Tema 12: Arboles de decisión

José A. Alonso Jiménez
Miguel A. Gutiérrez Naranjo

Dpto. de Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla

¿Qué es el Aprendizaje mediante árboles de decisión?

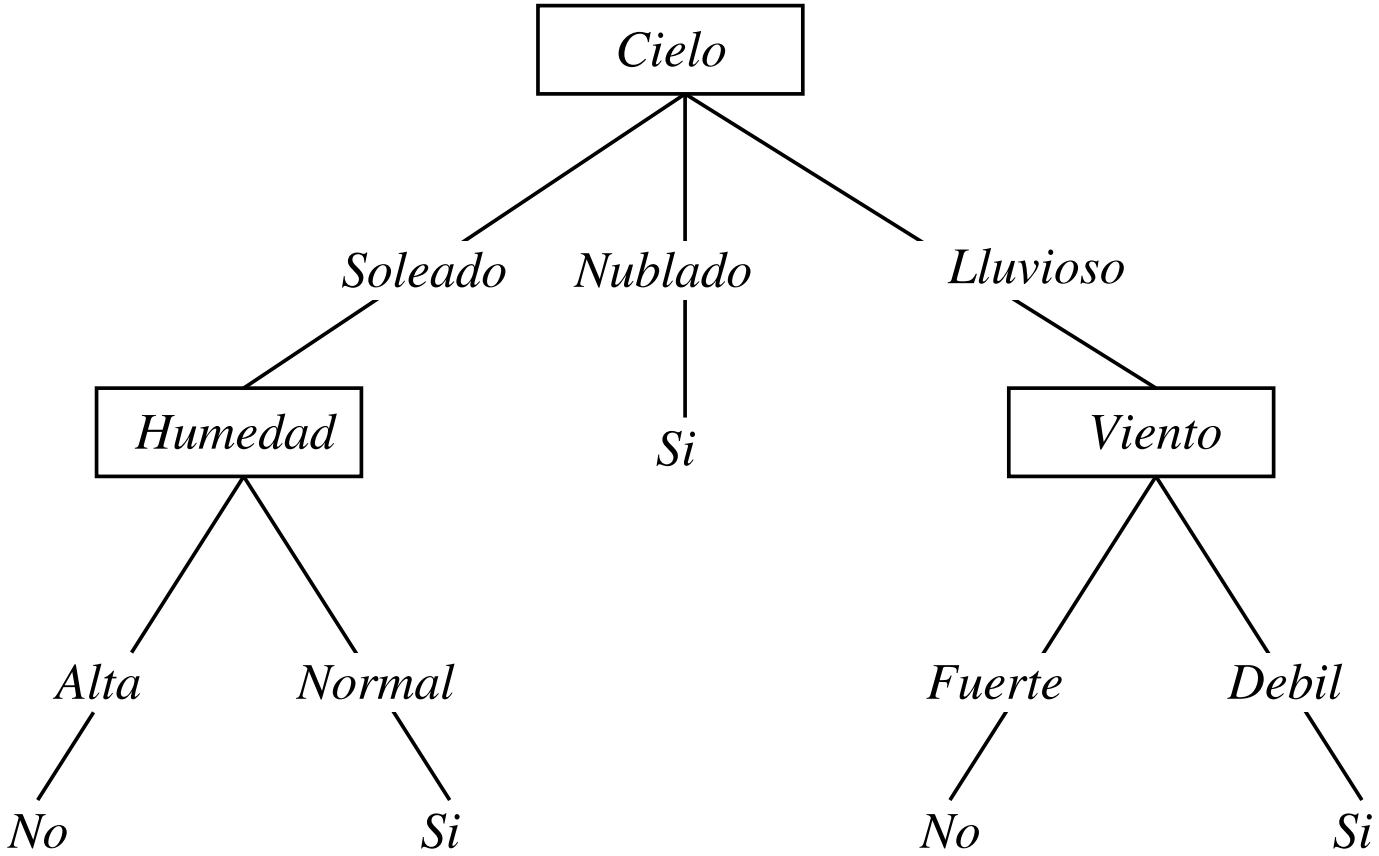
- El aprendizaje mediante árboles de decisión es un método de aproximación de una función objetivo de valores discretos en el cual la función objetivo es representada mediante un árbol de decisión.
- Los árboles aprendidos también pueden representarse como un conjunto de reglas *Si–entonces*

Un ejemplo

Cielo	Temperatura	Humedad	Viento	Agua	Forecast	Hacer_Deporte
Soleado	Templada	Alta	Débil	Fría	Sigue_igual	No
Soleado	Templada	Normal	Débil	Fría	Sigue_igual	Sí
Nublado	Templada	Normal	Débil	Fría	Sigue_igual	Sí
Lluvioso	Templada	Normal	Fuerte	Templada	Cambio	No
Lluvioso	Templada	Normal	Débil	Fría	Cambio	Sí

- ¿Cuándo hacemos deporte?
- ¿Podemos definir el concepto *Hacer_Deporte*?

Un ejemplo de árbol de decisión



El problema de la representación

El árbol anterior corresponde a la hipótesis:

$$\begin{aligned} & (Cielo = Soleado \wedge Humedad = Normal) \\ \vee\ & (Cielo = Nublado) \\ \vee\ & (Cielo = Lluvioso \wedge Viento = Debil) \end{aligned}$$

Representación:

- Cada nodo que no sea una hoja representa un atributo.
- Las aristas que partan del nodo etiquetado con el atributo A están etiquetadas con cada uno de los posibles valores del atributo A.
- Cada hoja corresponde a un valor de la clasificación

Cuestiones sobre árboles de decisión

- ¿Cuándo usar árboles de decisión?
  - Podemos describir los ejemplos en términos de pares atributo–valor
  - La función objetivo toma valores discretos
  - Posibilidad de ruido en el conjunto de entrenamiento
  - Pueden no conocerse los valores de algunos atributos en los ejemplos del conjunto de entrenamiento
- Ejemplos:
  - Diagnóstico médico
  - Análisis de riesgo en la concesión de créditos
  - Elaboración de horarios

Cuestiones sobre árboles de decisión

- ¿Cuál es el árbol de decisión correcto?
  - Navaja de Occam
  - El mundo es inherentemente simple
  - El árbol de decisión más pequeño consistente con la muestra es el que tiene más probabilidades de identificar objetos desconocidos de manera correcta
  - Menor profundidad
  - Menor número de nodos
- ¿Cómo se puede construir el árbol de decisión más pequeño?
  - Búsqueda en escalada

Algoritmo básico (I)

- **Árbol inicial:** Árbol con un único nodo, sin etiquetar, al que asignamos como conjunto de ejemplos todo el conjunto de entrenamiento.
- Bucle principal:
  - Consideramos el primer nodo, N, sin etiquetar
    - Si los ejemplos asignados N tienen todos la misma clasificación, etiquetamos N con esa clasificación.
    - En otro caso ...

Algoritmo básico (II)

- Etiquetamos N con el *mejor* atributo A según el conjunto de ejemplos asignado.
- Para cada valor de A creamos una nueva arista descendente en el nodo N, y al final de cada una de esas nuevas aristas creamos un nuevo nodo sin etiquetar, $N_1, \ldots, N_k$.
- Separamos los ejemplos asignados al nodo N según el valor que tomen para el atributo A y creamos nuevos conjuntos de ejemplos para $N_1, \ldots, N_k$.

• **Hasta** que todos los nodos estés etiquetados

¿Qué atributo clasifica *mejor*?

Entropía (Definición)

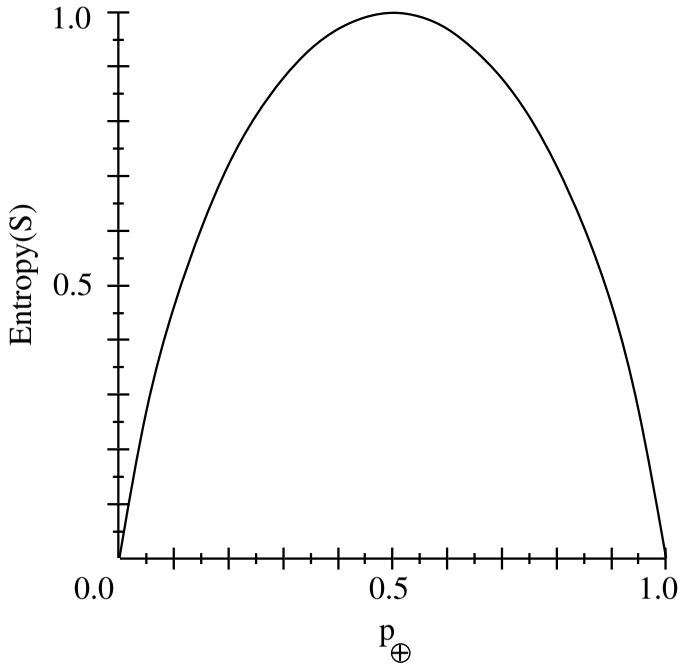
- **Definición:** Sea $P = \{p_1, p_2, \ldots, p_n\}$ una distribución de probabilidad. Llamamos **entropía** b-aria de la distribución P a

$$H_b(p_1, p_2, \ldots, p_n) = -\sum_{i=1}^{i=n} p_i \log_b p_i$$

- La función de entropía

$$H_2(p, 1-p) = p\, log_2 \frac{1}{p} + (1-p)\, log_2 \frac{1}{1-p}$$

es muy común y nos referimos a ella como *la* función de entropía y se denota por $H(p)$. Su gráfica es



Entropía

- D es un conjunto de entrenamiento
- P y N son, respectivamente, la cantidad de ejemplos positivos y negativos en D
- T es el número total de ejemplos de D
- La entropía de esta distribución es:
  - **Fórmula:** $\boldsymbol{I(P,N) = H_2(P/T, N/T) = \begin{cases} 0, & \text{si } N * P = 0; \\ -\frac{P}{T}\log_2\frac{P}{T} - \frac{N}{T}\log_2\frac{N}{T}, & \text{si } N * P \neq 0. \end{cases}}$
  - **Ejemplo:** $\boldsymbol{I(9,9) = -\frac{9}{18}\log_2\frac{9}{18} - \frac{9}{18}\log_2\frac{9}{18} = 1}$

(El término entropía *fue usado por primera vez por Clausius en 1864 e introducido en la teoría de la información por Shannon en 1948)*

Otro ejemplo

Día	Cielo	Temperatura	Humedad	Viento	Jugar_tenis
D1	Soleado	Alta	Alta	Débil	No
D2	Soleado	Alta	Alta	Fuerte	No
D3	Lluvioso	Alta	Alta	Débil	Sí
D4	Lluvioso	Media	Alta	Débil	Sí
D5	Lluvioso	Fría	Normal	Débil	Sí
D6	Lluvioso	Fría	Normal	Fuerte	No
D7	Lluvioso	Fría	Normal	Fuerte	Sí
D8	Soleado	Media	Alta	Débil	No
D9	Soleado	Fría	Normal	Débil	Sí
D10	Lluvioso	Media	Normal	Débil	Sí
D11	Soleado	Media	Normal	Fuerte	Sí
D12	Lluvioso	Media	Alta	Fuerte	Sí
D13	Lluvioso	Alta	Normal	Débil	Sí
D14	Lluvioso	Media	Alta	Fuerte	No

Entropia([9+,5-])= -(9/14)log_2(9/14)-(5/14)log_2(5/14)=0'940

Ganancia de información

Ganancia(D,A)= Estimación de la reducción de la entropía en el conjunto de entrenamiento D si tomamos el atributo A y clasificamos según sus valores

$$Ganancia(S, A) \equiv Entropia(S) - \sum_{v \in Valores(A)} \frac{|S_v|}{|S|} Entropia(S_v)$$

donde

- *Valores(A)* es el conjunto de valores del atributo A
- S_v es el subconjunto de S formado por aquellas instancias que en el atributo A toman el valor v

Otro ejemplo

Supongamos que S es un conjunto de entrenamiento con 14 ejemplos

- 9 ejemplos positivos y 5 negativos ([9+,5-])
- Unos de los atributos, $Viento$, puede tomar los valores *Débil* y *Fuerte*
- La distribución de ejemplos positivos y negativos según los valores de $Viento$ son

	Positivos	Negativos
Débil	6	2
Fuerte	3	3

Otro ejemplo

La *Ganancia de información* que obtenemos si clasificamos los 14 ejemplos según el atributo $Viento$ es:

Ganancia(S,A)

$$= \quad Entropia(S) - \sum_{v \in Valores(A)} \frac{|S_v|}{|S|} Entropia(S_v)$$

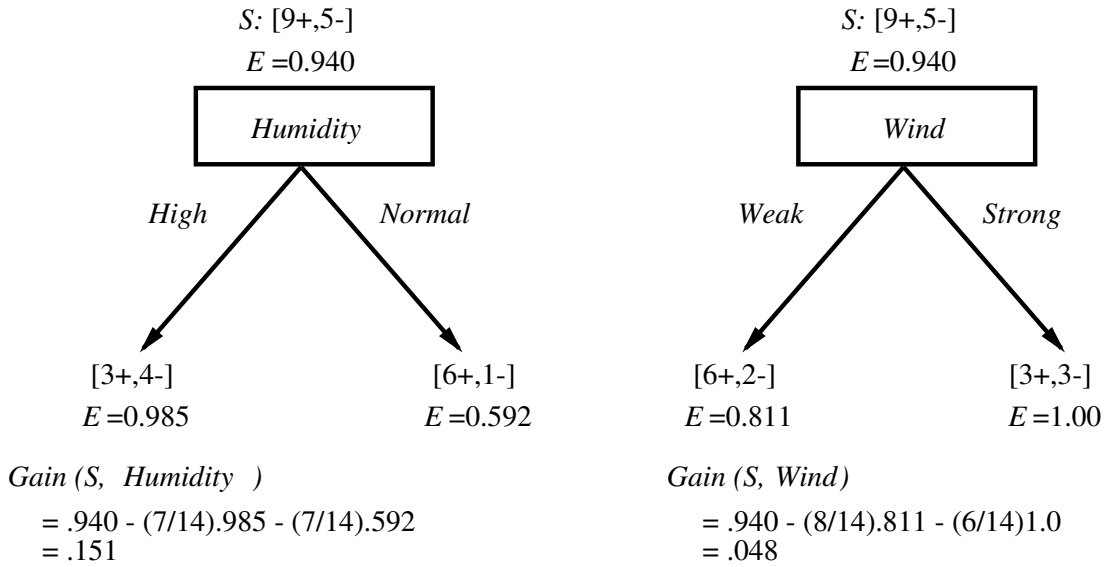
$$= \quad Entropia(S) - \frac{8}{14} Entropia(S_{Debil}) - \frac{6}{14} Entropia(S_{Fuerte})$$

$$= \quad 0'940 - \frac{8}{14} 0'811 - \frac{6}{14} 1'00$$

$$= \quad 0.048$$

Selección del mejor atributo (I)

Which attribute is the best classifier?



Gain (S, Humidity)
= .940 - (7/14).985 - (7/14).592
= .151

Gain (S, Wind)
= .940 - (8/14).811 - (6/14)1.0
= .048

Humedad proporciona mayor ganacia de información que *Viento*

Selección del mejor atributo (II)

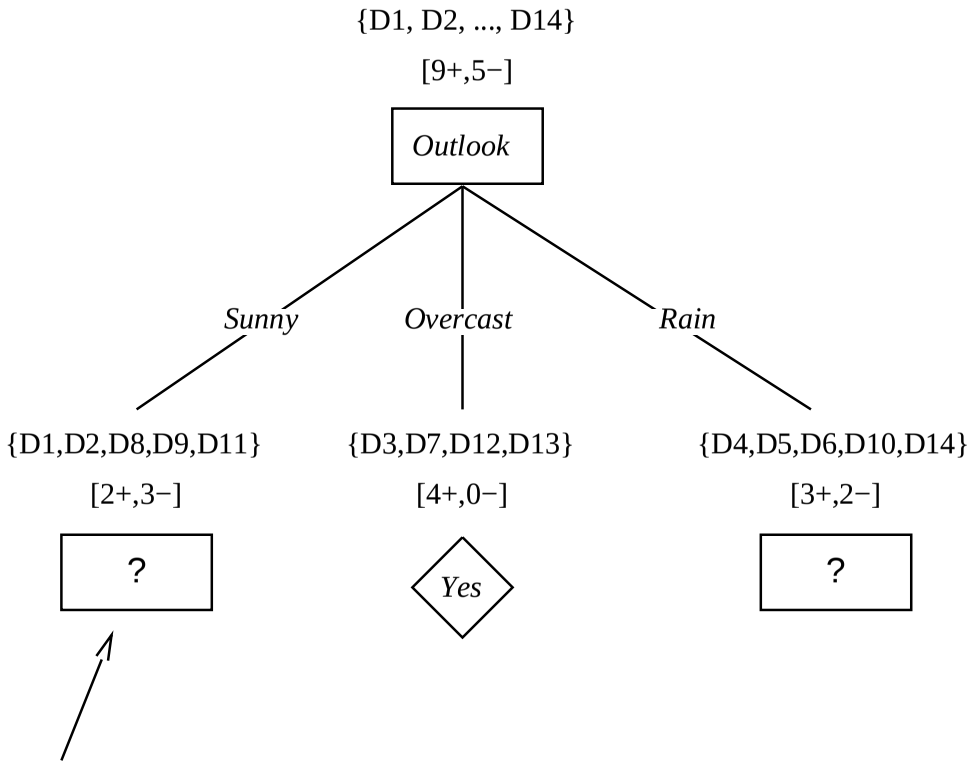
Consideremos el ejemplo anterior con 14 ejemplos

$$\begin{aligned} \text{Ganancia(S,Cielo)} &= 0'246 \\ \text{Ganancia(S,Humedad)} &= 0'151 \\ \text{Ganancia(S,Viento)} &= 0'048 \\ \text{Ganancia(S,Temperatura)} &= 0'029 \end{aligned}$$

- El atributo con el que tenemos mayor *Ganancia de información* es *Cielo*
- Luego, seleccionamos *Cielo* como atributo de decisión para el nodo raíz
- Para cada uno de sus posibles valores *(Soleado, Lluvioso, Lluvioso)* creamos una rama
- Clasificamos los ejemplos según los valores de *Cielo*
- En las ramas en las que los ejemplos no estén perfectamente clasificados, iteramos el proceso

Selección del mejor atributo (III)

Tras el primer paso, obtenemos un árbol de decisión *parcial*



Which attribute should be tested here?

S_{sunny} = {D1,D2,D8,D9,D11}

Gain (S_{sunny} , *Humidity)* = .970 − (3/5) 0.0 − (2/5) 0.0 = .970

Gain (S_{sunny} , *Temperature)* = .970 − (2/5) 0.0 − (2/5) 1.0 − (1/5) 0.0 = .570

Gain (S_{sunny} , *Wind)* = .970 − (2/5) 1.0 − (3/5) .918 = .019

Espacio de hipótesis

- El aprendizaje mediante árboles de decisión es un método de aproximación de una función objetivo de valores discretos en el cual la función objetivo es representada mediante un árbol de decisión.
- Podemos pensar este proceso como un proceso de *búsqueda* de un árbol que clasifique correctamente nuestros ejemplos.
- *Espacio de hipótesis:* Todos los posibles árboles de decisión.
- *Método:* Escalada (hill-climbing), empezando por el árbol vacío.
- *Función de evaluación que guía la búsqueda:* Ganancia de información

Comentarios (I)

- El espacio de hipótesis es *completo*: Cualquier función finita que tome valores discretos puede ser representada como un árbol de decisión.
- En cada elección consideramos una única hipótesis, a diferencia del algoritmo de ELIMINACIÓN DE CANDIDATOS, que considerábamos simultáneamente *todas* las hipótesis consistentes con los ejemplos.

Comentarios (II)

- No permite retroceso, por lo que podemos obtener óptimos locales en lugar de óptimos globales
- Considera en cada paso gran cantidad de ejemplos, en contraste con otros métodos, por ejemplo ELIMINACIÓN DE CANDIDATOS que consideran los ejemplos uno a uno. Esto supone mayor robustez frente al ruido.

Sesgo de restricción y sesgo preferencial (I)

Consideremos la búsqueda en el espacio de hipótesis en el algoritmo de árboles de decisión y el algoritmo de Eliminación de Candidatos

- **ID3** busca de manera *incompleta* en un espacio de búsqueda *completo*. El sesgo es consecuencia *únicamente* del método de búsqueda. El espacio de hipótesis no introduce ningún sesgo. Llamamos a este tipo de sesgo *preferencial*, ya que se debe a la preferencia de unas hipótesis sobre otras.

Sesgo de restricción y sesgo preferencial (II)

- El algoritmo de ELIMINACIÓN DE CANDIDATOS busca en un espacio de búsqueda *incompleto* de manera *completa* (i.e. devuelve *todas* las hipótesis del espacio consistentes con el conjunto de entrenamiento). En este caso el sesgo depende *únicamente* de la potencia expresiva en la representación de las hipótesis. Llamamos a este sesgo *sesgo de restricción* o *sesgo del lenguaje*.

Sistemas TDIDP

- **Los sistemas basados en árboles de decisión forman una familia llamada TDIDT (*Top–Down Induction of Decision Tree*)**
- **Representantes de TDIDT:**
  - **ID3 [Quinlan, 1986]**
  - **C4.5 [Quinlan, 93]**
- **Quinlan, J. R. *C4.5: Programs for Machine Learning* (Morgan Kaufmann, 1993)**

Otros temas de estudio (I)

- **Sobreajuste (overfitting)**
  - *Definición:* Dado un espacio de hipótesis H, una hipótesis $h \in H$ se dice que *sobreajusta* un conjunto de entrenamiento C si existe alguna hipótesis alternativa $h' \in H$ tal que h clasifica mejor que h' los elementos del conjunto de entrenamiento, pero h' clasifica mejor que h el conjunto completo de posibles instancias.
  - Exceso de ruido
  - Conjunto de entrenamiento demasiado pequeño
- **Poda (prunning)**
  - *Definición:* Podar un nodo de un árbol de decisión consiste en eliminar un subárbol anidado en ese nodo transformándolo en una hoja y asignándole la clasificación más común de los ejemplos de entrenamiento considerados en ese nodo

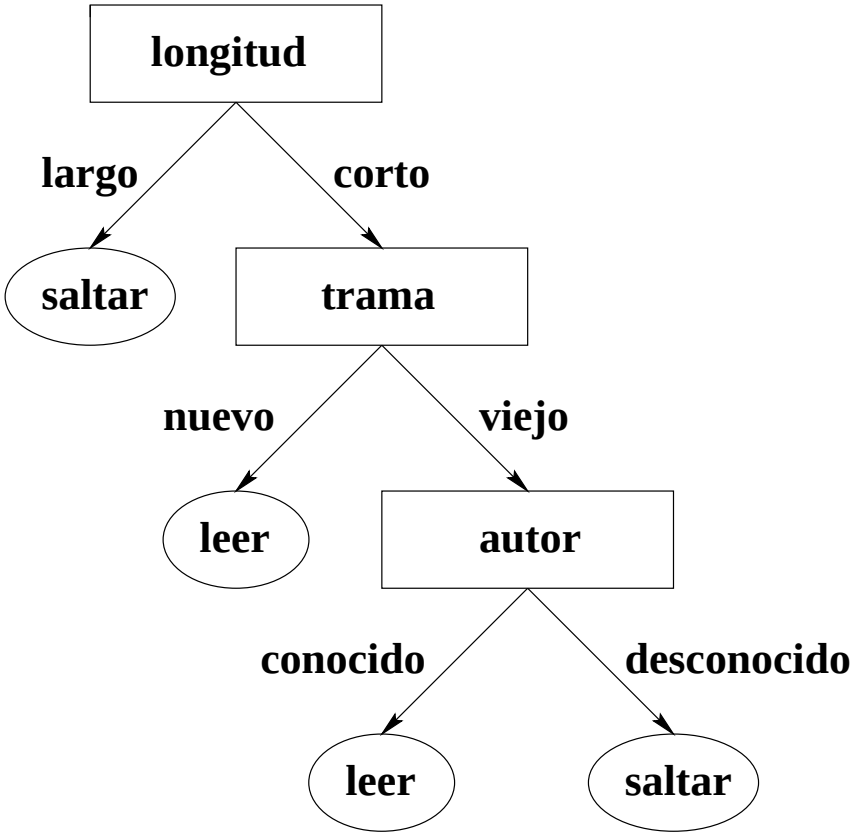
Otros temas de estudio (II)

- **Atributos de valores continuos**
- **Medidas alternativas en la selección de atributos**
- **Atributos con valores perdidos**
- **Atributos con pesos diferentes**

Ejemplo de datos

Ejemplo	Acción	Autor	Tema	Longitud	Sitio
e1	saltar	conocido	nuevo	largo	casa
e2	leer	desconocido	nuevo	corto	trabajo
e3	saltar	desconocido	viejo	largo	trabajo
e4	saltar	conocido	viejo	largo	casa
e5	leer	conocido	nuevo	corto	casa
e6	saltar	conocido	viejo	largo	trabajo
e7	saltar	desconocido	viejo	corto	trabajo
e8	leer	desconocido	nuevo	corto	trabajo
e9	saltar	conocido	viejo	largo	casa
e10	saltar	conocido	nuevo	largo	trabajo
e11	saltar	desconocido	viejo	corto	casa
e12	saltar	conocido	nuevo	largo	trabajo
e13	leer	conocido	viejo	corto	casa
e14	leer	conocido	nuevo	corto	trabajo
e15	leer	conocido	nuevo	corto	casa
e16	leer	conocido	viejo	corto	trabajo
e17	leer	conocido	nuevo	corto	casa
e18	leer	desconocido	nuevo	corto	trabajo

Arbol de decisión



Búsqueda del árbol de decisión

- **Información tras la división por un `Atributo`:**
  $I = \frac{N1*I1+N2*I2}{N1+N2}$, **donde**
  - **N1 = número de ejemplos en la clase 1**
  - **N2 = número de ejemplos en la clase 2**
  - **I1 = cantidad de información en los ejemplos de la clase 1**
  - **I2 = cantidad de información en los ejemplos de la clase 2**

Algoritmo ID3

- **Sesión**

```
?- ['aprende_ad.pl','aprende_ad_e1.pl'].
Yes

?- aprende_ad(accion,
                  [e1,e2,e3,e4,e5,e6,e7,e8,e9,
                   e10,e11,e12,e13,e14,e15,e16,e17,e18],
                  [autor,tema,longitud,sitio],
                  AD).
AD = si(longitud=largo, saltar,
            si(tema=nuevo, leer,
               si(autor=desconocido, saltar,
                  leer)))
Yes
```

Algoritmo ID3

- **Representación del problema** `aprende_ad_e1.pl`
  - `val(Objeto,Atributo,Valor)` **se verifica si el valor del** `Atributo` **del** `Objeto` **es** `Valor`

```
val(e1,accion,saltar).
val(e1,autor,conocido).
val(e1,tema,nuevo).
val(e1,longitud,largo).
val(e1,sitio,casa ).

.........................

val(e18,accion,leer).
val(e18,autor,desconocido).
val(e18,tema,nuevo).
val(e18,longitud,corto).
val(e18,sitio,trabajo).
```

Algoritmo ID3

- **Algoritmo de aprendizaje de árboles de decisión**
  - `aprende_ad(+Objetivo,+Ejemplos,+Atributos,-AD)` **se verifica si** `AD` **es el árbol de decisión inducido para el** `Objetivo` **a partir de la lista de** `Ejemplos` **y** `Atributos`

```
aprende_ad(Objetivo, Ejemplos, _ , Val) :-
   coinciden_todos_ejemplos(Objetivo, Ejemplos, Val).
aprende_ad(Objetivo, Ejemplos, Atributos, si(Atr=ValPos,APos,ANeg)) :-
   not(coinciden_todos_ejemplos(Objetivo, Ejemplos, _)),
   selecciona_division(Objetivo, Ejemplos, Atributos, Atr, Resto_Atr),
   divide(Ejemplos, Atr, ValPos, Positivos, Negativos),
   aprende_ad(Objetivo, Positivos, Resto_Atr, APos),
   aprende_ad(Objetivo, Negativos, Resto_Atr, ANeg).
```

Algoritmo ID3

- **Selección del mejor atributo para dividir**
  - `selecciona_division(+Objetivo,+Ejemplos,+Atributos, -Atrib,-Restantes_atrib)` **se verifica si** `Atributo` **es el mejor elemento de la lista de** `Atributos` **para determinar el** `Objetivo` **a partir de los** `Ejemplos` **(es decir, la información resultante del** `Objetivo` **en los** `Ejemplos` **usando como división el** `Atributo` **es mínima), y** `Restantes_atributos` **es la lista de los restantes** `Atributos`**. Falla si para ningún atributo se gana en información.**

```
selecciona_division(Objetivo, Ejemplos, [A|R], Atributo, Resto_Atr) :-
    informacion_division(Objetivo,Ejemplos,A,I),
    selecciona_max_ganancia_informacion(Objetivo,Ejemplos,R,A,I,
                                        Atributo,[],Resto_Atr).
```

Algoritmo ID3

- `informacion(+Objetivo,+Ejemplos,-I)` **se verifica si** `I` **es la cantidad de información en los** `Ejemplos` **respecto del** `Objetivo`**; es decir,**

$$\boldsymbol{I} = \begin{cases} \mathbf{0}, & \textbf{si } \boldsymbol{N} * \boldsymbol{P} = \mathbf{0}; \\ -\frac{\boldsymbol{P}}{\boldsymbol{T}} \log_2 \frac{\boldsymbol{P}}{\boldsymbol{T}} - \frac{\boldsymbol{N}}{\boldsymbol{T}} \log_2 \frac{\boldsymbol{N}}{\boldsymbol{T}}, & \textbf{si } \boldsymbol{N} * \boldsymbol{P} \neq \mathbf{0}. \end{cases}$$

```
informacion(Objetivo,Ejemplos,I) :-
   cuenta(Objetivo,_,Ejemplos,NP,NN),
   ( (NP=0 ; NN=0) -> I=0
     ;
     NT is NP + NN,
     I is - NP/NT * log2(NP/NT) - NN/NT * log2(NN/NT)).
```

Algoritmo ID3

- selecciona_max_ganancia_informacion(+Objetivo, +Ejemplos, +Atributos, +Mejor_atributo_actual, +Mejor_info_actual, -Atributo, +Atributos_analizados, -Resto_atributos) **se verifica si** Atributo **es el elemento de** Atributos **tal la información resultante del** Objetivo **en los** Ejemplos **usando como división el** Atributo **es mínima**

```
selecciona_max_ganancia_informacion(_,_,[],MejorA,_,
                                    MejorA, A_analizados, A_analizados).
selecciona_max_ganancia_informacion(Objetivo, Ejs, [A|R], MejorA, MejorI,
                                    Atributo, A_analizados, Resto_Atr) :-
   informacion_division(Objetivo,Ejs,A,Informacion),
   ( Informacion > MejorI
     -> selecciona_max_ganancia_informacion(
         Objetivo,Ejs,R,MejorA,MejorI,Atributo,[A|A_analizados],Resto_Atr)
     ;  selecciona_max_ganancia_informacion(
         Objetivo,Ejs,R,A,Informacion,Atributo,[MejorA|A_analizados],Resto_Atr)
   ).
```

Bibliografía

- Mitchell, T. M. *Machine Learning*
  - *Texto:* McGraw–Hill, 1997. Capítulo III.
  - *Web:*

    `http://www.cs.cmu.edu/~tom/mlbook.html`
- Poole D., et. al *Computational Intelligence. A Logical Approach* Oxford University Press, 1998.