1, \{2, 3\}, \{4, 5\}, 6, \{7, 8\}, \{9, 10\}\}$
3. Partición  $\{1, \{2, 3\}, \{4, 5\}, 6, \{7, 8, 9, 10\}\}$
4. Partición  $\{\{1, 2, 3\}, \{4, 5\}, 6, \{7, 8, 9, 10\}\}$
5. Partición  $\{\{1, 2, 3\}, \{4, 5, 6\}, \{7, 8, 9, 10\}\}$
6. Partición  $\{\{1, 2, 3, 4, 5, 6\}, \{7, 8, 9, 10\}\}$
7. Partición  $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$

## --Paradigma: Aprendizaje analítico/deductivo--

- Se aplica habitualmente a **resolución de problemas**
- Utiliza:
  - Muy pocos ejemplos (normalmente 1)
  - Teoría del dominio (en forma de reglas)
- El ejemplo guía las cadenas deductivas
  - Para resolver nuevos problemas
  - Para formular reglas de control de búsqueda más eficiente
- El ejemplo del que se parte
  - Es una descripción ó
  - Es una traza de la resolución del problema
    - junto con anotaciones sobre la justificación de las decisiones adoptadas.
- Este conocimiento se generaliza aplicando la teoría del dominio
- No intentan ampliar lo que sabe hacer el sistema
  - Sino mejorar su eficiencia

## ---Aprendizaje basado en explicaciones (EBL)---

- ❑ Clasificaciones del aprendizaje:
  - ❑ Realimentación: supervisado
  - ❑ Paradigma: analítico / deductivo (usa teoría del dominio)
  - ❑ Qué aprende: definición estructural de un concepto
    - ❑ Permite clasificar eficientemente un ejemplo desconocido
- ❑ Aplicable cuando se dispone de
  - ❑ Descripción funcional del concepto en estudio
  - ❑ Un ejemplo positivo de ese concepto
  - ❑ Una teoría del dominio en forma de reglas

# Sistemas EBL

- ❑ Entrada:
  - ❑ C: Concepto objetivo (a aprender) descrito funcionalmente
  - ❑ E: Ejemplo positivo (de C) descrito en estructura o lo físico
  - ❑ TD: Teoría del dominio (axiomas y reglas de inferencia)
    - ❑ Relaciona características estructurales con funcionales
    - ❑ Permite probar que E es un ejemplar de C usando TD
  - ❑ CO: Criterio de operacionalidad : debe cumplirlo cualquier
- ❑ Salida: descripción estructural más genérica (aprendida)
  - ❑ Utilizada para reconocer cualquier ejemplo desconocido
    - ❑ como de la clase del concepto objetivo

# Ejemplo EBL

(C) Concepto objetivo: TAZA

Definición funcional:

RECIPIENTE\_ABIERTO(x)  $\cap$  ESTABLE(x)  $\cap$  ALZABLE(x)  $\leftrightarrow$  TAZA(x)

(E) Ejemplo de entrenamiento OBJ1:

TIENE\_PARTE(OBJ1, CONCAVIDAD1)

COLOR(OBJ1, ROJO)

CONCAVIDAD(CONCAVIDAD1)

ORIENTADA\_HACIA\_ARRIBA(CONCAVIDAD1)

TIENE\_DUEÑO(OBJ1, SAM)

TIENE\_PARTE(OBJ1, FONDO1)

FONDO(FONDO1)

PLANO(FONDO1)

LIGERO(OBJ1)

TIENE\_PARTE(OBJ1, ASA1)

ASA(ASA1)

LONGITUD(ASA1, 5)

# Ejemplo EBL

(TD) Teoría del dominio:

(R1) TIENE\_PARTE(OBJ, F)  $\cap$  FONDO(F)  $\cap$  PLANO(F)  $\rightarrow$

ESTABLE(OBJ)

(R2) TIENE\_PARTE(OBJ, C)  $\cap$  CONCAVIDAD(C)  $\cap$   
ORIENTADA\_HACIA\_ARRIBA(C)  $\rightarrow$  RECIPIENTE\_ABIERTO(OBJ)

(R3) TIENE\_PARTE(OBJ, A)  $\cap$  ASA(A)  $\cap$  LIGERO(OBJ)  $\rightarrow$

ALZABLE(OBJ)

(CO) Criterio operacional:

El concepto tiene que definirse en términos de los  
predicados usados en el ejemplo.

# Aprendizaje EBL : Algoritmo

## Pasos:

1. Crea explicación: porqué un ejemplo satisface una definición funcional
  - Hacer backward chaining: empieza con el objetivo la definición funcional del (C)
  - Crear el árbol usando como hechos los contenidos en el ejemplo (E)
2. Crea una descripción estructural con la explicación
3. Generalizar la explicación:
  - Transformar los nodos hoja con variables
  - Formar un conjunto de términos : la Des. estructural buscada

## Explicación teórica del proceso de aprendizaje:

- Se usa la teoría del dominio para construir una explicación (demostración) de que el ejemplo de entrenamiento es un ejemplo positivo del concepto objetivo. Los nodos terminales del árbol de explicación tienen que ser operativos
- Transformar los nodos terminales en un conjunto de condiciones suficientes para que la demostración siga siendo válida (generalizar la explicación de acuerdo con la teoría del dominio)

# Aprendizaje EBL : Algoritmo

+ general - operacional

- general + operacional

Conceptos  
aprendidos:

Ejemplos:

Concepto  
objetivo:

atributo-i, atributo-j

Reglas TD

$C_1$ : atributo-k, atributo-l

Reglas TD

$E_1$ : atributo-1, atributo-3

$C_2$ : atributo-m, atributo-n

Reglas TD

$E_2$ : atributo-4, atributo-5

# Ejemplo EBL: Descripción Estructural Aprendida

Resultado del proceso de aprendizaje:

```
TIENE_PARTE(X, Y) ∩ CONCAVIDAD(Y) ∩
ORIENTADA_HACIA_ARRIBA(Y) ∩
TIENE_PARTE(X, Z) ∩ FONDO(Z) ∩ PLANO(Z) ∩
TIENE_PARTE(X, W) ∩ ASA(W) ∩ LIGERO(X)
```

# Métodos analíticos: Ventajas e Inconvenientes

## Ventajas:

- Proporcionan justificación lógica de la descripción del concepto que han obtenido
- Sólo necesitan un ejemplo positivo
- Permiten disyunciones y conjunciones
- Tolerancia al ruido

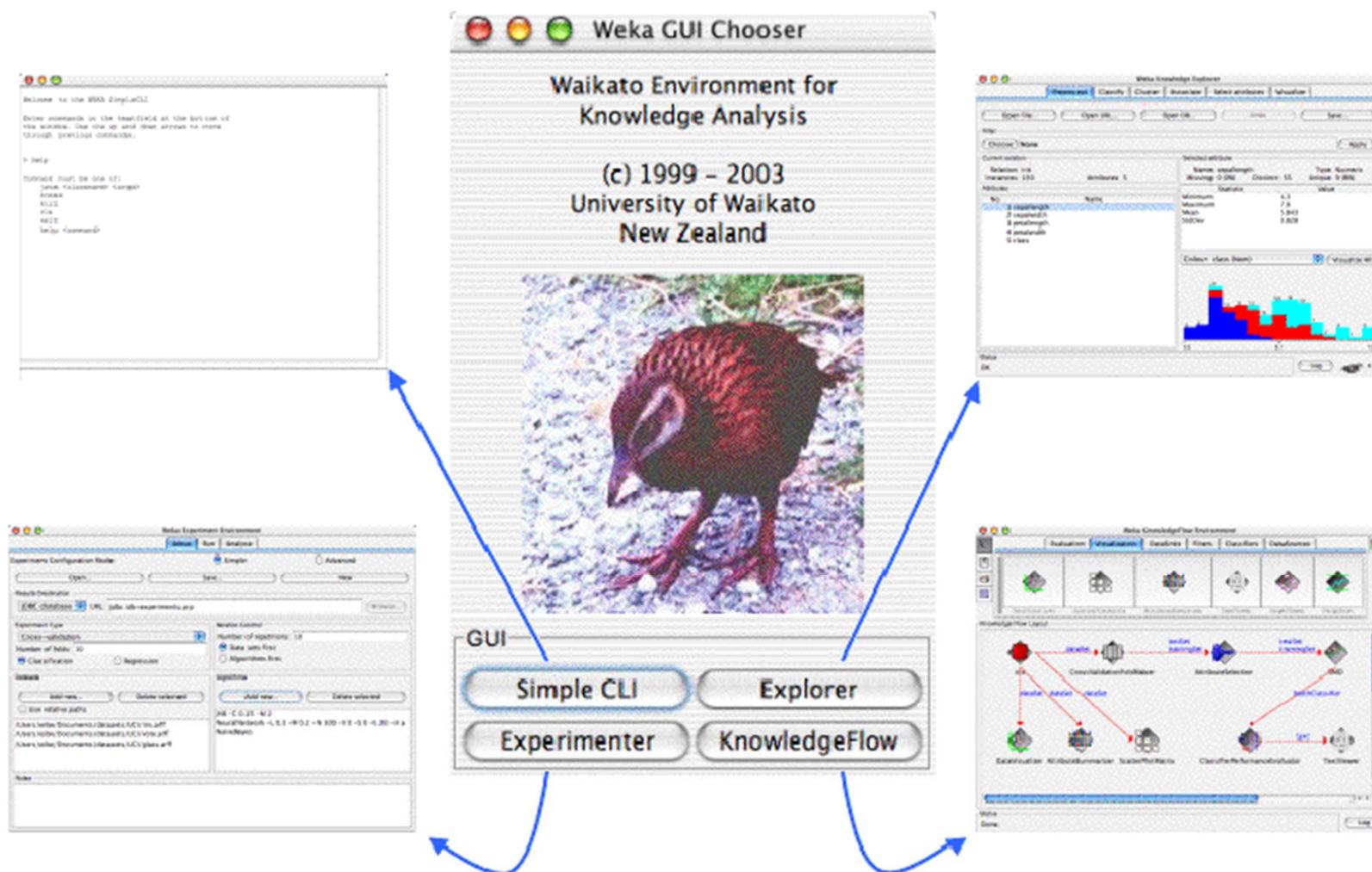
## Inconvenientes:

- Requieren considerable conocimiento del dominio  
(→ menor aplicabilidad)
- Sustituyen la búsqueda en el espacio de descripciones de conceptos por búsqueda en el espacio de explicaciones)

## Weka ([www.cs.waikato.ac.nz/ml/weka/](http://www.cs.waikato.ac.nz/ml/weka/))

- ❑ Weka: *Waikato Environment for Knowledge Analysis*
- ❑ Weka es un conjunto de herramientas sobre *Machine Learning*
- ❑ Incluye una colección de algoritmos de *Machine Learning* para tareas de *data mining*
- ❑ Está escrito en Java y distribuido con licencia GNU pública
- ❑ Los algoritmos pueden ser aplicados directamente a un conjunto de datos de entrada o invocados desde un programa Java
- ❑ Contiene herramientas para:
  - ❑ Preprocesamiento de datos
  - ❑ Clasificación (aprendizaje supervisado)
  - ❑ Regresión (datos continuos y no discretos)
  - ❑ Agrupamiento (*clustering*) (aprendizaje no supervisado)
  - ❑ Reglas de asociación
  - ❑ Visualización

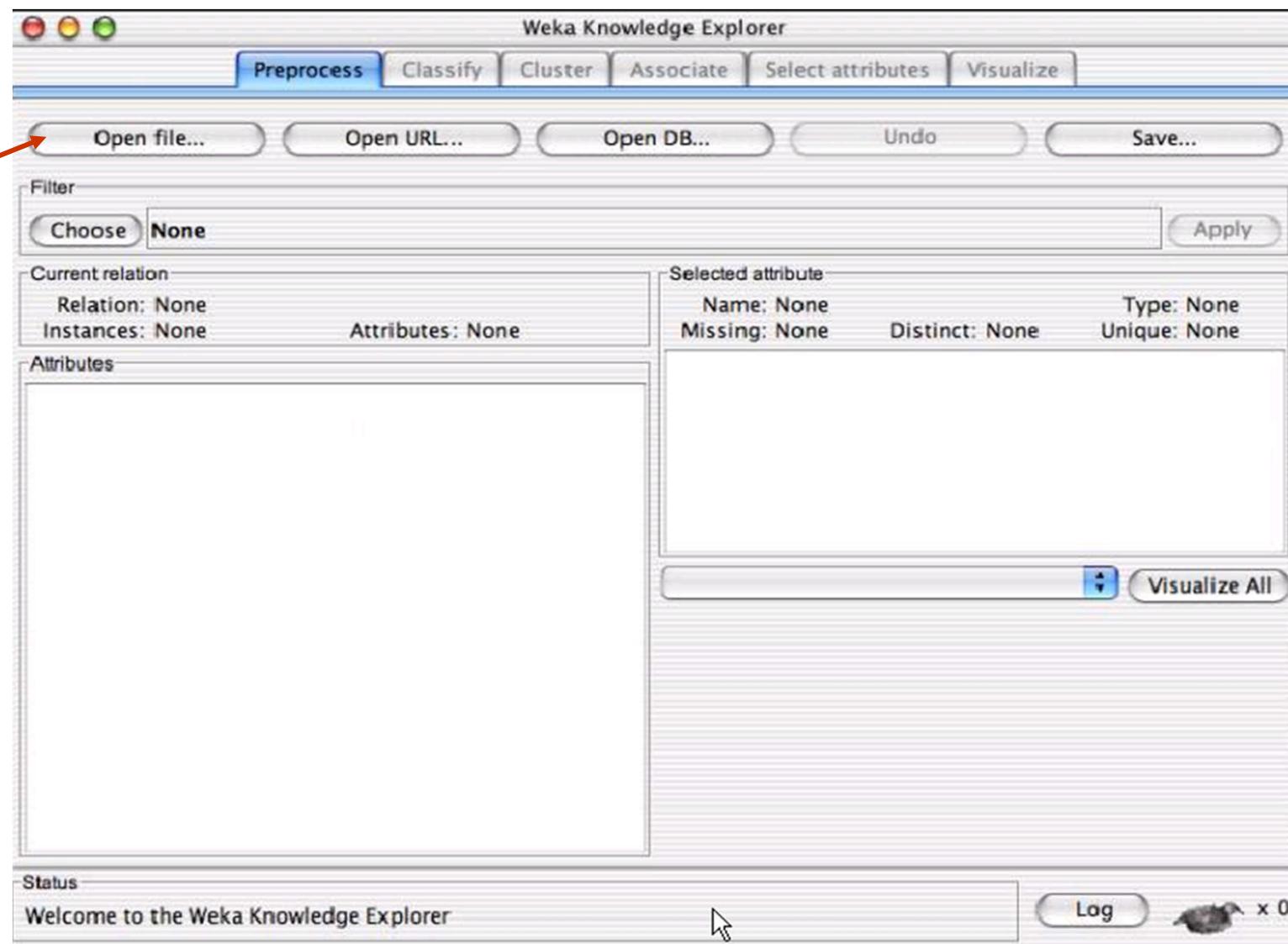
# Weka



# Explorer: preprocessamiento de datos

- ❑ Los datos pueden importarse de varios formatos
  - ❑ El más común es ARFF (Attribute-Relation File Format)
- ❑ También se pueden leer de una URL o de una base de datos (resultado de una consulta SQL) usando JDBC
- ❑ Las herramientas de preprocessamiento en Weka se llaman “filtros”
  - ❑ Discretización, normalización, selección de atributos, transformación de atributos, ...

# Weka Knowledge Explorer



# Weka Knowledge Explorer

The screenshot shows the Weka Knowledge Explorer interface. The top menu bar includes Preprocess, Classify, Cluster, Associate, Select attributes, and Visualize. Below the menu are buttons for Open file..., Open URL..., Open DB..., Undo, and Save... The Filter section shows 'None' selected. The 'Current relation' section displays 'Relation: tiempo', 'Instances: 14', and 'Attributes: 5'. The 'Attributes' table lists attributes: 1. pronostico, 2. temperatura, 3. humedad, 4. viento, and 5. jugar-tenis. The 'Selected attribute' section shows 'pronostico' as a Nominal attribute with 3 distinct values: soleado (5 instances), nublado (4 instances), and lluvioso (5 instances). A bar chart at the bottom visualizes the 'jugar-tenis' attribute, showing 5 instances for each value (soleado, nublado, lluvioso), with the top half colored red and the bottom half blue.

No.	Name
1	pronostico
2	temperatura
3	humedad
4	viento
5	jugar-tenis

Label	Count
soleado	5
nublado	4
lluvioso	5

Colour: jugar-tenis (Nom)

Visualize All

5

4

5

# Formato de archivo: texto plano

ARFF

```
@relation tiempo
@attribute pronostico {soleado, nublado, lluvioso}
@attribute temperatura real
@attribute humedad real
@attribute viento {SI, NO}
@attribute jugar-tenis {si, no}
@data
soleado,85,85,NO,no
soleado,80,90,SI,no
nublado,83,86,NO,si
lluvioso,70,96,NO,si
lluvioso,68,80,NO,si
lluvioso,65,70,SI,no
nublado,64,65,SI,si
soleado,72,95,NO,no
soleado,69,70,NO,si
lluvioso,75,80,NO,si
soleado,75,70,SI,si
nublado,72,90,SI,si
nublado,81,75,NO,si
lluvioso,71,91,SI,no
```

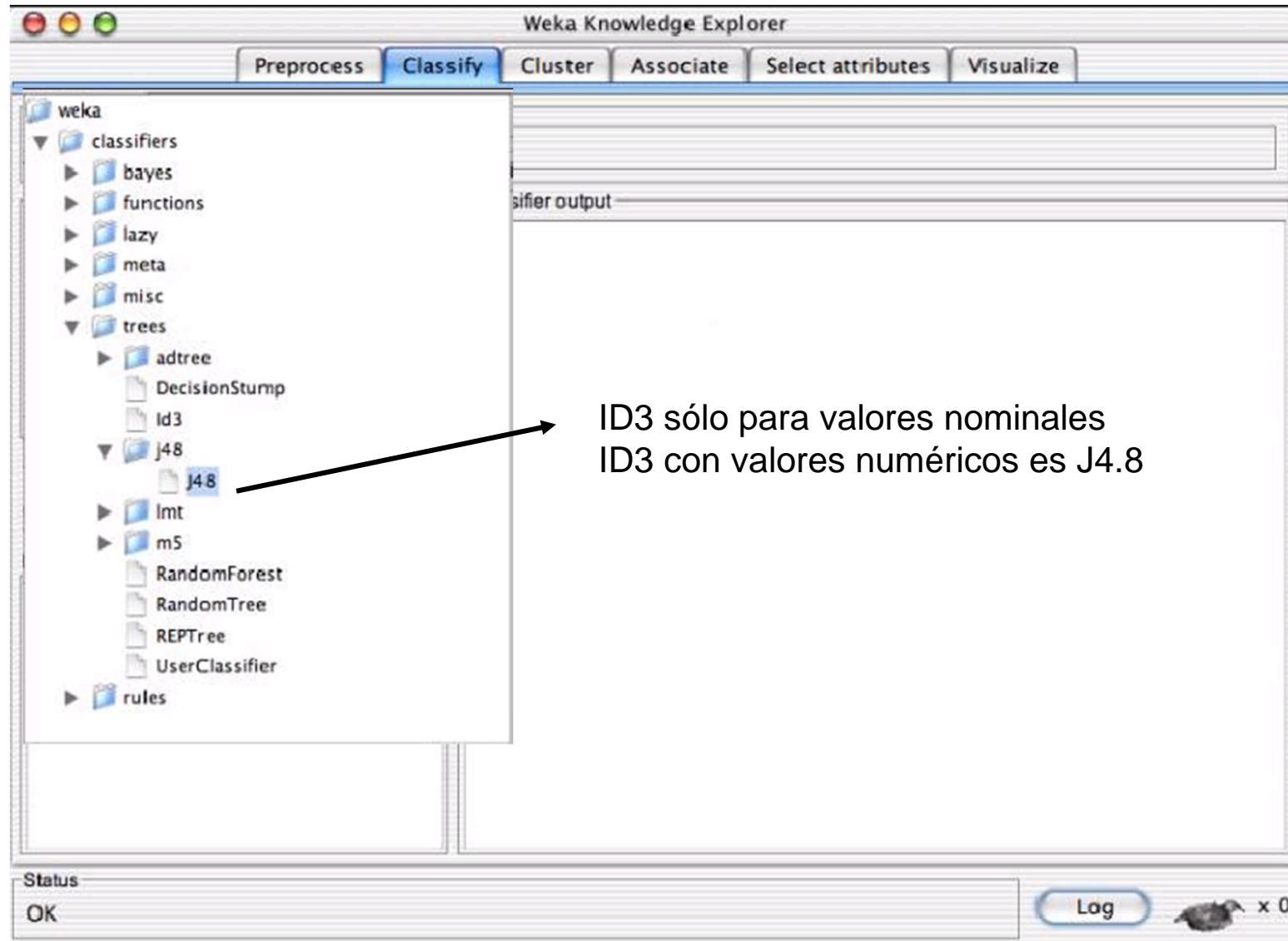
Nombre que se asigna al conjunto de datos

Atributo nominal

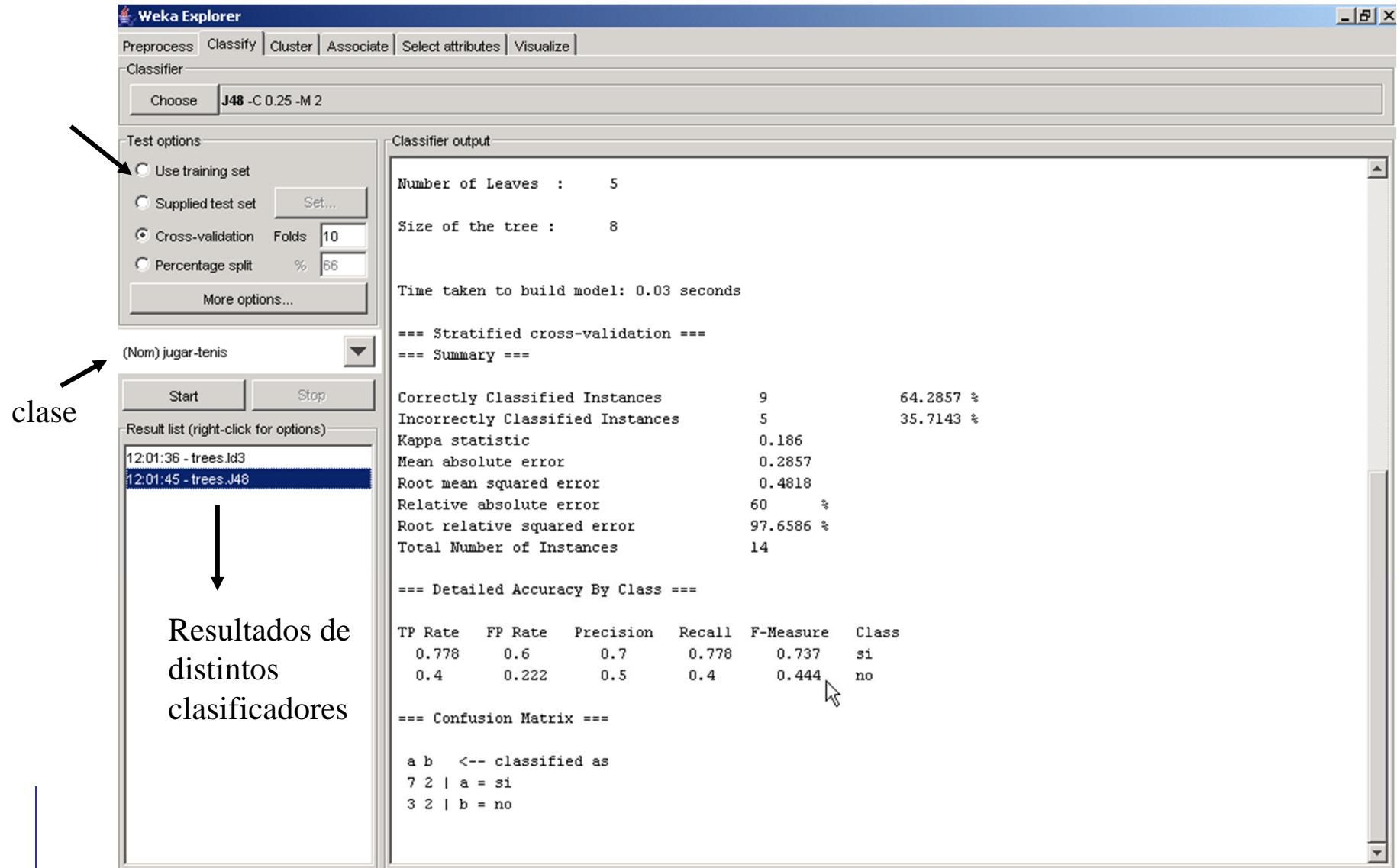
Atributo numérico

El último atributo es el sobre el que se construye el clasificador. Es el atributo que queremos predecir

# Cómo construir un clasificador



# Cómo construir un clasificador



# Resultado

```
== Run information ==
```

```
Scheme:      weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:    tiempo
Instances:   14
Attributes:  5
              pronostico
              temperatura
              humedad
              viento
              jugar-tenis
Test mode:   evaluate on training data
```

```
== Classifier model (full training set) ==
```

```
J48 pruned tree
-----
pronostico = soleado
|   humedad <= 75: si (2.0)
|   humedad > 75: no (3.0)
pronostico = nublado: si (4.0)
pronostico = lluvioso
|   viento = SI: no (2.0)
|   viento = NO: si (3.0)
```

Number of Leaves : 5

Size of the tree : 8

# Resultado

Weka Classifier Tree Visualizer: 12:05:38 - trees.J48 (tiempo)

Tree View

```
graph TD; A([pronostico]) -- "soleado" --> B([humedad]); A -- "nublado" --> C([viento]); A -- "lluvioso" --> D([viento]); B -- "<= 75" --> E([si(2.0)]); B -- "> 75" --> F([no(3.0)]); C -- "SI" --> G([no(2.0)]); C -- "NO" --> H([si(3.0)])
```

(Nom) jugar-tenis

Start Stop

Result list (right-click for options)

12:05:38 - trees.J48

- View in main window
- View in separate window
- Save result buffer
- Load model
- Save model
- Re-evaluate model on current test set
- Visualize classifier errors
- Visualize tree
- Visualize margin curve
- Visualize threshold curve
- Visualize cost curve

FP Rate Preci:

FP Rate	Preci
0	1
0	1

confusion Matrix ===

a b <-- classified as

9 0 | a = si

0 5 | b = no