

# CI5438. Inteligencia Artificial II

## Clase 4: Aprendizaje en Árboles de Decisión

### Cap 18.3: RN

Ivette C. Martínez

Universidad Simón Bolívar

5 de octubre de 2009

# Árboles de Decisión

- Un árbol de decisión es un árbol de búsqueda que retorna una “decisión”
  - Entrada: Un objeto o situación descrito por un conjunto de atributos
  - Salida: Una Decisión.
- La entrada puede ser discreta o continua.
- La salida también puede ser discreta o continua
- Aprender una función con salida **discreta** se denomina aprendizaje de **clasificación**
- Aprender una función con salida **continua** se denomina **regresión**

# Árboles de Decisión

- Cada **nodo** en el árbol corresponde con una prueba de uno de los atributos
- Cada **enlace** entre los nodos representa uno de los valores que puede tomar el nodo padre
- Cada **hoja** en el árbol es el valor booleano retornado si la búsqueda llegó a esa hoja
- La meta es aprender la definición del **predicado meta**, donde esta definición se expresa como un árbol de decisión

# Árboles de Decisión

El proceso de construir un árbol de decisión es como sigue:

- Comenzar con el problema
- Determinar cuál es el predicado meta para el problema
- Construir una lista de atributos que describen el dominio del problema
- Eliminar los atributos irrelevantes
- Construir el árbol usando la lista filtrada de atributos

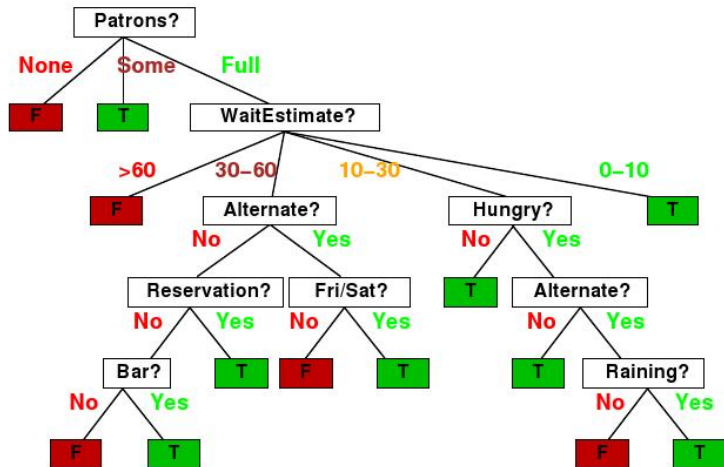
# Ejemplo de Árbol de Decisión

Attribute	Description
Alternate	Whether there is a suitable alternative restaurant nearby.
Bar	Whether the restaurant has a comfortable bar area to wait in.
Fri/Sat	True on Fridays and Saturdays.
Hungry	Whether the person is hungry.
Patrons	How many people are in the restaurant? (Values are None, Some, and Full)
Price	The restaurant's price range. (Values are \$, \$\$, \$\$\$)
Raining	Whether it is raining outside.
Reservation	Whether the person made a reservation.
Type	The kind of restaurant.
WaitEstimate	The wait estimated by the host. (Values are 0-10 minutes, 10-30, 30-60 and >60)

# Ejemplo de Árbol de Decisión

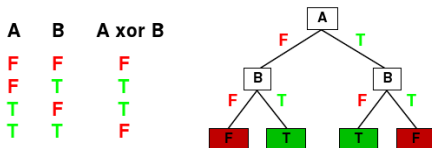
Example	Attributes										Target
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	<i>WillWait</i>
$X_1$	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>French</i>	<i>0-10</i>	<i>T</i>
$X_2$	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>30-60</i>	<i>F</i>
$X_3$	<i>F</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>Some</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Burger</i>	<i>0-10</i>	<i>T</i>
$X_4$	<i>T</i>	<i>F</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>10-30</i>	<i>T</i>
$X_5$	<i>T</i>	<i>F</i>	<i>T</i>	<i>F</i>	<i>Full</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>French</i>	<i>&gt;60</i>	<i>F</i>
$X_6$	<i>F</i>	<i>T</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$</i>	<i>T</i>	<i>T</i>	<i>Italian</i>	<i>0-10</i>	<i>T</i>
$X_7$	<i>F</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>None</i>	<i>\$</i>	<i>T</i>	<i>F</i>	<i>Burger</i>	<i>0-10</i>	<i>F</i>
$X_8$	<i>F</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$</i>	<i>T</i>	<i>T</i>	<i>Thai</i>	<i>0-10</i>	<i>T</i>
$X_9$	<i>F</i>	<i>T</i>	<i>T</i>	<i>F</i>	<i>Full</i>	<i>\$</i>	<i>T</i>	<i>F</i>	<i>Burger</i>	<i>&gt;60</i>	<i>F</i>
$X_{10}$	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>Italian</i>	<i>10-30</i>	<i>F</i>
$X_{11}$	<i>F</i>	<i>F</i>	<i>F</i>	<i>F</i>	<i>None</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>0-10</i>	<i>F</i>
$X_{12}$	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Burger</i>	<i>30-60</i>	<i>T</i>

# Ejemplo de Árbol de Decisión



# Expresividad

- Los árboles de decisión pueden expresar cualquier función de los atributos de entrada.  
Para funciones booleana, fila en la tabla de verdad  $\rightarrow$  camino a una hoja:



- Trivialmente, hay un árbol de decisión consistente para cualquier conjunto de entrenamiento, uno con un camino a una hoja para cada ejemplo (a menos que  $f$  sea no determinístico en  $x$ ), pero probablemente no generalice bien sobre nuevos ejemplos.
- Preferimos encontrar árboles de decisión más compactos.

- Cuántos árboles de decisión distintos pueden construirse con  $n$  atributos booleanos?
  - = el número de funciones booleanas
  - = el número de tablas de verdad con  $2^n$  filas. =  $2^{2^n}$
- Cuántas hipótesis conjuntivas puras?  
Ejm: (*Hungry*  $\wedge$   $\neg$ *Rain*)
- Espacios de Hipótesis más expresivos:
  - Incrementan el chance de que la función objetivo pueda ser expresada
  - Incrementan el número de hipótesis consistentes con el conjunto de entrenamiento
    - $\Rightarrow$  Pueden obtener peores predicciones.

# Induciendo árboles de decisión

- Objetivo: Encontrar el árbol más pequeño que sea consistente con los ejemplos de entrenamiento.
- Un ejemplo consiste en un vector de atributos de entrada  $X$ , y un valor booleano de salida  $y$ .
- **Clasificación** - Valor del predicado meta
- **Ejemplo Positivo** - El predicado meta es True
- **Ejemplo Negativo** - El predicado meta es False
- **Conjunto de Entrenamiento** - Un conjunto completo de ejemplos positivos y negativos

# Aprendizaje de árboles de decisión

Idea: Escoger recursivamente el atributo “mas significativo” como raíz del (sub)árbol.

```
function DTL(examples, attributes, default) returns a decision tree
  if examples is empty then return default
  else if all examples have the same classification then return the classification
  else if attributes is empty then return MODE(examples)
  else
    best ← CHOOSE-ATTRIBUTE(attributes, examples)
    tree ← a new decision tree with root test best
    for each value  $v_i$  of best do
      examplesi ← {elements of examples with best =  $v_i$ }
      subtree ← DTL(examplesi, attributes - best, MODE(examples))
      add a branch to tree with label  $v_i$  and subtree subtree
  return tree
```

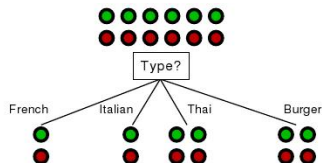
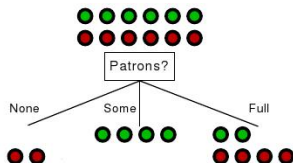
# Conjunto de Entrenamiento

Ejemplos descritos como valores de atributos (booleanos, discretos, continuos, etc.)

Example	Attributes										Target
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	<i>WillWait</i>
$X_1$	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>French</i>	<i>0-10</i>	<i>T</i>
$X_2$	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>30-60</i>	<i>F</i>
$X_3$	<i>F</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>Some</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Burger</i>	<i>0-10</i>	<i>T</i>
$X_4$	<i>T</i>	<i>F</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>10-30</i>	<i>T</i>
$X_5$	<i>T</i>	<i>F</i>	<i>T</i>	<i>F</i>	<i>Full</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>French</i>	<i>&gt;60</i>	<i>F</i>
$X_6$	<i>F</i>	<i>T</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$</i>	<i>T</i>	<i>T</i>	<i>Italian</i>	<i>0-10</i>	<i>T</i>
$X_7$	<i>F</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>None</i>	<i>\$</i>	<i>T</i>	<i>F</i>	<i>Burger</i>	<i>0-10</i>	<i>F</i>
$X_8$	<i>F</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$</i>	<i>T</i>	<i>T</i>	<i>Thai</i>	<i>0-10</i>	<i>T</i>
$X_9$	<i>F</i>	<i>T</i>	<i>T</i>	<i>F</i>	<i>Full</i>	<i>\$</i>	<i>T</i>	<i>F</i>	<i>Burger</i>	<i>&gt;60</i>	<i>F</i>
$X_{10}$	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>Italian</i>	<i>10-30</i>	<i>F</i>
$X_{11}$	<i>F</i>	<i>F</i>	<i>F</i>	<i>F</i>	<i>None</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>0-10</i>	<i>F</i>
$X_{12}$	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Burger</i>	<i>30-60</i>	<i>T</i>

# Escogiendo un atributo

Idea: Un buen atributo separa los ejemplos que son (idealmente) “todos positivos” o “todos negativos”



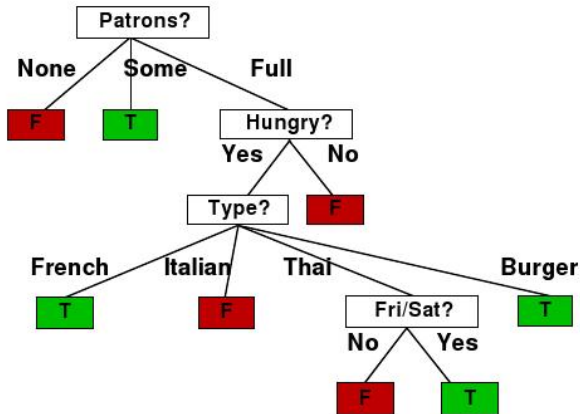
- La información responde a una pregunta.
- Mientras menos idea tengo sobre la respuesta inicialmente, hay más información contenida en la respuesta.
- Escala: 1 bit = respuesta a una pregunta booleana con previo  $\langle 0,5, 0,5 \rangle$
- La información es una respuesta cuando el previo es  $\langle P_1, \dots, P_n \rangle$  es
$$E(\langle P_1, \dots, P_n \rangle) = \sum -P_i \text{Log}_2(P_i)$$
También se denomina **entropía** del previo.

- Supongamos que tenemos  $p$  ejemplos positivos y  $n$  negativos en la raíz.
  - Se necesitan  $E(\langle p/(p+n), n/(p+n) \rangle)$  bits para clasificar un nuevo ejemplo
  - En el ejemplo del restaurant  $p = n = 6$ , luego necesitamos 1 bit
- Un atributo separa los ejemplos  $E$  en subconjuntos  $E_i$ , cada uno de los cuales (esperamos) necesite menos información para completar la clasificación.

- Sea  $E_i$  con  $p_i$  ejemplos positivos y  $n_i$  negativos
  - Se necesitan  $E(< p_i/(p_i + n_i), n_i/(p_i + n_i) >)$  bits para clasificar un nuevo ejemplo.
  - El número **esperado** de bits para cada ejemplo sobre todas las ramificaciones es:
$$\sum \frac{p_i+n_i}{p+n} E(< p_i/(p_i + n_i), n/(p_i + n_i) >)$$
- Para *patrons?* es 0,459 bits y para *Type* es 1
- $\Rightarrow$  Escoger el atributo que minimize la información que será requerida.

# Ejemplo

Árbol de decisión aprendido a partir de los 12 ejemplos.

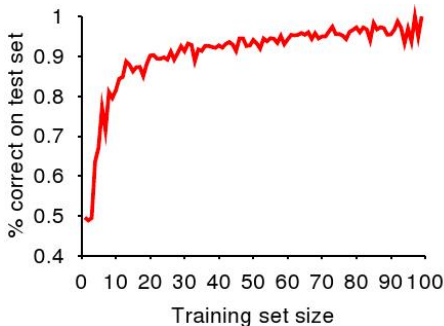


# Desempeño del algoritmo de Aprendizaje

Cómo sabemos si  $h \approx f$ ?

- 1 Usar los teoremas de la teoría de aprendizaje estadístico/computacional
- 2 Probar  $h$  en un nuevo conjunto de ejemplos de prueba.

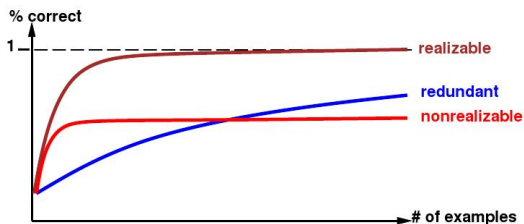
Curva de aprendizaje = % de aciertos en el conjunto de prueba como función del tamaño del conjunto de entrenamiento



# Desempeño del algoritmo de Aprendizizaje

Las curvas de aprendizaje dependen de

- realizable vs. no-realizable
  - Lo no-realizable puede deberse a atributos faltantes o a una clase de hipótesis restringida
- Expsividad Redundante (i.e., atributos irrelevantes)



# Desempeño del algoritmo de Aprendizaje

- Un algoritmo de aprendizaje es bueno si produce hipótesis que predicen adecuadamente la clasificación de ejemplos desconocidos.
- Una metodología para verificar la calidad de la predicción:
  - 1 Coleccionar un gran conjunto de ejemplos
  - 2 Separarlo en dos conjuntos disjuntos: el conjunto de entrenamiento y el conjunto de prueba
  - 3 Usar en algoritmo en es conjunto de entrenamiento para generar la hipótesis  $h$
  - 4 Medir el porcentaje de los ejemplos del conjunto de pruebas que son clasificados correctamente por  $h$
  - 5 Repetir los pasos 1 al 4 variando el tamaño del conjunto de entrenamiento y seleccionando aleatoriamente los elementos para cada tamaño