Tema 6: Introducción al Aprendizaje Automático

Objetivos

- Conocer el concepto de aprendizaje automático.
- Conocer los fundamentos, necesidad y utilidad de agentes capaces de aprender.
- Entender las técnicas básicas de aprendizaje automático. Conocer las técnicas necesarias para el aprendizaje de árboles de decisión. Saber resolver problemas en los que sea necesaria la aplicación de esta técnica.

Estudia este tema en …

- S. Russell, P. Norvig, Artificial Intelligence: A modern Approach, Tercera Edición, Ed. Pearson, 2010.

Contenido

- Distintos tipos de aprendizaje
- Modelos inductivos sobre árboles de decisión

Aprendizaje Automático (Machine Learning)

El aprendizaje es una capacidad fundamental de la inteligencia humana, que nos permite:

- Adaptarnos a cambios de nuestro entorno.
- Desarrollar una gran variedad de habilidades.
- Adquirir experiencia en nuevos dominios.

¿Para qué?

- El aprendizaje automático cubre una amplia gama de fenómenos como:
  - El perfeccionamiento de la habilidad.
  - La adquisición del conocimiento.
- El aprendizaje es esencial en entornos desconocidos.
- Programa de IA (Búsqueda, SBC, Planificación, ...):
  - Su límite está en el conocimiento que se les ha proporcionado.
  - No resuelven problemas mas allá de esos límites.
- El aprendizaje modifica el mecanismo de decisión del agente para mejorar su comportamiento.
- Aprendizaje automático: programas que mejoran su comportamiento con la experiencia.

¿Qué aprender?

Does Machine Learning Really Work?
Tom Mitchell. AI Magazine 1997

- ¿Donde y para que se puede usar el aprendizaje automático?
  - Tareas difíciles de programar (reconocimiento de caras, voz, ...)
  - Aplicaciones auto adaptables (interfaces inteligentes, spam killers, sistemas recomendadores, ...)
  - Minería de datos (análisis de datos inteligente)

Definición

- Un programa de ordenador se dice que aprende de la experiencia E con respecto a alguna clase de tareas T y a alguna medida de comportamiento P, si su comportamiento en tareas de T, medido a través de P, mejora con la experiencia E.

Ejemplos

- Aprendizaje de damas:
  - T: jugar a las damas.
  - P: % de juegos ganados.
  - E: partidas jugadas contra una copia de si mismo.

Ejemplos

- Aprendizaje de reconocimiento de caracteres de escritos a mano:
  - T: reconocer y clasificar palabras escritas a mano a través de imágenes.
  - P: % de palabras correctamente clasificadas.
  - E: una base de datos de palabras escritas a mano con su correspondiente clasificación.

Ejemplos

- Aprendizaje de un sistema para conducir:
  - T: conducir por una carretera usando sensores de visión.
  - P: distancia promedio antes de que se produzca un error (juzgado por un humano).
  - E: secuencia de imágenes y comandos de conducción registrados en la observación de un conductor humano.

Estrategias de aprendizaje

- Aprendizaje memorístico
- Aprendizaje a través de consejos
- Aprendizaje en la resolución de problemas
- Aprendizaje a partir de ejemplos: inducción
- Aprendizaje basado en explicaciones
- Aprendizaje a través de descubrimiento
- Aprendizaje por analogía

Aprendizaje inductivo

Aprende a partir de ejemplos

- El objetivo es aprender la función f.
- Un ejemplo es un par (x,f(x)).

Problema: encontrar una hipótesis h

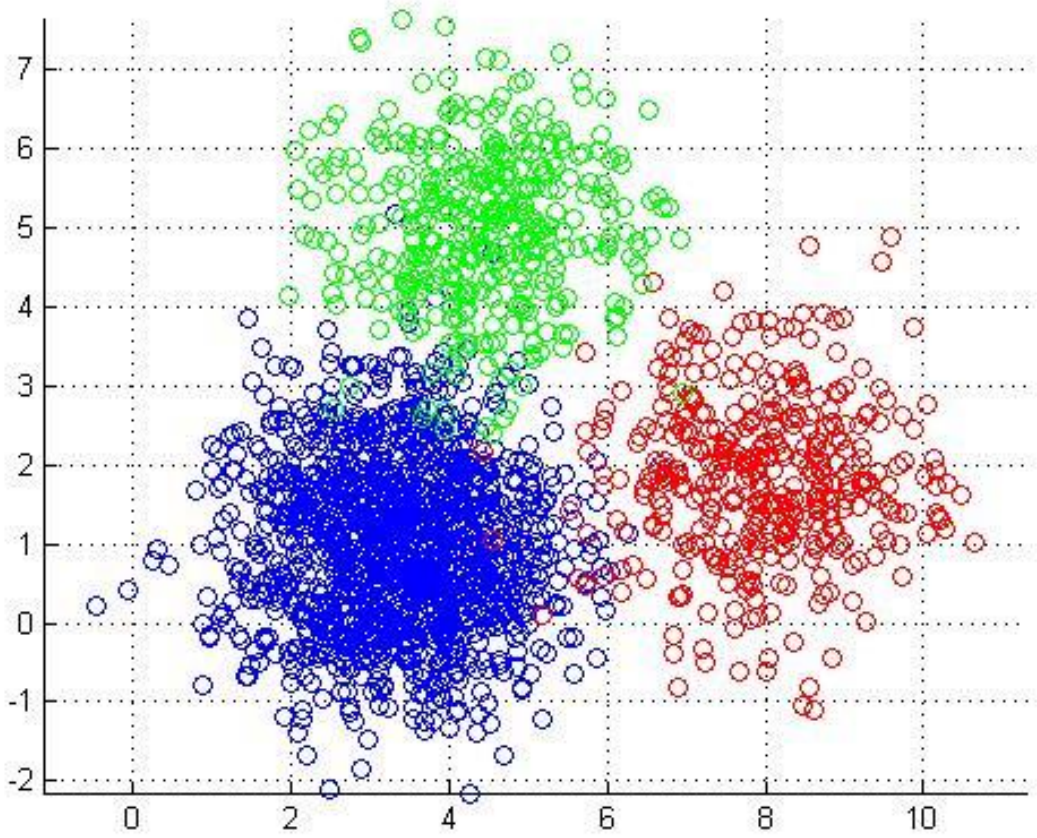
tal que h=f

sobre los conjuntos del ejemplos de entrenamiento.

Tipos de aprendizaje

- Uno de los puntos clave para el aprendizaje es el tipo de realimentación disponible en el proceso:
  - Aprendizaje supervisado: Aprender una función a partir de ejemplos de sus entradas y salidas.
  - Aprendizaje no supervisado: Aprender a partir de patrones de entradas para los que no se especifican los valores de su salidas.

Aprendizaje no supervisado



Aprendizaje supervisado

- Métodos basados en modelos: representan el conocimiento aprendido en algún lenguaje de representación (modelo o hipótesis).

- Métodos basados en instancias: representan el conocimiento aprendido como un conjunto de prototipos descritos en el mismo lenguaje usado para representar la evidencia.

Ejemplo

- Supermercado: se desea clasificar los clientes entre buenos y malos clientes
- Base de datos: información acerca de los clientes y forma de pago de los mismos.

Id	Casado	N-hijos	Sexo	Pago	Buen-cliente
1	si	3	m	Tarjeta	sí
2	no	0	h	Tarjeta	sí
3	no	1	m	Efectivo	no
4	si	4	m	Crédito	sí
5	si	2	h	Efectivo	no
6	no	1	m	Tarjeta	no
7	no	0	h	Efectivo	sí
8	no	0	h	Crédito	sí
9	no	1	h	Tarjeta	no
10	si	4	m	Crédito	sí

SI N-hijos > 2 ENTONCES Buen-cliente=si

Si Casado=no Y sexo=h Y N-hijos=0 ENTONCES Buen-cliente=si

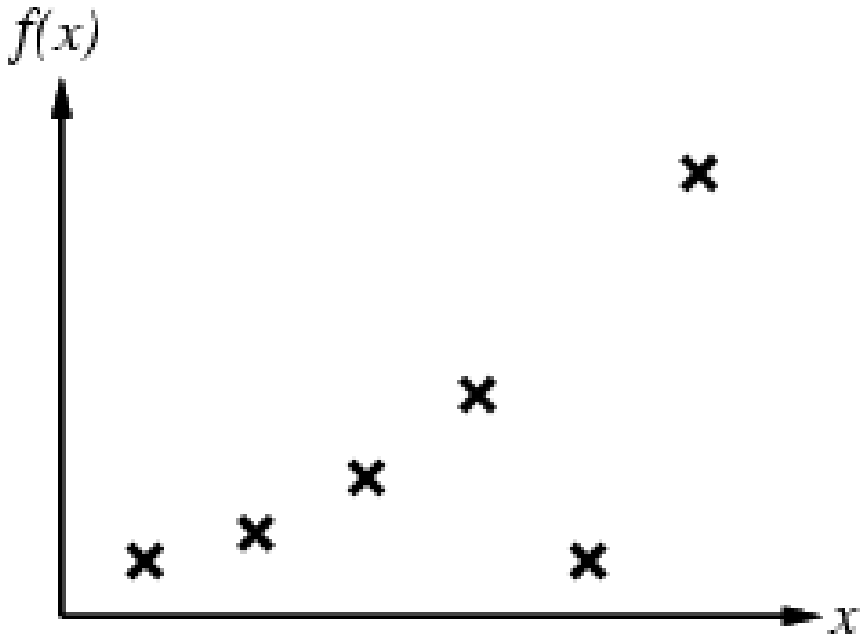
Ejemplo

- Ejemplo: modelado de la probabilidad de fallo de una máquina.
- Clase: la máquina fallará / la máquina no fallará.
- Atributos: conjunto de medidas:
  - Temperatura.
  - Nivel de vibraciones.
  - Horas de funcionamiento.
  - Meses desde la última revisión.
- Instancias: ejemplos pasados (situaciones conocidas).
- Hipótesis: relación entre las medidas y la probabilidad de fallo:

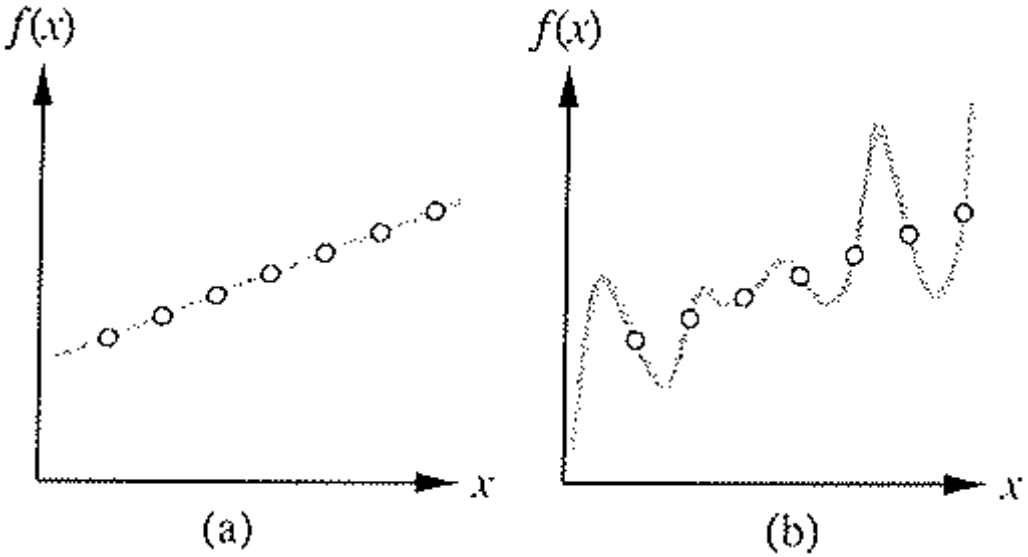
SI nivel_vibraciones = alto Y temperatura = alta ENTONCES fallará.

Aprendizaje inductivo

- Una hipótesis estará bien generalizada si puede predecir ejemplos que no se conocen.



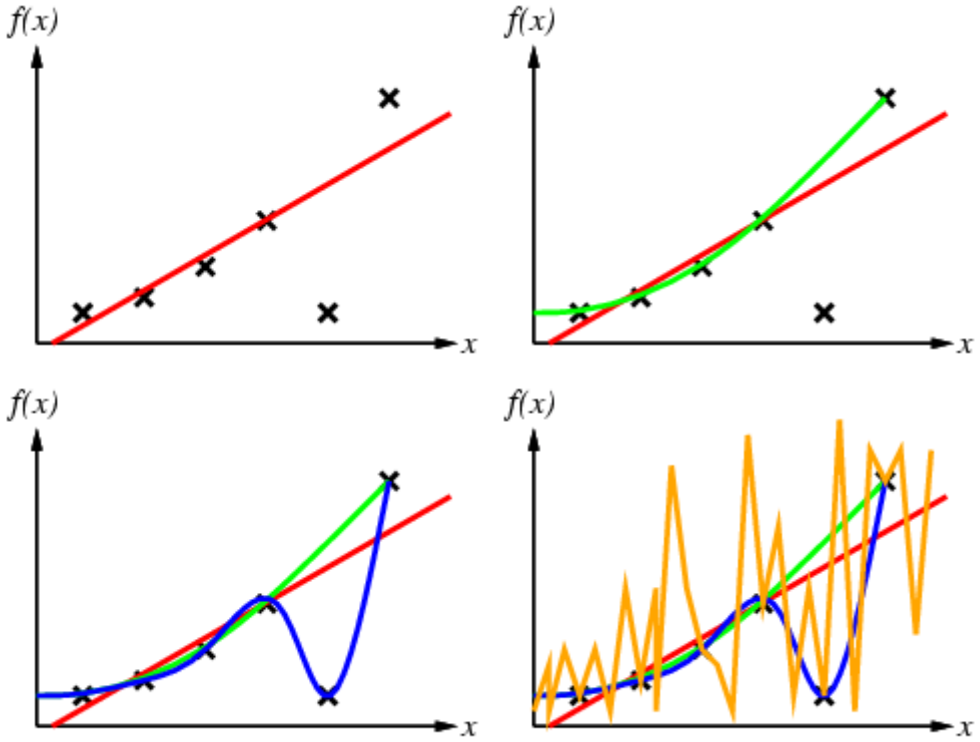
¿Cómo elegir entre múltiples hipótesis consistentes?



La hipótesis se dice **consistente** ya que satisface a todos los datos

Navaja de Ockham: elegir la hipótesis más simple consistente con los datos

¿Cómo elegir entre múltiples hipótesis consistentes?



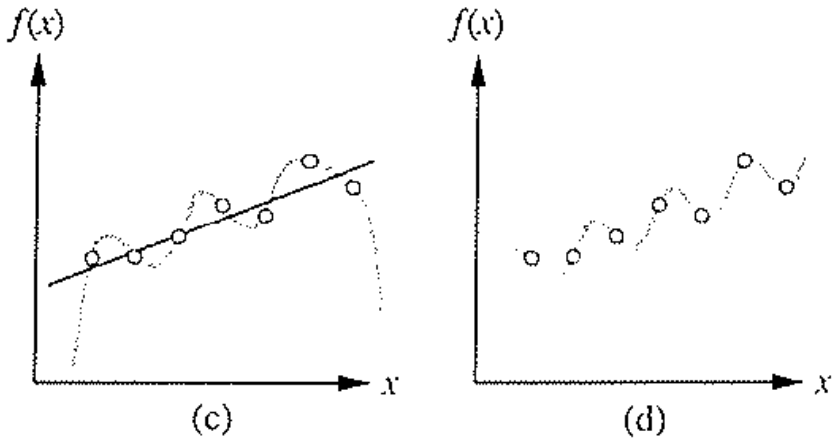
Espacio de hipótesis

- Las hipótesis se pueden expresar de diversas formas:
  - Árboles de decisión.
  - Reglas.
  - Redes neuronales.
  - Modelos bayesianos o probabilísticos.
  - etc.
- Los árboles de decisión y las reglas son algunos de los modelos más usados en aprendizaje automático.

Problema de aprendizaje realizable

- Se dice que un problema de aprendizaje es realizable si el espacio de hipótesis contiene a la función verdadera

f(x)=ax+b+csen(x)

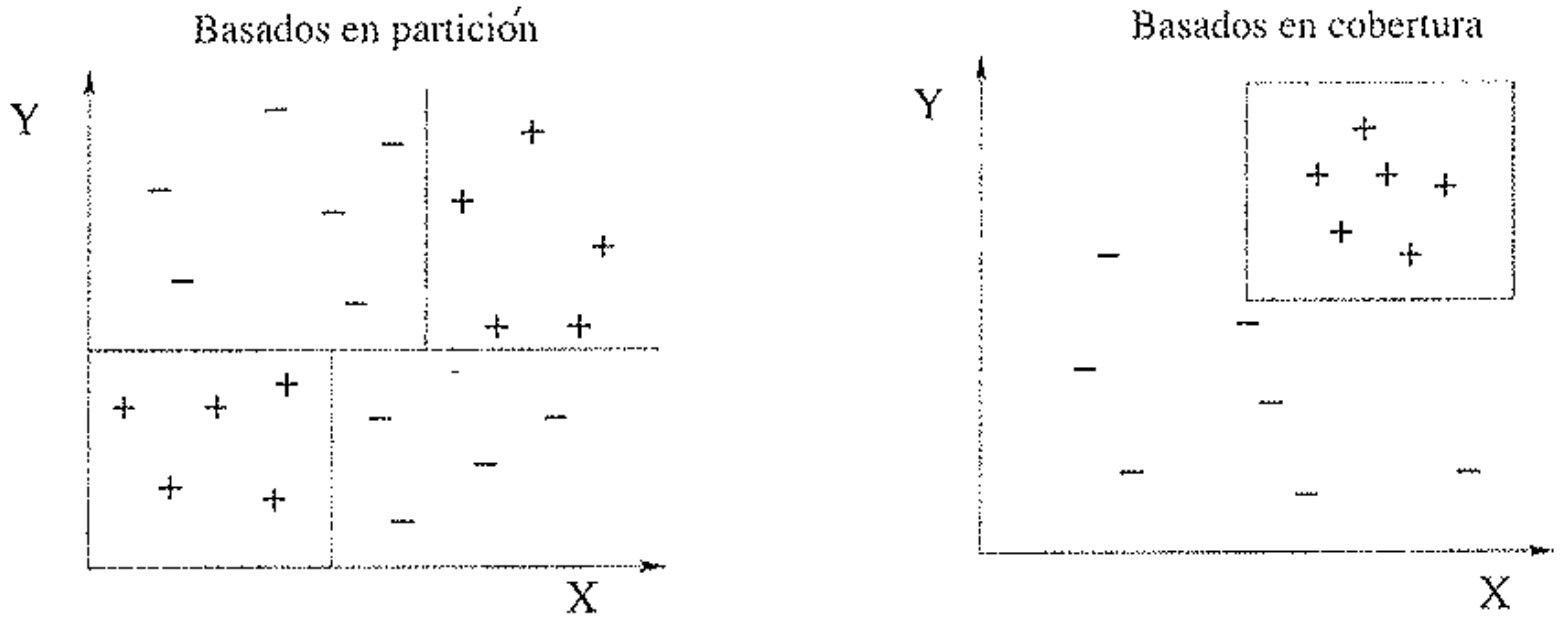


Algoritmos más ampliamente utilizados

- Algoritmos basados en el “divide y vencerás” (splitting): consisten en ir partiendo sucesivamente los datos en función del valor de un atributo seleccionado cada vez (aprendizaje de árboles de decisión).

- Algoritmos basados en el “separa y vencerás” (covering): consisten en encontrar condiciones de las reglas que cubran la mayor cantidad de ejemplos de una clase y la menor en el resto de la clase (aprendizaje de reglas).

Clasificación basada en partición versus cobertura



Aprender árboles de decisión

La inducción de árboles de decisión es uno de los métodos más sencillos y con más éxito para construir algoritmos de aprendizaje

- Árboles de decisión como herramienta de desarrollo.
- Expresividad de los árboles de decisión.
- Inducción de árboles de decisión.

Árboles de decisión como herramienta de desarrollo

- Un árbol de decisión toma como entrada un objeto o una situación descrita a través de un conjunto de atributos y devuelve una "decisión", el valor previsto de la salida dada la entrada.

- Atributos: discreto o continuos.
- Salida:
  - Discreta: clasificación.
  - Continua: regresión.

Ejemplo

Problema: decidir si se debe de esperar por una mesa en un restaurante utilizando los siguientes atributos:

1. Alternate: is there an alternative restaurant nearby?
2. Bar: is there a comfortable bar area to wait in?
3. Fri/Sat: is today Friday or Saturday?
4. Hungry: are we hungry?
5. Patrons: number of people in the restaurant (None, Some, Full)
6. Price: price range ($, $$, $$$)
7. Raining: is it raining outside?
8. Reservation: have we made a reservation?
9. Type: kind of restaurant (French, Italian, Thai, Burger)
10. WaitEstimate: estimated waiting time (0-10, 10-30, 30-60, >60)

Ejemplo

- Los ejemplos se describen mediante los valores de los atributos (Booleanos, discretos, continuos)

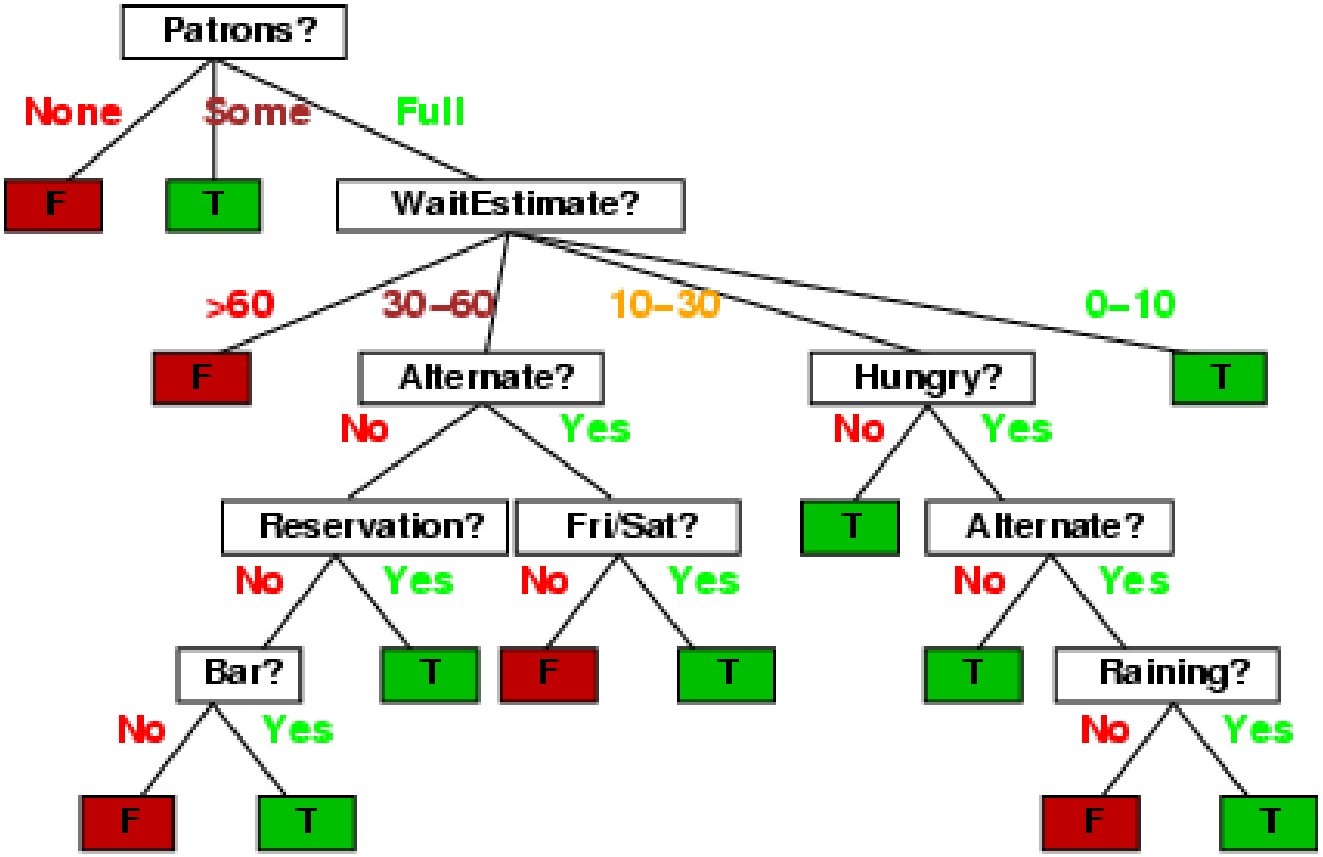
Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

- La clasificación de los ejemplos es positiva (T) o negativa (F).

Ejemplos positivos y negativos

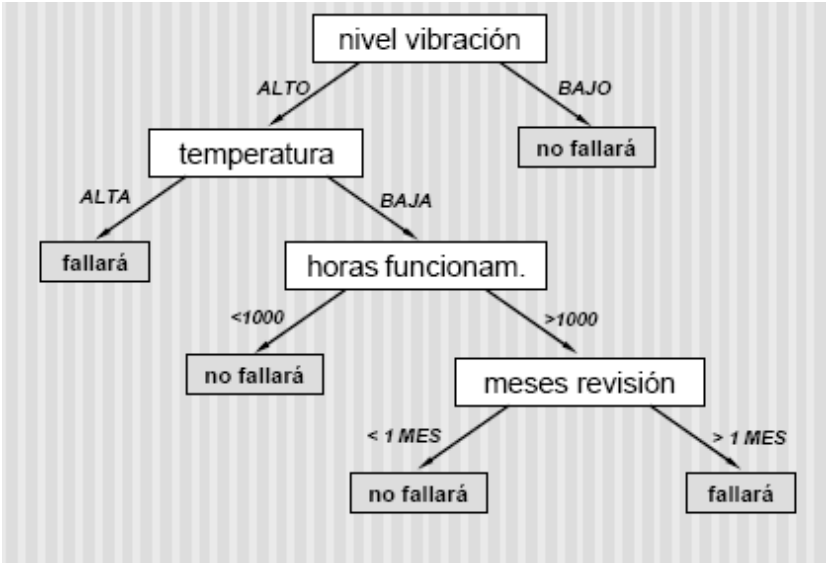
- Los ejemplos positivos son aquellos en los que la meta esperar es verdadera (X_1, X_3,…).
- Los ejemplos negativos son aquellos en los que es falsa (X_2, X_5,…).
- El conjunto de ejemplos completo se denomina **conjunto de entrenamiento**.

Una posible representación para la hipótesis



Ejemplo

Modelado de la probabilidad de fallo de una máquina



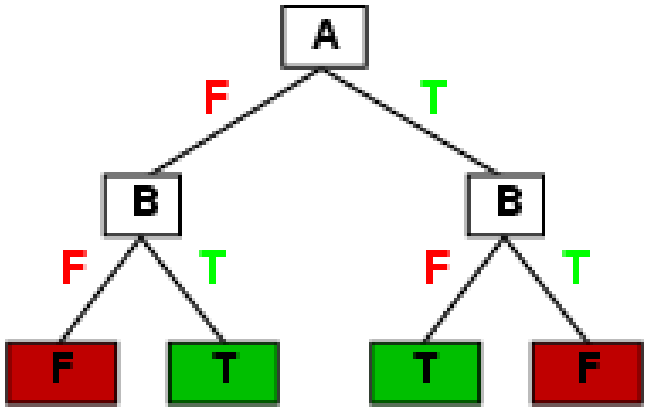
Ejemplo

Temperatura	Nivel de vibraciones	Horas de funcionamiento	Meses desde revisión	Probabilidad de fallo
ALTA	ALTO	< 1000	> 1 MES	fallará
BAJA	BAJO	< 1000	< 1 MES	no fallará
ALTA	BAJO	>1000	> 1 MES	no fallará
ALTA	BAJO	< 1000	> 1 MES	no fallará
BAJA	ALTO	< 1000	> 1 MES	no fallará
BAJA	ALTO	>1000	> 1 MES	fallará
ALTA	ALTO	< 1000	< 1 MES	fallará

Expresividad de los árboles de decisión

- Los árboles de decisión pueden expresar cualquier función a partir de los atributos de entrada.
- Por ejemplo, para funciones Booleanas, cada fila de la tabla de verdad se traslada a un camino del árbol:

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



- De forma trivial, hay un árbol de decisión consistente para cualquier conjunto de entrenamiento con un camino asociado a cada ejemplo, pero seguramente no será bueno para generalizar nuevos ejemplos.

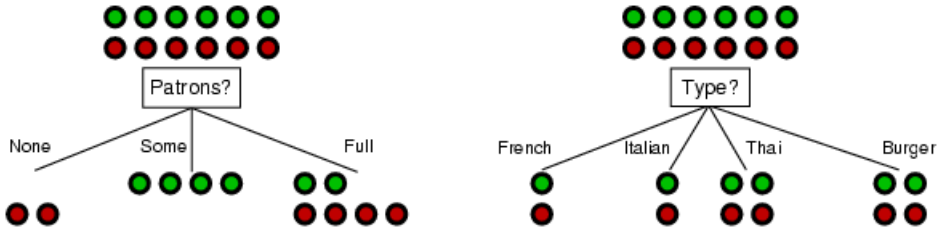
- Es preferible encontrar árboles de decisión más compactos.

Inducción de árboles de decisión

- Múltiples formas de inferir el árbol:
  - Trivial: se crea una ruta del árbol por cada instancia de entrenamiento.
    - Árboles excesivamente grandes.
    - No funcionan bien con instancias nuevas.
  - Optimo: el árbol más pequeño posible compatible con todas las instancias (navaja de Ockham).
    - Inviable computacionalmente.
  - Pseudo-optimo (heurístico): selección del atributo en cada nivel del árbol en función de la calidad de la división que produce.
    - Los principales programas de generación de árboles utilizan procedimientos similares (ID3, C4.5, CART, etc).

Elección de atributos

Idea: un buen atributo debería dividir el conjunto de ejemplos en subconjuntos que sean o “todos positivos” o “todos negativos”.



Patrons=Clientes es una buena elección

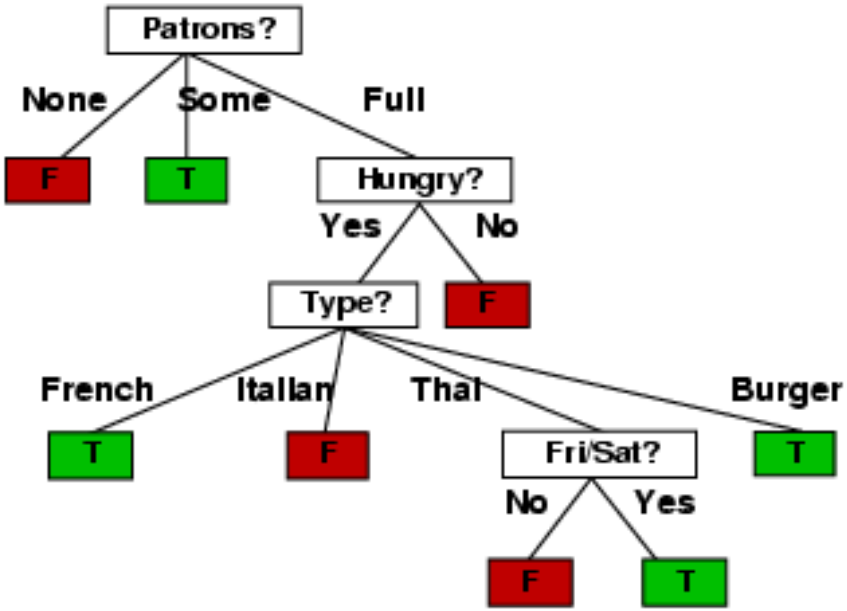
DTL

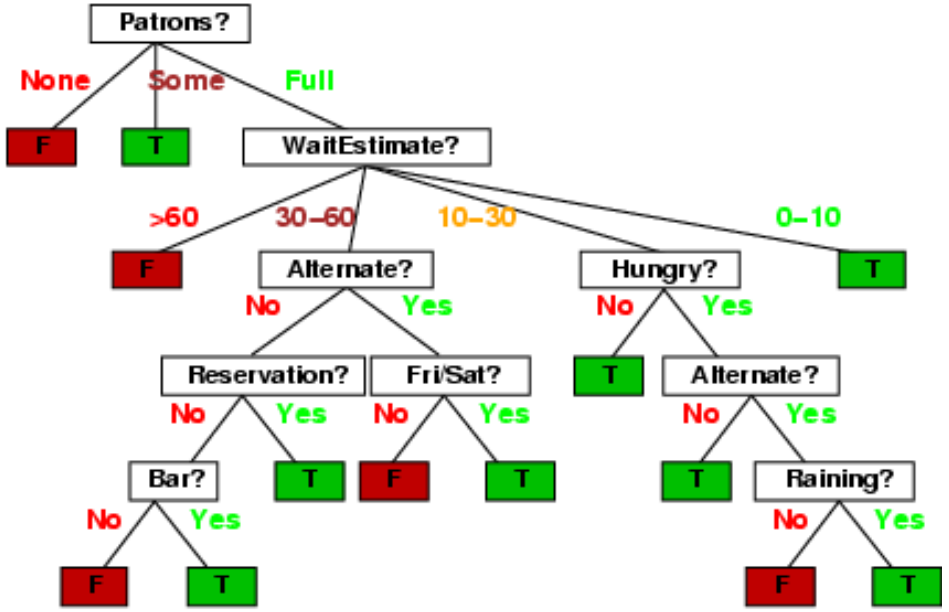
```
function DTL(examples, attributes, default) returns a decision tree
    if examples is empty then return default
    else if all examples have the same classification then return the classification
    else if attributes is empty then return MODE(examples)
    else
        best ← CHOOSE-ATTRIBUTE(attributes,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

1. No quedan ejemplos: valor por defecto calculado a partir de la mayoría en el nodo padre.
2. Todos los ejemplos son positivos o negativos.
3. No quedan atributos: voto de la mayoría de los ejemplos que quedan.
4. Quedan ejemplos positivos y negativos.

Árbol de decisión obtenido

Árbol de decisión obtenido utilizando los 12 ejemplos





Es más simple que el árbol “verdadero”.

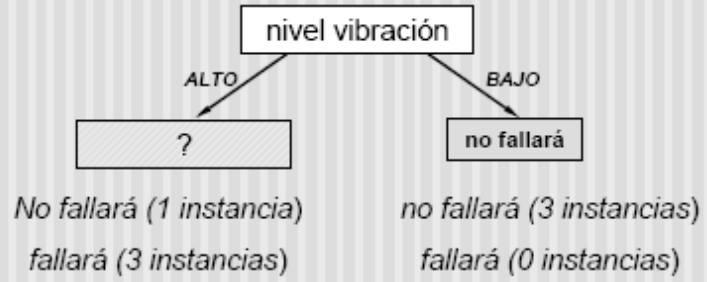
Ejemplo detallado

- ¿Qué atributo elegir para el primer nodo?

ATRIBUTO	VALORES	CLASE	
		fallará	*no fallará*
Temperatura	Alto	2	2
	Bajo	1	2
Nivel de vibraciones	Alto	3	1
	Bajo	0	3
Horas defuncionamiento	< 1000	2	3
	>1000	1	1
Meses desde revisión	> 1 mes	2	3
	< 1 mes	1	1

Ejemplo detallado

- Árbol construido hasta el momento:



- ¿Qué atributo se debe usar en el siguiente nivel del árbol (rama izquierda)?

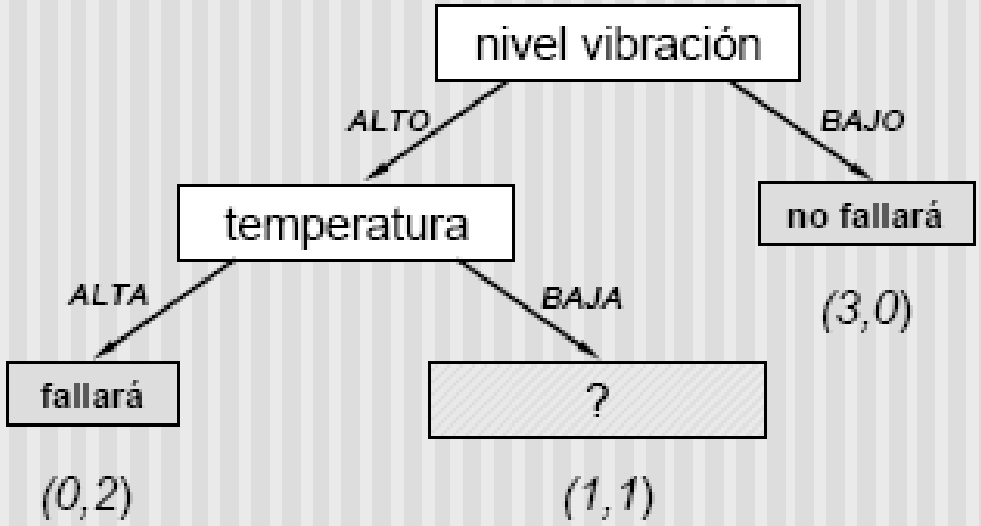
Ejemplo detallado

Sólo aquellos ejemplos de entrenamiento que llegan al nodo se utilizan para elegir el nuevo atributo:

ATRIBUTO	VALORES	CLASE	
		fallará	*No fallará*
Temperatura	Alta	2	0
	BAja	1	1
Horas de funcionamiento	< 1000	2	1
	>1000	1	0
Meses desde revisión	> 1 mes	2	1
	< 1 mes	1	0

Ejemplo detallado

- Árbol construido hasta el momento:



- ¿Qué atributo se debe usar en el siguiente nivel del árbol (rama dereecha)?

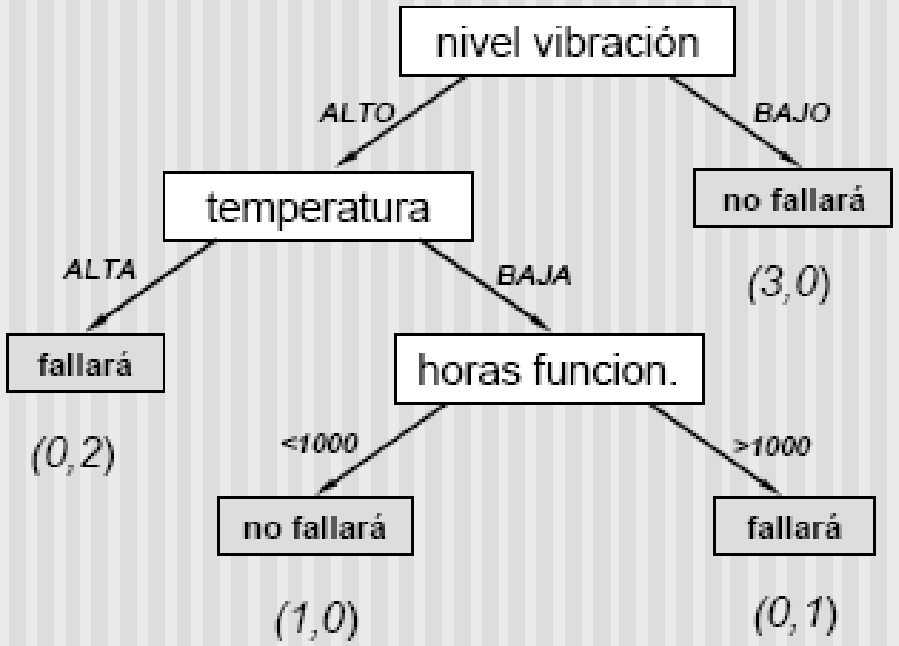
Ejemplo detallado

De nuevo, sólo aquellos ejemplos de entrenamiento que llegan al nodo se utilizan para elegir el nuevo atributo:

ATRIBUTO	VALORES	CLASE	
		fails	*works*
Horas de funcionamiento	< 1000	0	1
	>1000	1	0
Meses desde revisión	> 1 mes	1	1
	< 1 mes	0	0

Ejemplo detallado

- Árbol obtenido finalmente:



... muy similar al árbol original, utilizando sólo 7 ejemplos de entrenamiento!

Elección de los atributos de test

- Un atributo perfecto divide los ejemplos en conjuntos que contienen solo ejemplos positivos o negativos.
- Definir una medida de atributo “bastante adecuado” o “inadecuado”.

$$I(P(v_1), \ldots, P(v_n)) = \Sigma_{i=1} -P(v_i) \log_2 P(v_i)$$

- Para un conjunto de entrenamiento que contenga p ejemplos positivos y n ejemplos negativos

$$I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

Elección de los atributos de test

- Intuición: Mide la ausencia de “homegeneidad” de la clasificación.

- Teoría de la Información: cantidad media de información (en bits) necesaria para codificar la clasificación de un ejemplo.

- Ejemplos:
- $I([9+, 5-]) = -\ 9/14 \times \log_2 9/14 - 5/14 \times \log_2 5/14 = 0{,}94$
- $I([k+, k-]) = 1$ (ausencia total de homogeneidad)
- $I([p+, 0]) = I([0, n-]) = 0$ (homogeneidad total)

Ganancia de información

- Entropía esperada después de usar un atributo A en el árbol:

$$resto(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I\left(\frac{p_i}{p_i + n_i}, \frac{n_i}{p_i + n_i}\right)$$

- Ganancia de información esperada después de usar un atributo:

$$Ganancia(A) = I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) - resto(A)$$

- Se elige el atributo con mayor valor de G (ID3).

Ejemplo

Para el conjunto de entrenamiento, *p = n = 6, I(6/12, 6/12) = 1* bit

Considerando los atributos Clientes y Tipo:

$$Ganancia(Clientes) = 1 - [\frac{2}{12} I(0,1) + \frac{4}{12} I(1,0) + \frac{6}{12} I(\frac{2}{6}, \frac{4}{6})] = .0541 \text{ bits}$$

$$Ganancia(Tipo) = 1 - [\frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4})] = 0 \text{ bits}$$

Otras medidas

El criterio de ganancia tiene un fuerte sesgo a favor de test con muchas salidas:

Ratio de Ganancia:

$$RGanancia(A) = \frac{Ganancia(A)}{dINFO(A)}$$

con

$$dINFO(A) = -\sum_{i=1}^{v} \frac{p_i + n_i}{p+n} \log_2 \left(\frac{p_i + n_i}{p+n}\right)$$

Valoración de la calidad del algoritmo de aprendizaje

Un algoritmo de aprendizaje es bueno si produce hipótesis que hacen un buen trabajo al predecir clasificaciones de ejemplos que no ha sido observados.

Metodología

- Recolectar un conjunto de ejemplos grande.
- Dividir el conjunto de ejemplos en dos conjuntos: el conjunto de entrenamiento y el conjunto de test.
- Aplicar el algoritmo de aprendizaje al conjunto de entrenamiento, generando la hipótesis h.
- Medir el porcentaje de ejemplos del conjunto de test que h clasifica correctamente.
- Repetir los pasos del 1 al 4 para conjuntos de entrenamiento seleccionados aleatoriamente para cada tamaño.

Ruido y sobreajuste

- Ruido: dos o más ejemplos con la misma descripción (en términos de atributos) pero diferentes clasificaciones.
- Sobreajuste: encontrar “regularidades” poco significativas en los datos.

> Se dice que una hipótesis h se sobreajusta al conjunto de entrenamiento si existe alguna otra hipótesis h' tal que el error de h es menor que el de h' sobre el conjunto de entrenamiento, pero es mayor sobre la distribución completa de ejemplos del problema (entrenamiento + test).

Valores perdidos

- Asignar el valor más común entre todos los ejemplos de entrenamiento pertenecientes al nodo.

- Asignar una probabilidad a cada uno de los posibles valores del atributo basada en la frecuencia observada en los ejemplos pertenecientes al nodo. Finalmente, distribuir de acuerdo a dicha probabilidad.

WEKA

- Algoritmos clásicos como CART, ID3, C4.5 o CN2.
- También existen sistemas de inducción de reglas o árboles de inducción integrados en paquetes de minería de datos o aplicaciones estadísticas.
- WEKA es un conjunto de librerías JAVA para la extracción del conocimiento desde bases de datos.
- Fue desarrollado por la Universidad de Waikato bajo licencia GPL.
- WEKA contiene una gran cantidad de métodos y un entorno de experimentación para comparar el rendimiento de los algoritmos utilizando técnicas estadísticas.
- http://www.cs.waikato.ac.nz/ml/weka/