Árboles de decisión

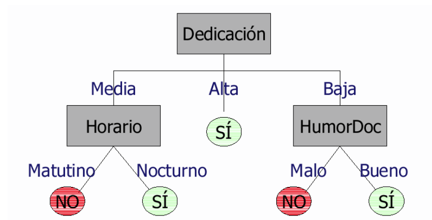
Introducción

- Cada rama del árbol es una restricción cobre los valores expresada como una conjunción
- Los árboles pueden verse como como las disyunciones de las restricciones representadas por sus ramas

Introducción

- ¿Que es un árbol de decisión?
  - Aproximan funciones discretas
- ¿Cómo?
  - Clasifican las instancias yendo de la raiz hacia las hojas
  - En cada nodo se prueba un atributo, y se baja por la rama asociada al valor de la instancia
  - El proceso se repite hasta llegar a una hoja

Introducción



(Dedicación=Media ∧ Horario=Nocturno)

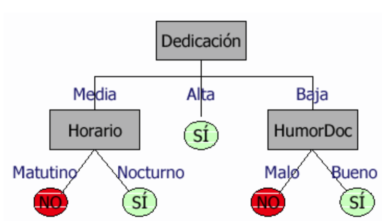
∨

(Dedicación=Alta)

∨

(Dedicación=Baja ∧ HumorDoc=Bueno)

Introducción



¿Cómo son clasificadas...?

<Ded=Media, Dif=Alta, Hor=Noc, Hum=Alta, Hdoc=Malo>

<Ded=Baja, Dif=Alta, Hor=Noc, Hum=Alta, Hdoc=Bueno>

Introducción

- **Problemas en los que se aplica A.D.:**
  - Las instancias se representan como parejas atributo-valor
  - La función objetivo toma valores discretos
  - Las descripciones requieren disyunciones
  - El conjunto de entrenamiento puede contener errores
  - Las instancias de entrenamiento pueden no tener todos los atributos
- **Algunas aplicaciones prácticas:**
  - Clasificación de rayos cósmicos
  - Detección de fraudes con tarjetas de crédito
  - Toma de decisiones médicas

Introducción

- La mayoría de los algoritmos para aprender un A.D. utilizan una técnica greedy
- En particular, vamos a ver el ID3:
  - Construye el árbol de manera top-down
  - En cada paso se pregunta: ¿cuál atributo se debe testear en cada nodo?
  - Por cada valor del atributo seleccionado, se genera una rama y se repite el proceso en cada una de ellas tomando sólo las instancias que tienen el valor del atributo correspondiente a la rama
  - Nunca se vuelve hacia atrás una decisión (i.e. nunca se verifica que el atributo seleccionado haya sido realmente el mejor)

Introducción

- Por ejemplo:

#Ej	Dedicación	Dificultad	Horario	Humedad	Humor Doc	Salva
1	Alta	Alta	Nocturno	Media	Bueno	SÍ
2	Baja	Media	Matutino	Alta	Malo	NO
3	Media	Alta	Nocturno	Media	Malo	SÍ
4	Media	Alta	Matutino	Alta	Bueno	NO

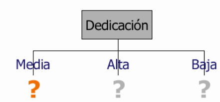
- Determinamos que el mejor atributo es Dedicación:

Introducción

- Crear una raíz
- Si todos los ej. son positivos → etiquetar con **+**
- Si todos los ej. son negativos → etiquetar con **-**
- Si no me quedan atributos → etiquetar con el valor más común
- En caso contrario:
  - **A** = atributo que mejor clasifica los ejemplos
  - Hago que la raíz pregunte por **A**
  - Para cada valor $\mathbf{v_i}$ de **A**
    - Genero una rama
    - Ejemplos$_{vi}$={Ejemplos en los cuales A= $\mathbf{v_i}$ }
    - Si Ejemplos$_{vi}$ es vacío → etiquetar con el valor más probable
    - En caso contrario → ID3(Ejemplos$_{vi}$, Atributos -{A})

Introducción

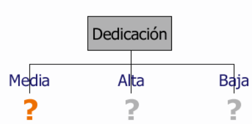
- Aplicamos ID3 a la primera rama :



#Ej	~~Dedicación~~	Dificultad	Horario	Humedad	Humor Doc	Salva
3	~~Media~~	Alta	Nocturno	Media	Malo	SÍ
4	~~Media~~	Alta	Matutino	Alta	Bueno	NO

Introducción

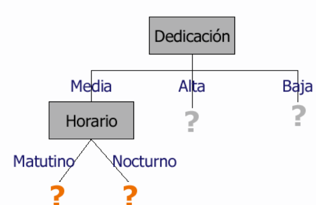
- Aplicamos ID3 a la primera rama :



#Ej	~~Dedicación~~	Dificultad	Horario	Humedad	Humor Doc	Salva
3	~~Media~~	Alta	Nocturno	Media	Malo	SÍ
4	~~Media~~	Alta	Matutino	Alta	Bueno	NO

Introducción

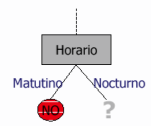
- Determinamos que el mejor atributo es Horario:



Introducción

- Aplicamos recursivamente ID3 en cada rama de horario:

#Ej	~~Dedicación~~	Dificultad	~~Horario~~	Humedad	Humor Doc	Salva
4	~~Media~~	Alta	~~Matutino~~	Alta	Bueno	NO



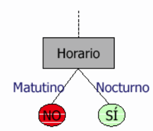
Selección del atributo

- Queda por determinar en el algoritmo anterior cómo seleccionamos el atributo
- Vamos a utilizar una medida de cuán bien un atributo separa los ejemplos.

Introducción

- Aplicamos recursivamente ID3 en cada rama de horario:

#Ej	~~Dedicación~~	Dificultad	~~Horario~~	Humedad	Humor Doc	Salva
3	~~Media~~	Alta	~~Nocturno~~	Media	Malo	SÍ



- ¿Qué pasaría si existieran exámenes vespertinos?

Selección del atributo

- Entropía:

  Si la función objetivo toma los valores $c_0 \ldots c_n$

  $$\textbf{Entropía(S)} = - \Sigma\, p_i \log p_i$$

  Donde p_i es la proporción de ejemplos $x \in S$ / $f(x)=c_i$

- En particular para funciones booleanas:

  $$\textbf{Entropía(S)} = - p_+ \log p_+ - p_- \log p_-$$

Selección del atributo

- Queda por determinar en el algoritmo anterior cómo seleccionamos el atributo
- Vamos a utilizar una medida de cuán bien un atributo separa los ejemplos.

Selección del atributo

- La entropía determina la cantidad mínima de bits que en promedio se requiere para codificar los elementos de S.
- Es una manera de medir la heterogeneidad de los datos.
- Algunos ejemplos:

  Entropía([9+, 5-]) = -(9/14) log(9/14) -(5/14) log(5/14)
  = 0.940

  Entropía([9+, 0-]) = -(9/9) log(9/9) = 0

  Entropía([9+, 9-]) = - 2* (9/18) log(9/18) = 1

Selección del atributo

- ¿Cómo utilizamos la entropía para elegir el atributo?
  - Definimos la **Ganancia de Información** de un atributo **A** sobre una muestra **G**:

$$\text{Ganancia}(S,A)=\text{Entropía}(S) - \sum_{v\in Val(A)}(|S_v|/|S|)\text{Entropía}(S_v)$$

  - Buscamos medir la reducción en la entropía al particionar por el atributo A
  - *Ganancia* es el número de bits que nos ahorramos, si sabemos el valor del atributo A.

Selección del atributo

- Sin embargo, *Horario* sí es una mejor elección.

$S_{Hor=Mat} \leftarrow [0+, 2-]$ $\quad$ $S_{Hor=Noct} \leftarrow [2+, 0-]$

$$\begin{aligned}\text{Gan}(S,Hor) &= 1 - (2/4)E(S_{H=M}) - (2/4)E(S_{H=N})\\ &= 1 - [0.5 * 1] - [0.5 * 1]\\ &= 1\end{aligned}$$

#Ej	Dedicación	Dificultad	Horario	Humedad	Humor Doc	Salva
1	Alta	Alta	Nocturno	Media	Bueno	SÍ
2	Baja	Media	Matutino	Alta	Malo	NO
3	Media	Alta	Nocturno	Media	Malo	SÍ
4	Media	Alta	Matutino	Alta	Bueno	NO

Selección del atributo

- En nuestro ejemplo, ¿no convendría particionar primero por ***HumorDoc***?

#Ej	Dedicación	Dificultad	Horario	Humedad	Humor Doc	Salva
1	Alta	Alta	Nocturno	Media	Bueno	SÍ
2	Baja	Media	Matutino	Alta	Malo	NO
3	Media	Alta	Nocturno	Media	Malo	SÍ
4	Media	Alta	Matutino	Alta	Bueno	NO

S = [2+,2-]

Entropía(S) = -(0.5)*log(0.5)-(0.5)*log(0.5) = 1

Selección del atributo

- ID3 realiza una búsqueda en el espacio de árboles, del más sencillo al más complejo.
- El espacio de búsqueda es **completo**: como toda función discreta se puede representar con un árbol, estamos seguros que el concepto objetivo está en el espacio de hipótesis.
- En todo momento, ID3 **mantiene una única hipótesis**: no hay forma de saber cuáles ni cuántos árboles equivalentes hay en el espacio.

Selección del atributo

#Ej	Dedicación	Dificultad	Horario	Humedad	Humor Doc	Salva
1	Alta	Alta	Nocturno	Media	Bueno	SÍ
2	Baja	Media	Matutino	Alta	Malo	NO
3	Media	Alta	Nocturno	Media	Malo	SÍ
4	Media	Alta	Matutino	Alta	Bueno	NO

$S_{Ded=Alta} \leftarrow [1+, 0-]$ $\quad$ $S_{Ded=Media} \leftarrow [1+, 1-]$ $\quad$ $S_{Ded=Baja} \leftarrow [0+, 1-]$

$$\begin{aligned}\text{Gan}(S,Ded) &= 1 - (1/4)E(S_{D=A}) - (2/4)E(S_{D=M}) - (1/4)E(S_{D=B})\\ &= 1 - [0.25 * 0] - [0.5 * 1] - [0.25 * 0] = 0.5\end{aligned}$$

$S_{HDoc=Bueno} \leftarrow [1+, 1-]$ $\quad$ $S_{HDoc=Malo} \leftarrow [1+, 1-]$

$$\begin{aligned}\text{Gan}(S,HDoc) &= 1 - (2/4)E(S_{HD=B}) - (2/4)E(S_{HD=M})\\ &= 1 - [0. 5 * 1] - [0.5 * 1] = 0\end{aligned}$$

- ¡*HDoc* **no** era una mejor elección!

Búsqueda con id3

- ID3 **no realiza backtracking**: una vez elegido un atributo, nunca se pregunta si esa fue la mejor opción. Esto puede conducir a una solución que es óptima local pero no globalmente
- El algoritmo **utiliza todos los ejemplos** en cada paso: esto evita ser muy sensible al ruido [como candidate-elimination].

Búsqueda con id3

- ¿Cuál es el sesgo inductivo del algoritmo ID3?
- Elige un árbol que:
  - cubre los ejemplos
  - es el más simple posible
  - los atributos con más ganancia de información, están más cerca de la raíz.
- A diferencia del candidate-elimination, el sesgo está en el algoritmo y no en el espacio.

Sesgo inductivo

- Sin embargo:
  - ¿Por qué no preferir hipótesis con **exactamente** 13 nodos?
  - ¡Dos algoritmos con distinta representación interna pueden llegar a árboles completamente distintos!
- Más adelante veremos una justificación bayesiana a este principio...

Sesgo inductivo

- Definiciones:
  - **Sesgo preferencial**: se maneja un espacio de hipótesis completo, pero el algoritmo prefiere ciertas hipótesis sobre otras. [**ID3**]
  - **Sesgo restrictivo**: se maneja un espacio de hipótesis incompleto. [**Candidate-elimination**]
- Por lo general, es deseable un sesgo preferencial ya que asegura que el concepto objetivo está en *H*.
- Un algoritmo puede tener un sesgo de ambos tipos.

Variaciones al ID3

- Algunas de las variaciones implementadas:
  - Evitar sobreajustes al conjunto de entrenamiento
  - Manejar atributos con valores continuos
  - Utilizar medidas alternativas para la elección del atributo
  - Manejar ejemplos de entrenamiento con atributos incompletos
  - Testeo de múltiples atributos en un nodo
  - Combinación lineal de atributos numéricos

Sesgo inductivo

- ¿Por qué preferir hipótesis más simples?

  Navaja de Occam: Prefiera la hipótesis más simple que se ajuste a los datos.
  William de Ockham (S.XIV)

  - Cómo hay [combinatoriamente] menos hipótesis simples, es menos probable que una de ellas se ajuste a los datos "porque sí".

Evitar sobre ajustes

- El sobreajuste se puede dar por:
  - **Error en los datos de entrenamiento**: el árbol clasifica correctamente el ejemplo erróneo, y pude clasificar erróneamente ejemplos del dominio.
  - **Cantidad insuficiente de ejemplos**: aparecen regularidades que no se cumplen en el dominio [en nuestro ejemplo: ¿es tan importante el horario en que se da un examen?]

Evitar sobre ajustes

- ¿Cómo evitarlo?
  - Detener el crecimiento del árbol antes que se ajuste perfectamente a los datos.
  - Luego de obtenido el árbol, aplicar técnicas de podado.
- ¿Cómo determinamos el tamaño óptimo del árbol?
  - Usando un conjunto de validación.
  - Aplicando técnicas estadísticas, para evaluar si es conveniente agregar/quitar nodos.
  - Usando una medida de complejidad para codificar los datos y el tamaño del árbol, y parando cuándo esta medida se minimiza.

Evitar sobre ajustes

- Ventajas de la poda de reglas
  - Como separamos las ramas, se puede cortar atributos "comunes" en algunas ramas y no en otras
  - Al convertir el árbol perdemos el orden en que aparecían los atributos: ¡es más fácil sacar conjunciones que quitar ramas!
  - Las reglas son legibles para un ser humano

Evitar sobre ajustes

- Poda por reducción de error
  - Consideramos cada nodo, y lo sustituimos por la valoración más común de los ejemplos que engloba.
  - Esto se realiza mientras mejoran los aciertos sobre el conjunto de validación.
  - La principal desventaja es que se precisa un conjunto relativamente grande de ejemplos.

Resumiendo

- Los A.D. permiten aprender funciones discretas
- La familia de algoritmos ID3 infiere los árboles seleccionando de forma greedy el atributo a testear en cada nodo
- El espacio considerado es completo: estoy seguro que la función que busco está en H.
- El sesgo inductivo del ID3 es: preferir árboles lo más sencillos posibles, estando más cerca de la raíz los atributos con más ganancia de información.
- Hay técnicas para evitar [en lo posible] el sobreajuste
- El algoritmo básico se puede extender para considerar atributos continuos, incompletos, etc.

Evitar sobre ajustes

- Poda de reglas [C4.5]
  - Transformamos el árbol en el conjunto de reglas que representa
  - Quitamos las condiciones de las reglas de forma que esta mejora su estimación en el conjunto de validación
  - Ordenamos las reglas de acuerdo a su porcentaje de acierto y, ante nuevas instancias, las aplicamos en ese orden