DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

Clasificación

© Fernando Berzal, berzal@acm.org

Clasificación

- Introducción
  - Construcción y uso de modelos de clasificación
  - Evaluación de la precisión de un modelo de clasificación
  - El problema del sobreaprendizaje
  - Descomposición del error en sesgo y varianza
- Modelos de clasificación
- Evaluación
  - Métricas
  - Métodos de evaluación
  - Teorema de Wolpert
- Ajuste de hiperparámetros
- Apéndice: Modelos de clasificación

Introducción

Clasificación

Modelo

Predictivo y, si es interpretable, descriptivo.

Objetivo

Aprendizaje supervisado de una función que relaciona un conjunto de variables (X) con una variable discreta/categórica (y).

Datos

Categóricos y numéricos.

Variantes y extensiones

Detección de anomalías, predicción de enlaces…

Introducción

Clasificación vs. Predicción

- **Clasificación:**
  Para predecir el valor de un atributo categórico (discreto o nominal).

- **Predicción:**
  Para modelar funciones que toman valores continuos (esto es, predecir valores numéricos desconocidos).

Introducción

Aplicaciones

- Concesión de créditos
- Campañas de marketing dirigido
- Diagnóstico médico
- Detección de fraudes
- …

Introducción

Construcción del modelo

- El conjunto de datos utilizado para construir el modelo de clasificación se denomina **conjunto de entrenamiento**.
- Cada caso/tupla/muestra corresponde a una clase predeterminada: los casos de entrenamiento vienen etiquetados por su atributo de clase.

Uso del modelo

- El modelo construido a partir del conjunto de entrenamiento se utiliza para clasificar nuevos datos.

Introducción

Aprendizaje
Supervisado vs. No Supervisado

- **Aprendizaje supervisado** (**clasificación**):
  Los casos del conjunto de entrenamiento aparecen etiquetados con la clase a la que corresponden.

- **Aprendizaje no supervisado** (**clustering**) :
  No se conocen las clases de los casos del conjunto de entrenamiento (ni siquiera su existencia).

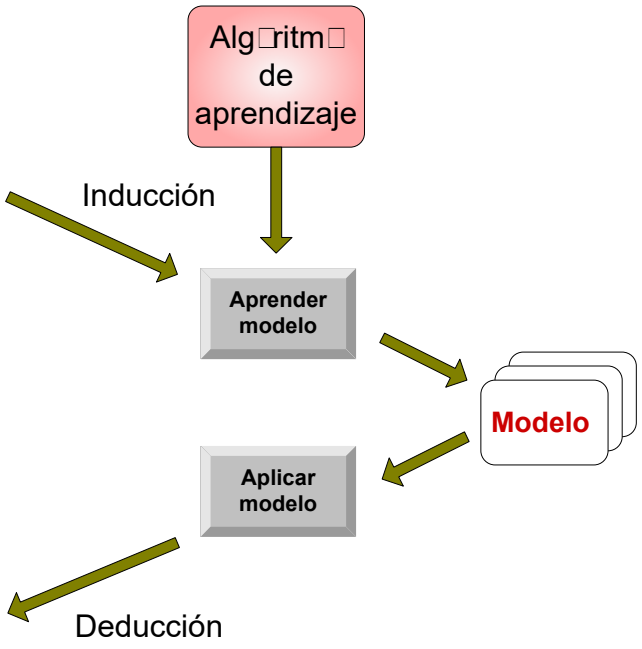
Introducción

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	N□	Medium	100K	No
3	N□	Small	70K	No
4	Yes	Medium	120K	No
5	N□	Large	95K	Yes
6	N□	Medium	60K	No
7	Yes	Large	220K	No
8	N□	Small	85K	Yes
9	N□	Medium	75K	No
10	N□	Small	90K	Yes

C□njunt□ de entrenamient□

Tid	Attrib1	Attrib2	Attrib3	Class
11	N□	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	N□	Small	95K	?
15	N□	Large	67K	?

C□njunt□ de prueba



Introducción

Estimación de la precisión del modelo

Antes de construir el modelo de clasificación, se divide el conjunto de datos disponible en

- un **conjunto de entrenamiento** (para construir el modelo) y
- un **conjunto de prueba** (para evaluar el modelo).

Introducción

Estimación de la precisión del modelo

- Una vez construido el modelo a partir del conjunto de entrenamiento, se usa dicho modelo para clasificar los datos del conjunto de prueba:
- Comparando los casos etiquetados del conjunto de prueba con el resultado de aplicar el modelo, se obtiene un **porcentaje de clasificación**.
- Si la precisión del clasificador es aceptable, podremos utilizar el modelo para clasificar nuevos casos (de los que desconocemos realmente su clase).

Introducción

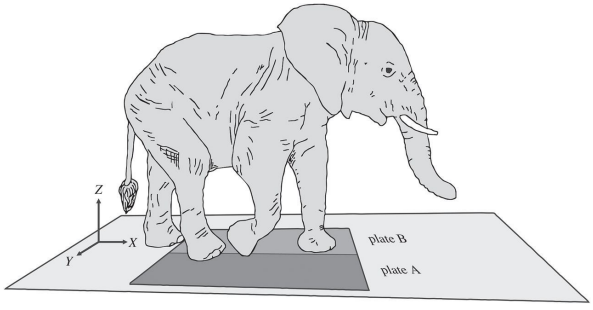
El problema del sobreaprendizaje

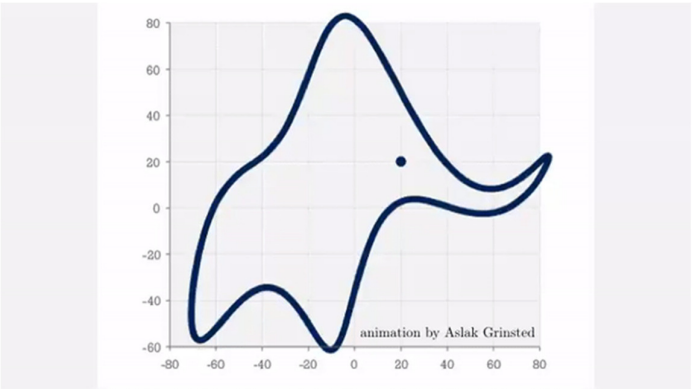
- Cuanto mayor sea su complejidad, los modelos de clasificación tienden a ajustarse más al conjunto de entrenamiento utilizado en su construcción (**sobreaprendizaje**), lo que los hace menos útiles para clasificar nuevos datos.
- En consecuencia, el conjunto de prueba debe ser siempre independiente del conjunto de entrenamiento.
- El error de clasificación en el conjunto de entrenamiento **NO** es un buen estimador de la precisión del clasificador.

Introducción

El problema del sobreaprendizaje

"Con cuatro parámetros puedo ajustar un elefante, con cinco puedo hacer que menee su trompa."
-- John von Neumann



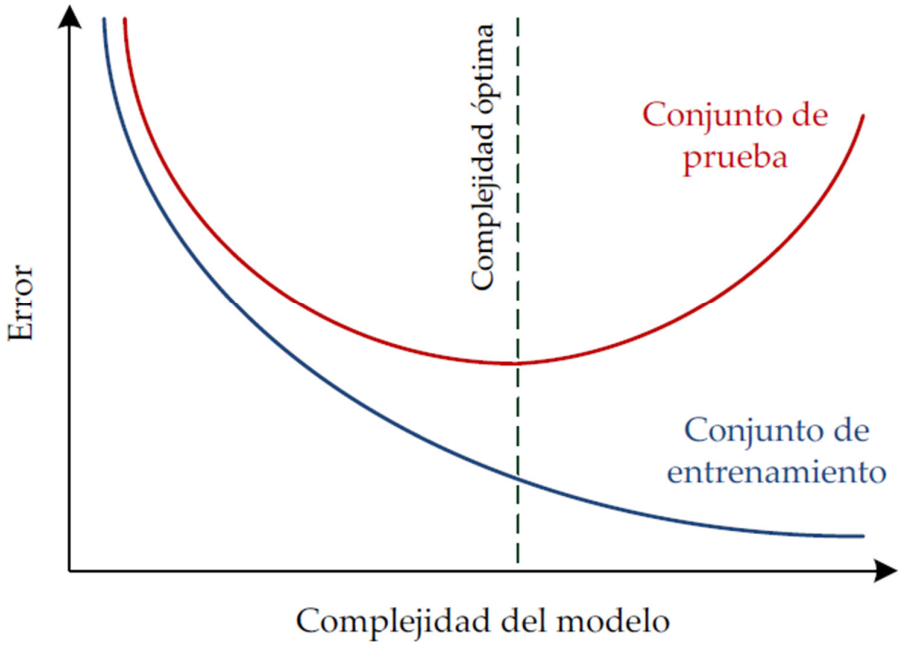


Jürgen Mayer, Khaled Khairy, y Jonathan Howard.
Drawing an elephant with four complex parameters.
American Journal of Physics, 78(6):648–649, 2010.
DOI 10.1119/1.3254017

Introducción

El problema del sobreaprendizaje

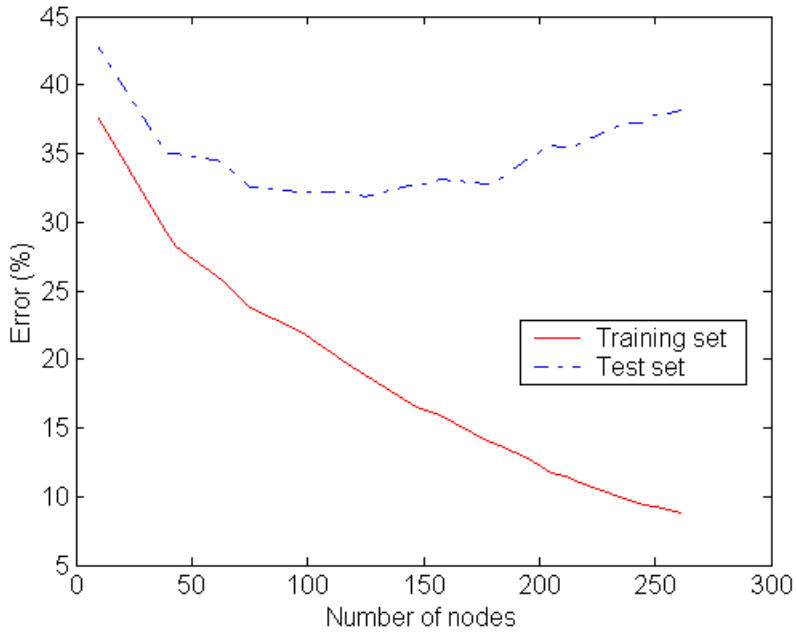
debido a la complejidad del modelo



Introducción

Sobreaprendizaje

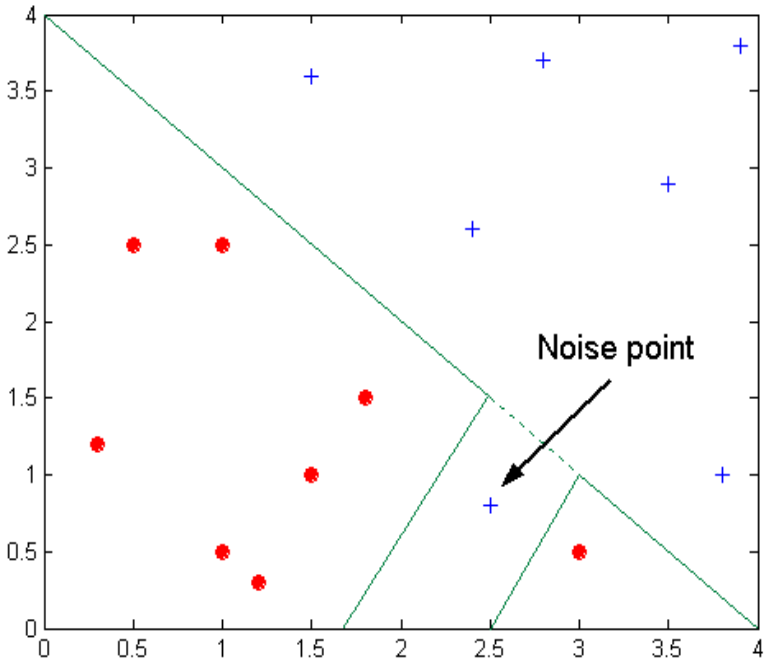
debido a la complejidad del clasificador



Introducción

Sobreaprendizaje

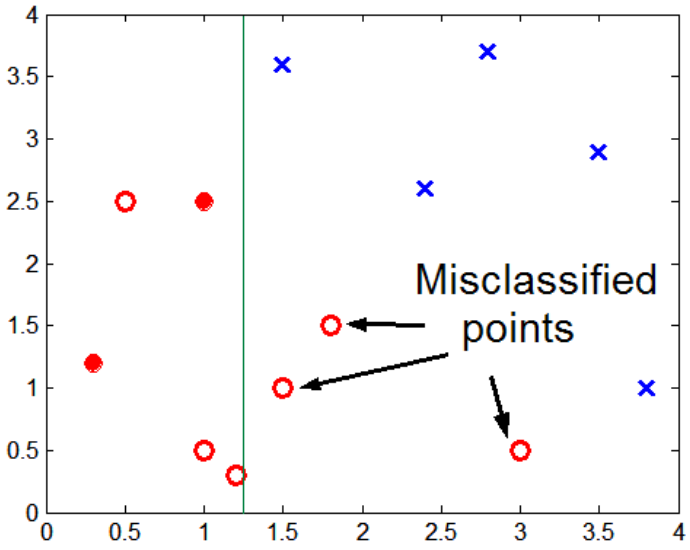
debido a la presencia de ruido en los datos:



Introducción

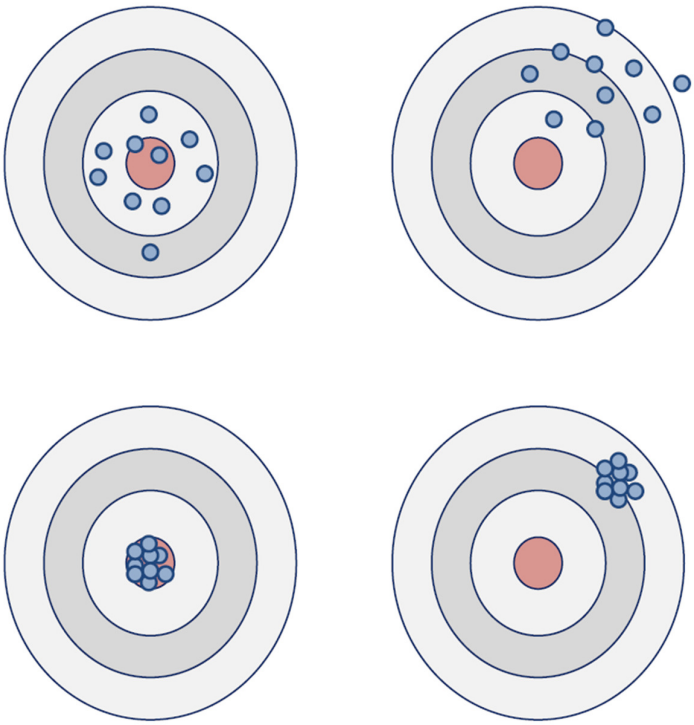
Sobreaprendizaje

debido a la escasez de muestras:



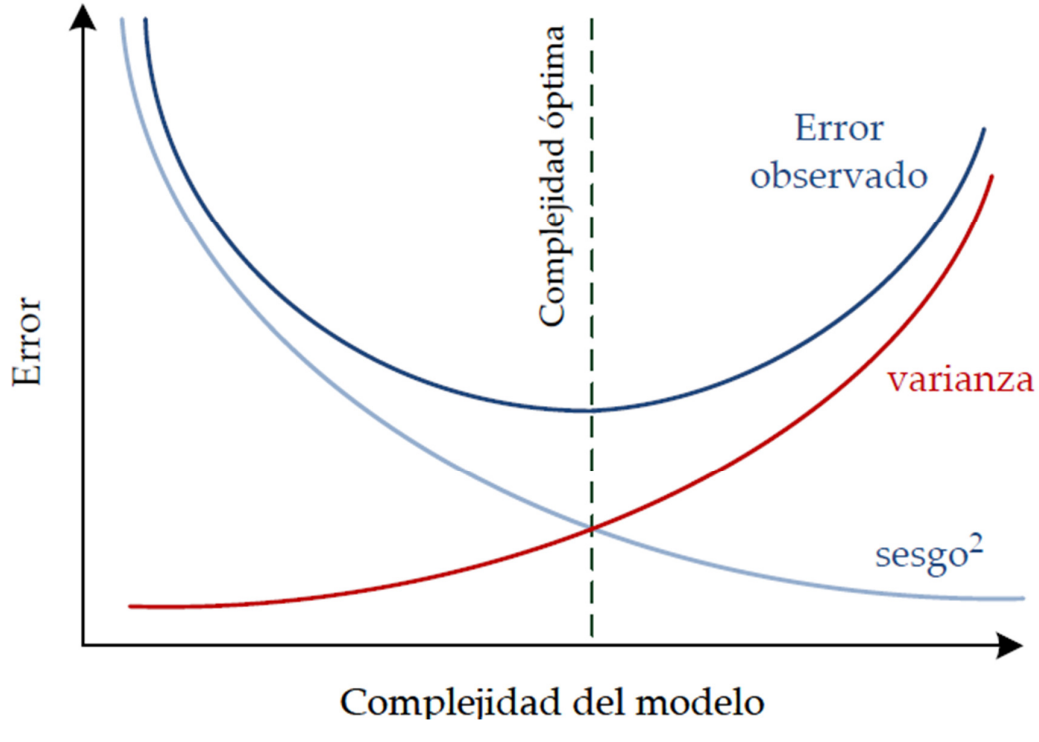
Introducción

Descomposición del error en sesgo y varianza



Introducción

Descomposición del error en sesgo y varianza



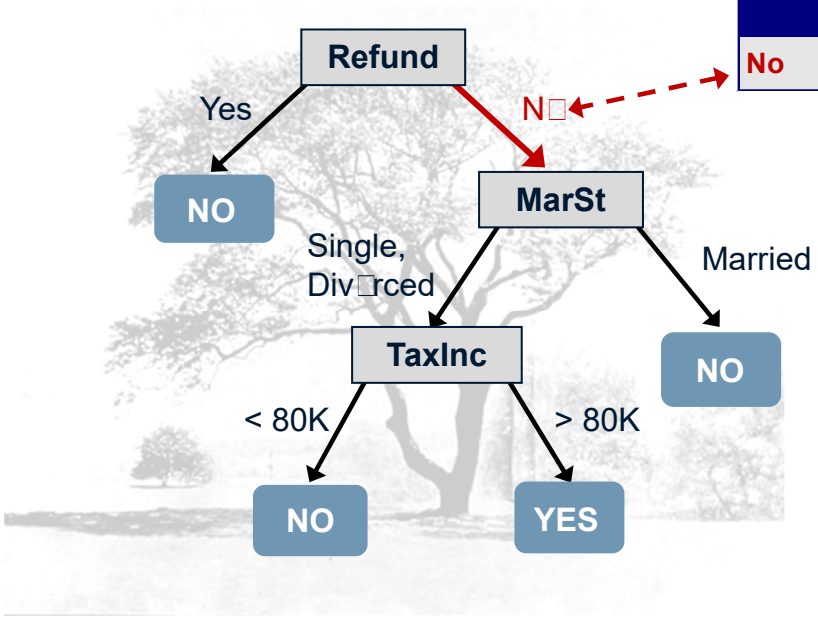
Modelos de clasificación

Se pueden construir distintos tipos de clasificadores:

- Árboles de decisión
- Reglas (p.ej. listas de decisión)
- Clasificadores basados en casos
- Clasificadores paramétricos
- Redes neuronales
- Redes bayesianas
- SVMs (Support Vector Machines)
- ...

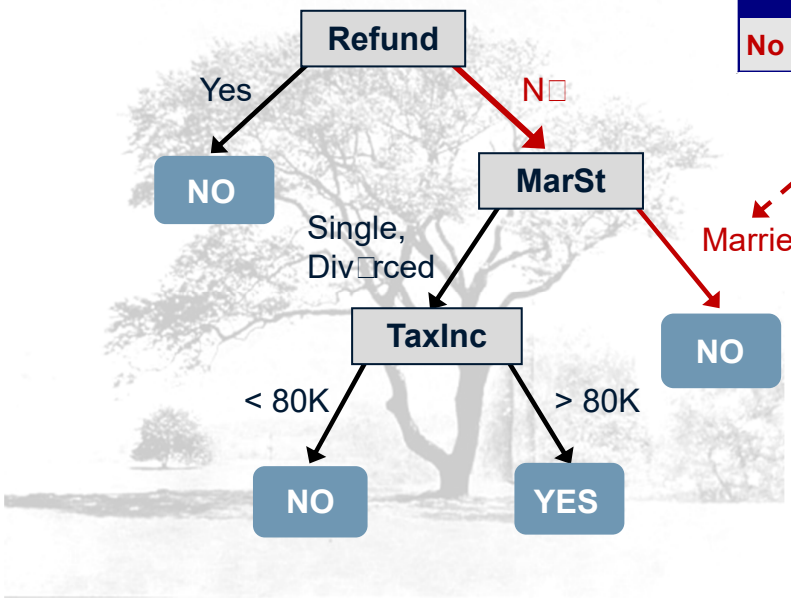
Árboles de decisión

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



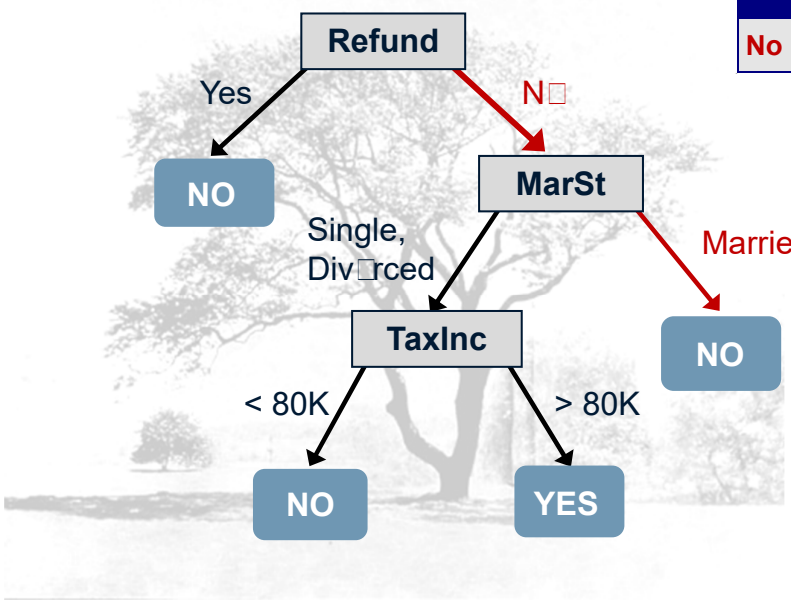
Árboles de decisión

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Árboles de decisión

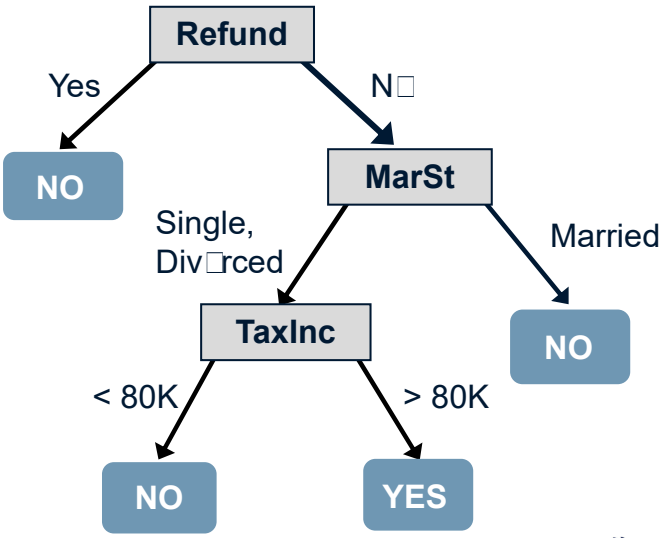
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	NO



Clase 'NO'

Árboles de decisión

	categórico	categórico	continuo	clase
Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**
1	Yes	Single	125K	No
2	N□	Married	100K	No
3	N□	Single	70K	No
4	Yes	Married	120K	No
5	N□	Div□rced	95K	Yes
6	N□	Married	60K	No
7	Yes	Div□rced	220K	No
8	N□	Single	85K	Yes
9	N□	Married	75K	No
10	N□	Single	90K	Yes



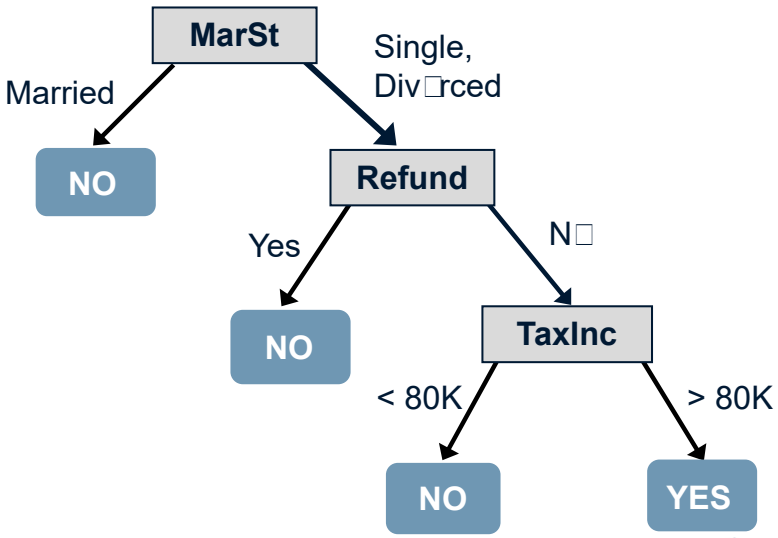
Conjunto de entrenamiento

Modelo de clasificación: Árbol de decisión

Árboles de decisión

	categórico	categórico	continuo	clase
Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**
1	Yes	Single	125K	No
2	N□	Married	100K	No
3	N□	Single	70K	No
4	Yes	Married	120K	No
5	N□	Div□rced	95K	Yes
6	N□	Married	60K	No
7	Yes	Div□rced	220K	No
8	N□	Single	85K	Yes
9	N□	Married	75K	No
10	N□	Single	90K	Yes

Podemos construir distintos árboles: ¿cuál es mejor?



Conjunto de entrenamiento

Modelo de clasificación: Árbol de decisión

Árboles de decisión

Construcción de árboles de decisión

- Estrategia greedy (problema NP)

- Algoritmo "divide y vencerás":
  - Comenzamos con todos los ejemplos de entrenamiento en la raíz del árbol de decisión.
  - Los ejemplos se van dividiendo en función del atributo que se seleccione para ramificar el árbol en cada nodo.
  - Los atributos que se usan para ramificar se eligen en función de una heurística (**regla de división**).

Árboles de decisión

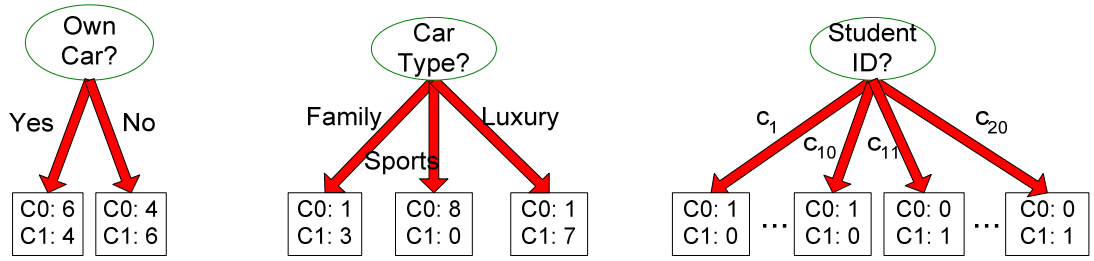
Construcción de árboles de decisión

- ¿Cuándo se detiene la construcción del árbol de decisión? **Criterios de parada**:
  - Cuando todos los ejemplos que quedan pertenecen a la misma clase (se añade una hoja al árbol con la etiqueta de la clase).
  - Cuando no quedan atributos por los que ramificar (se añade una hoja etiquetada con la clase más frecuente en el nodo).
  - Cuando no nos quedan datos que clasificar.

Árboles de decisión

Construcción de árboles de decisión

- ¿Qué heurísticas se pueden utilizar para decidir cómo ramificar el árbol?

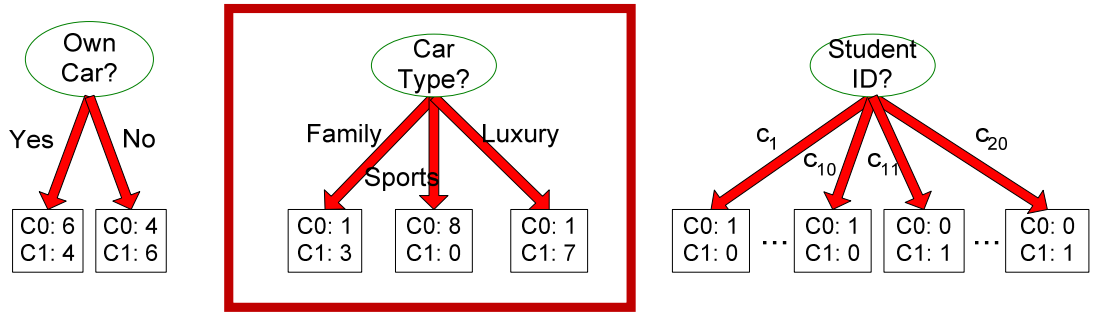


¿Cuál es mejor?

Árboles de decisión

Construcción de árboles de decisión

- ¿Qué heurísticas se pueden utilizar para decidir cómo ramificar el árbol?



La que nos proporciona nodos más homogéneos.

Necesitamos medir la impureza de un nodo.

Árboles de decisión

Construcción de árboles de decisión

- **Reglas de división**
  (heurísticas para la selección de atributos):
  - Ganancia de información (ID3, C4.5)
  - Índice de Gini (CART, SLIQ, SPRINT)

  Existen otras muchas reglas de división:
  χ^2, MDL (Minimum Description Length)…

Árboles de decisión

Entropía

Teoría de la Información: $$info(D) = -\sum_{i=1}^{k} p_i \log_2 p_i$$

C1	**0**
C2	**6**

Entropía = 0
= – 0 l□g$_2$ 0 – 1 l□g$_2$ 1 = 0

C1	**1**
C2	**5**

Entropía = 0.65
= – (1/6) l□g$_2$ (1/6) – (5/6) l□g$_2$ (5/6)

C1	**2**
C2	**4**

Entropía = 0.92
= – (2/6) l□g$_2$ (2/6) – (4/6) l□g$_2$ (4/6)

C1	**3**
C2	**3**

Entropía = 1
= – (1/2) l□g$_2$ (1/2) – (1/2) l□g$_2$ (1/2)

Árboles de decisión

Ganancia de información (ID3)

p_i Estimación de la probabilidad de que un ejemplo de D pertenezca a la clase C_i

Entropía

(información necesaria para clasificar un ejemplo en D)

$$info(D) = -\sum_{i=1}^{k} p_i \log_2 p_i$$

Árboles de decisión

Ganancia de información (ID3)

Información necesaria para clasificar D después de usar el atributo A para dividir D en v particiones:

$$info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times I(D_j)$$

Ganancia obtenida
al ramificar utilizando el atributo A:

$$gain(A) = info(D) - info_A(D)$$

Árboles de decisión

Criterio de proporción de ganancia (Gain Ratio, C4.5)

ID3 tiende a ramificar el árbol utilizando los atributos que tengan más valores diferentes,
por lo que se "normaliza" la ganancia de información usando la entropía de la partición
(que será mayor cuantas más particiones pequeñas haya):

$$splitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2 \frac{|D_j|}{|D|}$$

$$gainRatio(A) = gain(A) / splitInfo(A)$$

Árboles de decisión

Índice de Gini (CART, SLIQ, SPRINT)

Medida estadística de impureza:

$$gini(D) = 1 - \sum_{i=1}^{k} p_i^2$$

C1	0
C2	6
Gini=0.000	

C1	1
C2	5
Gini=0.278	

C1	2
C2	4
Gini=0.444	

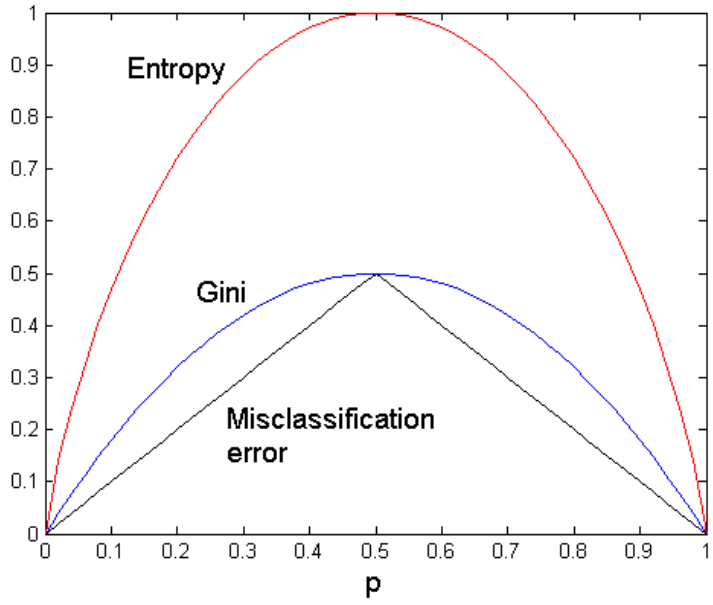
C1	3
C2	3
Gini=0.500	

- Para construir el árbol, elegimos el atributo que proporciona la mayor reducción de impureza

Árboles de decisión

Comparación de reglas de división

Para problemas con dos clases:



$$info(D) = -\sum_{i=1}^{k} p_i \log_2 p_i$$

$$gini(D) = 1 - \sum_{i=1}^{k} p_i^2$$

$$error(D) = 1 - \max p_i$$

Árboles de decisión

Comparación de reglas de división

- **Ganancia de información**
  Sesgado hacia atributos con muchos valores diferentes.
- **Criterio de proporción de ganancia**
  Tiende a preferir particiones poco balanceadas (con una partición mucho más grande que las otras).
- **Índice de Gini**
  Funciona peor cuando hay muchas clases y tiende a favorecer particiones de tamaño y pureza similares.

Ninguna regla de división es significativamente mejor que los demás.

Árboles de decisión

Otros aspectos de interés

- ¿Árboles binarios o n-arios?
  (CART binario; C4.5 n-ario para atributos categóricos, binario para atributos continuos).

- Manejo de atributos continuos
  (selección del conjunto de tests candidatos para ramificar el árbol, p.ej. discretización previa).

- Manejo de valores nulos
  (cómo se tratan los valores nulos/desconocidos).

Árboles de decisión

Ejemplo

Outlook	Temperature	Humidity	Windy	Play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Árboles de decisión

Ejemplo

Para el cálculo de las entropías…

n	$\log_2(n)$
1	0,000
2	1,000
3	1,585
4	2,000
5	2,322
6	2,585
7	2,807
8	3,000
9	3,170
10	3,322
11	3,459
12	3,585
13	3,700
14	3,807
15	3,907
16	4,000

Árboles de decisión

Ejemplo

Cálculo de las entropías E(+,-)

$E(+,-) = - P(+) \log_2 P(+) - P(-) \log_2 P(-)$

E(+,-)	0-	1-	2-	3-	4-	5-
0+		0,000	0,000	0,000	0,000	0,000
1+	0,000	1,000	0,918	0,811	0,722	0,650
2+	0,000	0,918	1,000	0,971	0,918	0,863
3+	0,000	0,811	0,971	1,000	0,985	0,954
4+	0,000	0,722	0,918	0,985	1,000	0,991
5+	0,000	0,650	0,863	0,954	0,991	1,000
6+	0,000	0,592	0,811	0,918	0,971	0,994
7+	0,000	0,544	0,764	0,881	0,946	0,980
8+	0,000	0,503	0,722	0,845	0,918	0,961
9+	0,000	0,469	0,684	0,811	0,890	0,940

Árboles de decisión

Ejemplo

Raíz del árbol (9+,5-)

Info(D) = E(9+,5-) = 0.940 bits

Ramificando por el atributo **"Outlook"**

$Info_{Outlook}(D)$
= (5/14) $Info(D_{sunny})$ + (4/14) $Info(D_{overcast})$ + (5/14) $Info(D_{rainy})$
= (5/14) E(2+,3-) + (4/14) E(4+,0-) + (5/14) E(3+,2-)
= (5/14) · 0.971 + (4/14) · 0 + (5/14) · 0.971 = 0.693 bits

Gain(Outlook) = Info(D) - $Info_{Outlook}(D)$ = **0.247 bits**

Árboles de decisión

Ejemplo

Raíz del árbol (9+,5-)

Info(D) = E(9+,5-) = 0.940 bits

Ramificando por el atributo **"Temperature"**

$Info_{Temperature}(D)$
= (4/14) $Info(D_{cool})$ + (6/14) $Info(D_{mild})$ + (4/14) $Info(D_{hot})$
= (4/14) E(3+,1-) + (6/14) E(4+,2-) + (4/14) E(2+,2-)
= (4/14) · 0.811 + (6/14) · 0.918 + (4/14) · 1 = 0.911 bits

Gain(Temperature) = Info(D) - $Info_{Temperature}(D)$ = **0.029 bits**

Árboles de decisión

Ejemplo

Raíz del árbol (9+,5-)

Info(D) = E(9+,5-) = 0.940 bits

Ramificando por el atributo **"Humidity"**

$Info_{Humidity}(D)$
= (7/14) $Info(D_{high})$ + (7/14) $Info(D_{normal})$
= (7/14) E(3+,4-) + (7/14) E(6+,1-)
= (7/14) · 0.985 + (7/14) · 0.592 = 0.789 bits

Gain(Humidity) = Info(D) - $Info_{Humidity}(D)$ = **0.151 bits**

Árboles de decisión

Ejemplo

Raíz del árbol (9+,5-)

Info(D) = E(9+,5-) = 0.940 bits

Ramificando por el atributo **"Windy"**

$Info_{Windy}(D)$
= (8/14) $Info(D_{false})$ + (6/14) $Info(D_{true})$
= (8/14) E(6+,2-) + (6/14) E(3+,3-)
= (8/14) · 0.811 + (6/14) · 1 = 0.892 bits

Gain(Windy) = Info(D) - $Info_{Windy}(D)$ = **0.048 bits**

Árboles de decisión

Ejemplo

Raíz del árbol (9+,5-)

Gain(Outlook) = Info(D) - $Info_{Outlook}(D)$ = **0.247 bits**

Gain(Temperature) = Info(D) - $Info_{Temperature}(D)$ = **0.029 bits**

Gain(Humidity) = Info(D) - $Info_{Humidity}(D)$ = **0.151 bits**

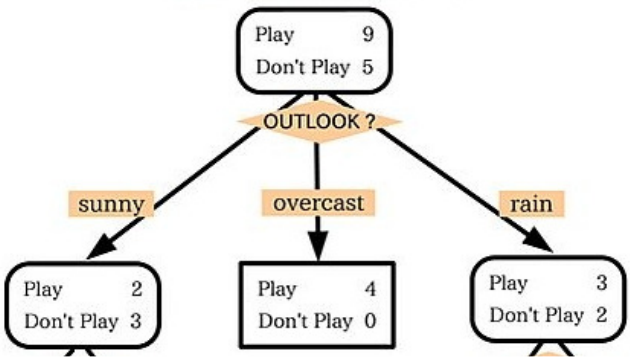
Gain(Windy) = Info(D) - $Info_{Windy}(D)$ = **0.048 bits**

Por tanto, ramificamos usando el atributo "Outlook"...

Árboles de decisión

Ejemplo

Nuestro árbol de decisión está así ahora mismo...

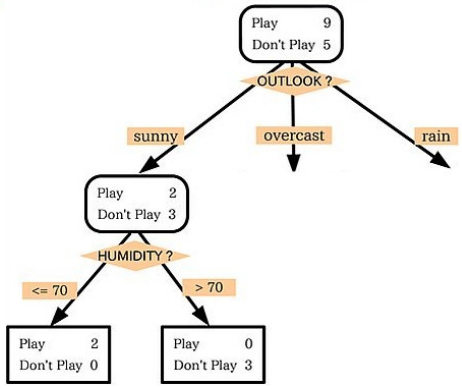


... pero aún tenemos que seguir construyéndolo.

Árboles de decisión

Ejemplo

Nodo "Outlook = sunny" (2+,3-)



$Info(D_s) = E(2+,3-) = 0.971$

Temperature: { (0+,2-), (1+,1-), (1+,0-) }
$Gain(Temperature) = Info(D_s) - Info_{Temperature}(D_s)$ = **0.571 bits**

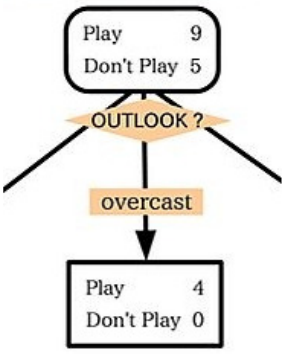
Humidity: { (0+,3-), (2+,0-) }
$Gain(Humidity) = Info(D_s) - Info_{Humidity}(D_s)$ = **0.971 bits**

Windy: { (1+,2-), (1+,1-) }
$Gain(Windy) = Info(D_s) - Info_{Windy}(D_s)$ = **0.019 bits**

Árboles de decisión

Ejemplo

Nodo "Outlook = overcast" (4+,0-)



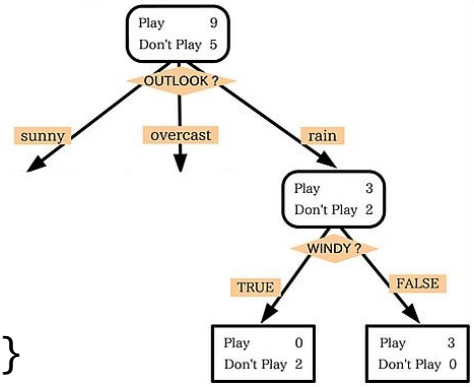
$Info(D_o) = E(4+,0-) = 0.000$

Creamos un nodo hoja directamente,
ya que todos los ejemplos son de la misma clase.

Árboles de decisión

Ejemplo

Nodo "Outlook = rainy" (3+,2-)



$Info(D_r) = E(3+,2-) = 0.971$

Temperature: { (0+,0-), (2+,1-), (1+,1-) }
$Gain(Temperature) = Info(D_r) - Info_{Temperature}(D_r) < \mathbf{0}$

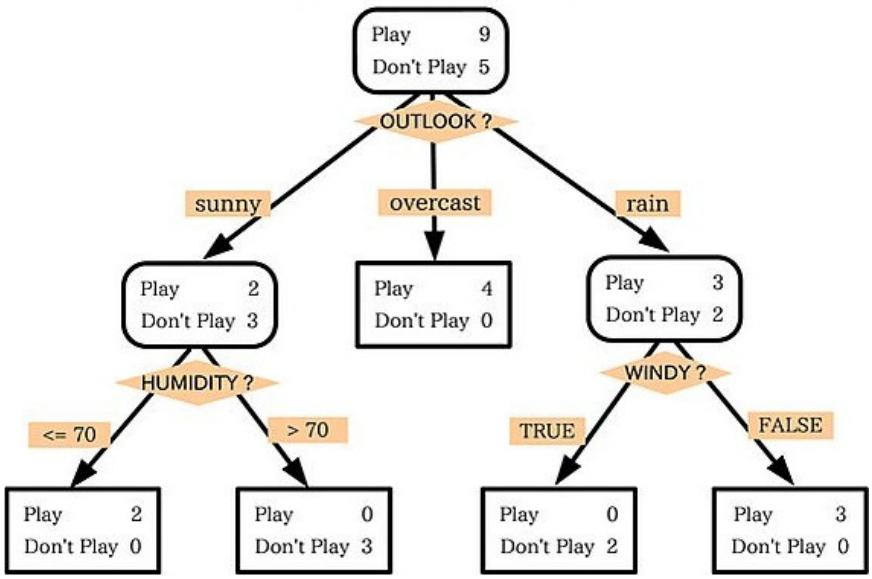
Humidity: {(2+,1-), (1+,1-) }
$Gain(Humidity) = Info(D_r) - Info_{Humidity}(D_r) < \mathbf{0}$

Windy: { (0+,2-), (3+,0-) }
$Gain(Windy) = Info(D_r) - Info_{Windy}(D_r) =$ **0.971 bits**

Árboles de decisión

Ejemplo

Resultado final…



Árboles de decisión

El problema del sobreaprendizaje

Los árboles de decisión tienden a ajustarse demasiado al conjunto de entrenamiento utilizado para construir el árbol:

- Demasiadas ramas del árbol reflejan anomalías del conjunto de entrenamiento (ruido y outliers).
- El árbol resultante es más complejo de lo que debería ser.
- Como consecuencia, disminuye la precisión del clasificador de cara a situaciones nuevas.

Árboles de decisión

El problema del sobreaprendizaje

Una solución al problema del sobreaprendizaje:

Técnicas de poda

Una vez construido el árbol, se van eliminando ramas: utilizando un conjunto de datos distinto al conjunto de entrenamiento [CART: Poda por coste-complejidad] o no [C4.5: Poda pesimista].

Árboles de decisión

El problema del sobreaprendizaje

Técnicas de poda

Para podar un árbol de decisión, se sustituye…

- un subárbol por un nodo hoja (correspondiente a la clase más frecuente en el subárbol), o bien,
- un subárbol por otro subárbol contenido en el primero.

Por tanto, se introducirán errores de clasificación adicionales en el conjunto de entrenamiento (aunque, si la poda se realiza correctamente, la precisión del clasificador aumentará).

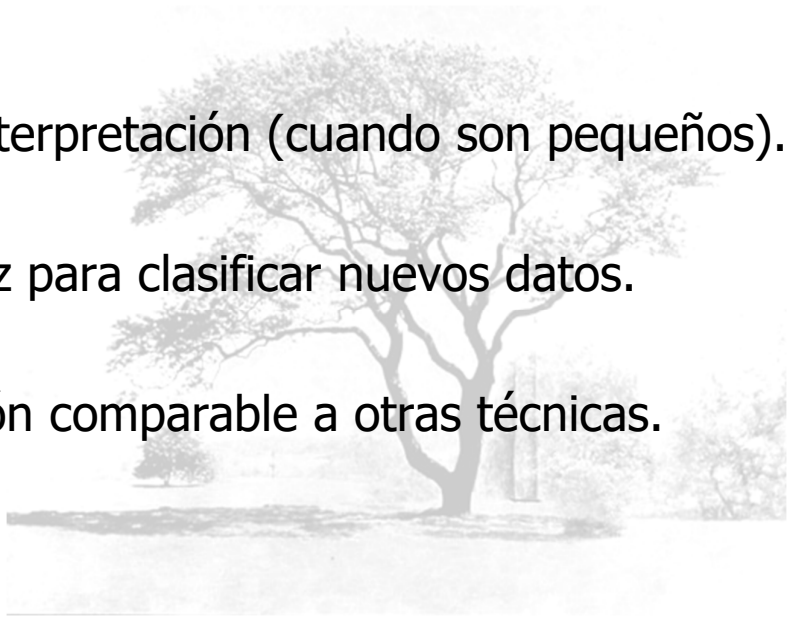
Árboles de decisión

Algoritmos eficientes y escalables

- **PUBLIC** (Rastogi & Shim, VLDB'1998)
  integra la poda en el proceso de construcción del árbol.
- **RainForest** (Gehrke et al., VLDB'1998)
  separa lo que determina la escalabilidad del algoritmo.
- **BOAT** (Gehrke et al., PODS'1999)
  sólo necesita recorrer 2 veces el conjunto de datos.

Árboles de decisión

Ventajas de los árboles de decisión



- Fácil interpretación (cuando son pequeños).
- Rapidez para clasificar nuevos datos.
- Precisión comparable a otras técnicas.

Árboles de decisión

DEMO

TDIDT

Top-Down Induction of Decision Trees

Reglas

Objetivo

Clasificar registros utilizando una colección de reglas "if then" de la formna **IF condición THEN y**, donde:

- **condición** es una conjunción de condiciones sobre el valor de varios atributos (antecedente de la regla)
- **y** es el valor de la clase (consecuente de la regla).

EJEMPLOS DE REGLAS

si sangre=caliente **y** pone_huevos=sí **entonces** pájaro

si ingresos<=30k€ **y** devolución=sí **entonces** no evasor

Reglas

Existen muchas formas de construir modelos de clasificación basados en reglas:

- A partir de un árbol de decisión.
- Diseñando algoritmos específicos de inducción de reglas:
  - Metodología STAR de Michalski
  - Listas de decisión (p.ej. RIPPER).
- A partir de reglas de asociación (a.k.a. clasificadores asociativos).

Reglas

A partir de un árbol de decisión

¿Por qué?

Las reglas son más fáciles de interpretar que un árbol de decisión complejo.

¿Cómo?

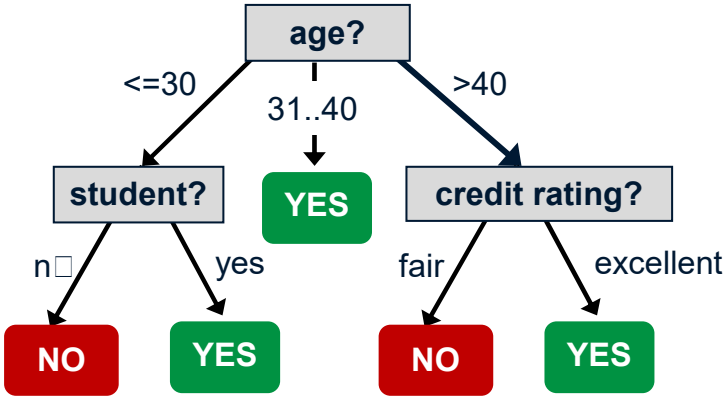
Se crea una regla para cada hoja del árbol.

Las reglas resultantes son

- mutuamente excluyentes y
- exhaustivas.

Reglas

A partir de un árbol de decisión



IF (age<=30) AND (student=no)
THEN buys_computer = NO
IF (age<=30) AND (student=yes)
THEN buys_computer = YES
IF (30<age<=40)
THEN buys_computer = YES
IF (age>40) AND (credit_rating=excellent)
THEN buys_computer = YES
IF (age>40) AND (credit_rating=fair)
THEN buys_computer = NO

Reglas

A partir de un árbol de decisión

Las reglas que se derivan de un árbol se pueden simplificar (generalizar), aunque entonces:

- Dejan de ser mutuamente excluyentes: varias reglas pueden ser válidas para un mismo ejemplo (hay que establecer un orden entre las reglas [lista de decisión] o realizar una votación).
- Dejan de ser exhaustivas: puede que ninguna regla sea aplicable a un ejemplo concreto (hace falta incluir una clase por defecto).

Reglas

Inducción de reglas

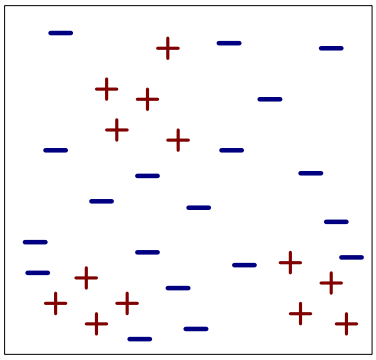
(directamente a partir del conjunto de entrenamiento)

p.ej**. LISTAS DE DECISIÓN**

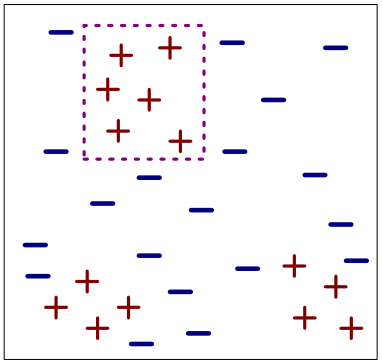
¿Cómo? Algoritmo de recubrimiento secuencial

- Las reglas se aprenden de una en una.
- Cada vez que se escoge una regla, se eliminan del conjunto de entrenamiento todos los casos cubiertos por la regla seleccionada.
- El proceso se repite iterativamente hasta que se cumpla alguna condición de parada.

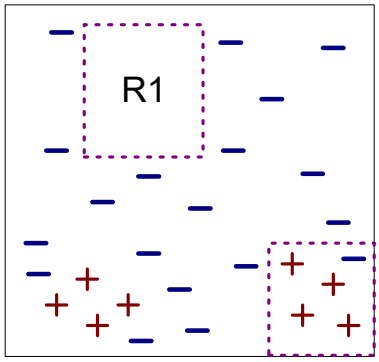
Reglas



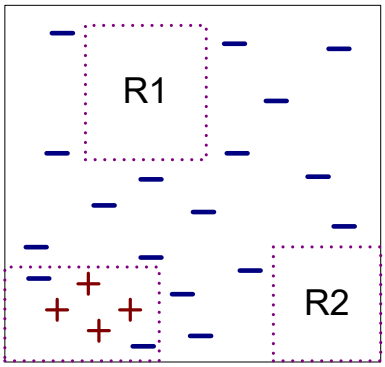
(i) Original Data



(ii) Step 1



(iii) Step 2



(iv) Step 3

Reglas

Inducción de reglas

(directamente a partir del conjunto de entrenamiento)

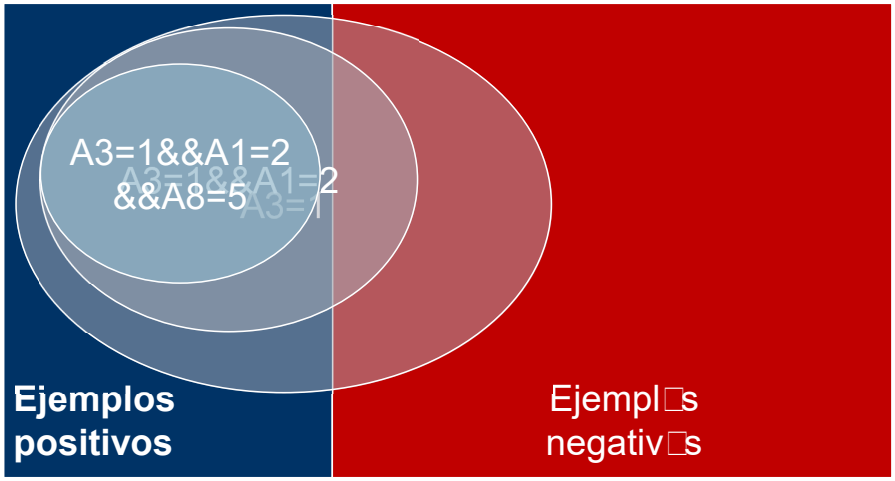
p.ej. **LISTAS DE DECISIÓN**

¿Cómo se aprende una regla?

- Se empieza con la regla más general posible.
- Se le van añadiendo antecedentes a la regla para maximizar la "calidad" de la regla (cobertura y precisión).

Reglas

Inducción de reglas



Reglas

Inducción de reglas

(directamente a partir del conjunto de entrenamiento)

p.ej. **LISTAS DE DECISIÓN**

Algoritmos de inducción de reglas

- FOIL (Quinlan, Machine Learning, 1990)
- CN2 (Clark & Boswell, EWSL'1991)
- RIPPER (Cohen, ICML'1995)
- PNrule (Joshi, Agarwal & Kumar, SIGMOD'2001)

Reglas

DEMO

CN2

- Metodología STAR: Unordered CN2
- Listas de decisión: Ordered CN2

RIPPER

Repeated Incremental Pruning to Produce Error Reduction
(basado en **IREP**, Iterative Reduced Error Pruning)

Reglas

Una regla tiene…

- **Cobertura**:
  Proporción de registros que satisfacen su antecedente.
- **Precisión**: Proporción de registros que satisfacen antecedente y consecuente simultáneamente.

En cuanto a los conjuntos de reglas:

- Si no son exhaustivos (no cubren cualquier combinación posible de valores de atributos), se define una **clase por defecto**.
- Si no son mutuamente excluyentes (un caso puede estar cubierto por varias reglas), se ordenan de acuerdo a algún criterio (p.ej. listas de decisión) o se realiza una "votación".

Evaluación

Criterios de evaluación

- **Precisión**
  (porcentaje de casos clasificados correctamente).
- **Eficiencia**
  (tiempo necesario para construir/usar el clasificador).
- **Robustez**
  (frente a ruido y valores nulos)
- **Escalabilidad**
  (utilidad en grandes bases de datos)
- **Interpretabilidad**
  (el clasificador, ¿es sólo una caja negra?)
- **Complejidad**
  (del modelo de clasificación) → Navaja de Occam.

Evaluación

Métricas

Cómo evaluar la "calidad"
de un modelo de clasificación.

Métodos

Cómo estimar, de forma fiable,
la calidad de un modelo.

Comparación

Cómo comparar el rendimiento relativo
de dos modelos de clasificación alternativos

Evaluación: Métricas

Matriz de confusión [confusion matrix]

		Predicción	
		C_P	C_N
Clase real	C_P	TP: True positive	FN: False negative
	C_N	FP: False positive	TN: True negative

Número de ejemplos
N= TP+TN+FP+FN

Precisión del clasificador
accuracy = (TP+TN)/N
Tasa de error del clasificador
error = (FP+FN)/N = 1 - accuracy

Evaluación: Métricas

Limitaciones de la precisión [accuracy]

Supongamos un problema con 2 clases no equilibradas:

- 99900 personas sanas
- 100 personas que padecen una enfermedad

Precisión engañosa: Si el modelo de clasificación siempre dice que los ejemplos son de la clase 1, su precisión es 99900/100000 = **99.9%**

Paradoja de los falsos positivos: Una prueba diagnóstica con el 99% de precisión identificará 99 de los 100 casos existentes, pero también 999 falsos positivos. Sólo un **9%** de los positivos lo son realmente!!!

Evaluación: Métricas

Alternativa: Matriz de costes

C(i|j)		Predicción	
		C_P	C_N
Clase real	C_P	C(P|P)	C(N|P)
	C_P	C(P|N)	C(N|N)

El coste de clasificación será proporcional a la precisión del clasificador sólo si

$\forall i,j: i \neq j \quad C(i|j) = C(j|i)$

$C(i|i) = C(j|j)$

Evaluación: Métricas

Medidas "cost-sensitive"

		Predicción	
		C_P	C_N
Clase real	C_P	TP: True positive	FN: False negative
	C_N	FP: False positive	TN: True negative

precision = TP/(TP+FP)

True positive recognition rate

recall = sensitivity = TP/P = TP/(TP+FN)

True negative recognition rate

specificity = TN/N = TN/(TN+FP)

Evaluación: Métricas

Medidas "cost-sensitive"

		Predicción	
		C_P	C_N
Clase real	C_P	TP: True positive	FN: False negative
	C_N	FP: False positive	TN: True negative

F-measure

Media armónica de precision y recall:

F = 2*precision*recall / (precision+recall)

F = 2TP / (2TP+FP+FN)

Evaluación: Métricas

Medidas sensibles a costes [cost-sensitive]

		Predicción	
		C_P	C_N
Real	C_P	TP	FN
	C_N	FP	TN

Accuracy

		Predicción	
		C_P	C_N
Real	C_P	TP	FN
	C_N	FP	TN

Recall

		Predicción	
		C_P	C_N
Real	C_P	TP	FN
	C_N	FP	TN

Precision

		Predicción	
		C_P	C_N
Real	C_P	TP	FN
	C_N	FP	TN

F-measure

Datos no balanceados

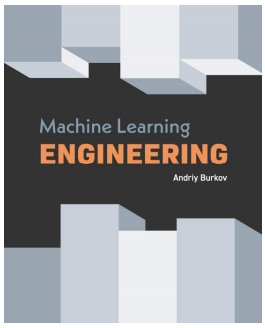
Una distribución muy desigual de las clases puede ocasionar problemas a la hora de entrenar un modelo.

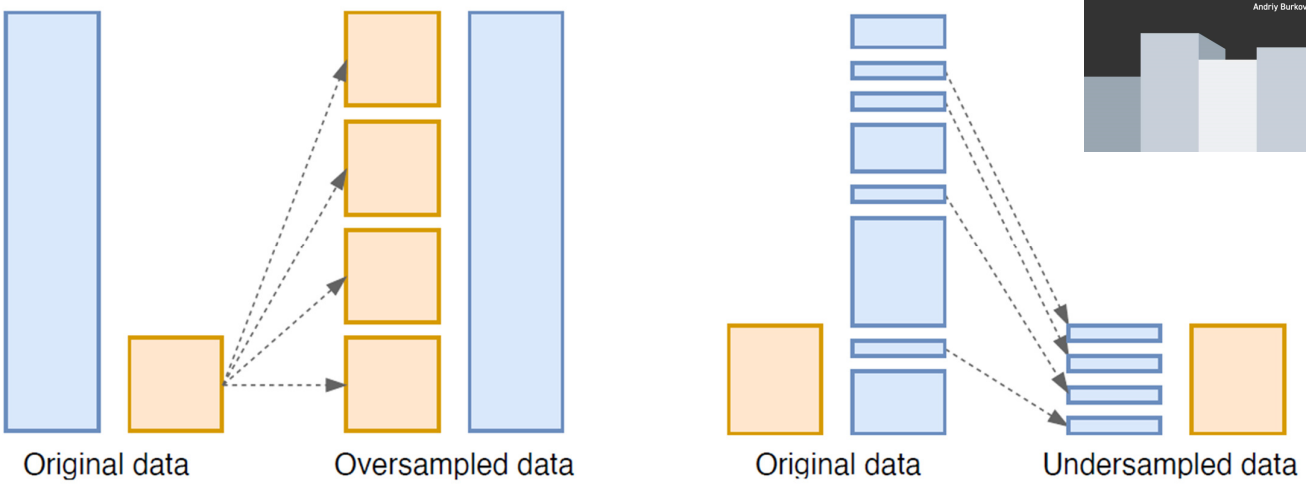
Posibles soluciones:

- Ponderación de las clases [class weighting], e.g. SVMs, árboles de decisión & random forests
- Técnicas de muestreo
- Técnicas de detección de anomalías

Datos no balanceados

Técnicas de muestreo

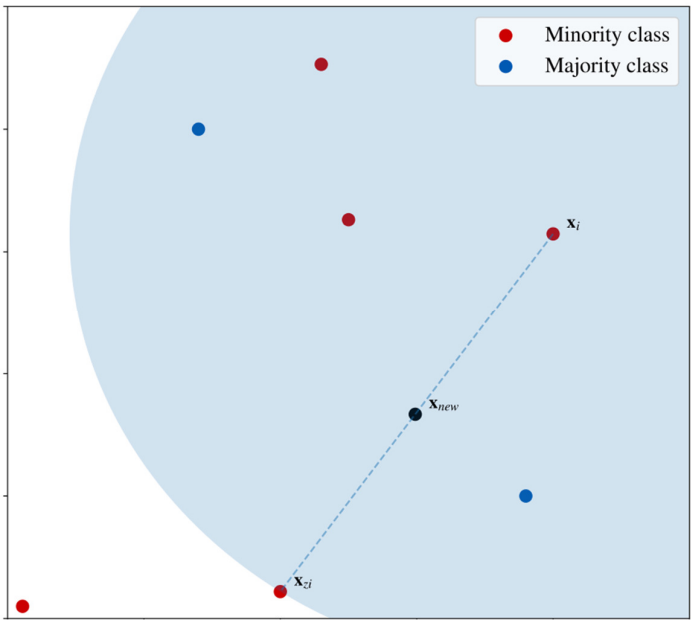




Datos no balanceados

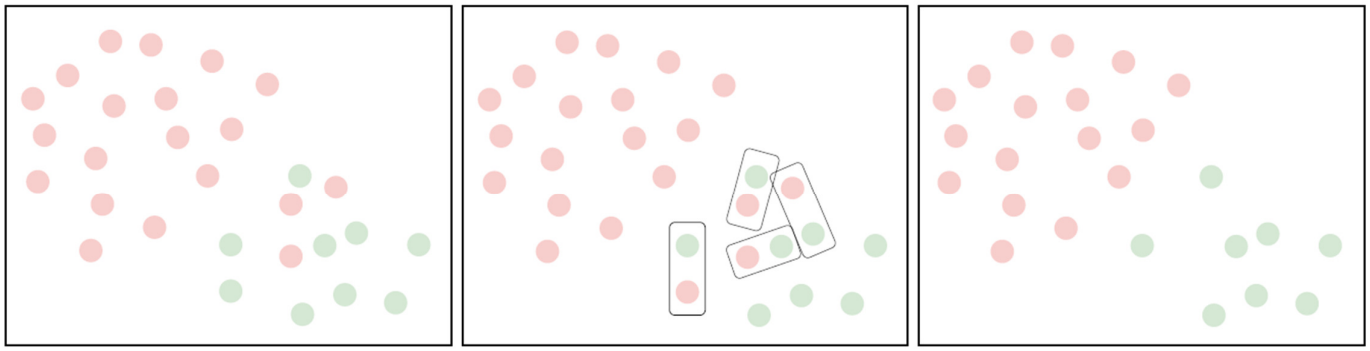
Sobremuestreo de la clase minoritaria

- SMOTE [Synthetic Minority Oversampling Technique]
- ADASYN [Adaptive Synthetic Sampling Method]



Datos no balanceados

Submuestreo de la clase mayoritaria



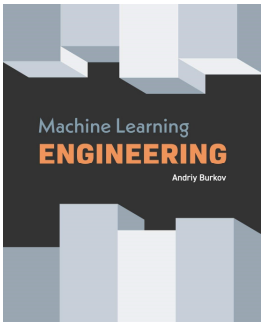
(a) original data (b) Tomek links (c) undersampled data

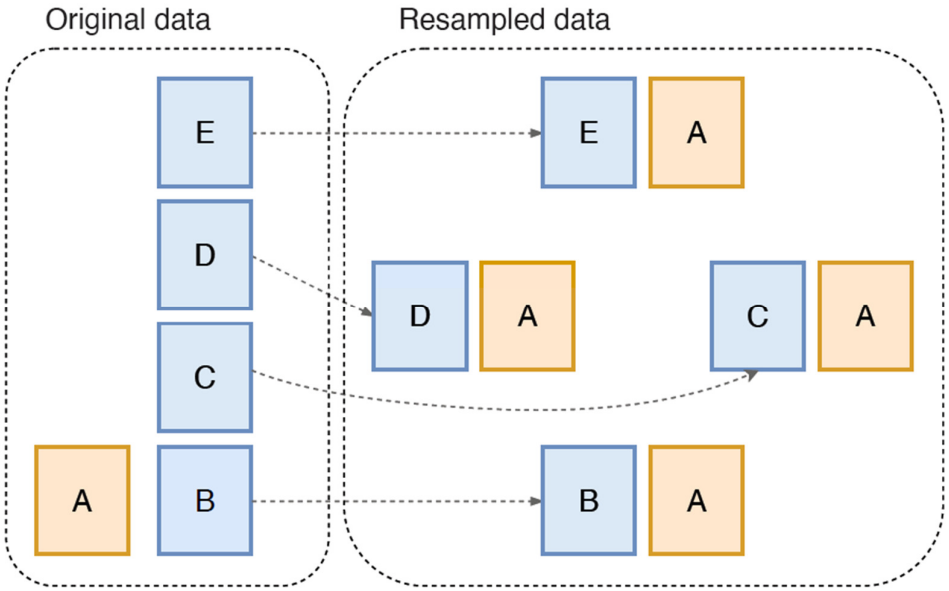
- Enlaces de Tomek (criterio de selección de los ejemplos que se eliminan).
- Métodos de agrupamiento (sustituyen los datos por representantes de cada cluster, p.ej. centroides).

Datos no balanceados

Ensemble [remuestreo del conjunto de datos]

Se transforma un problema no balanceado en N problemas balanceados usando "bagging":





Evaluación: Métodos

Para evaluar la precisión de un modelo de clasificación nunca debemos utilizar el conjunto de entrenamiento (lo que nos daría el "**error de resustitución**" del clasificador), sino un conjunto de prueba independiente:

Por ejemplo, podríamos reservar 2/3 de los ejemplos disponibles para construir el clasificador y el 1/3 restante lo utilizaríamos de **conjunto de prueba** para estimar la precisión del clasificador.

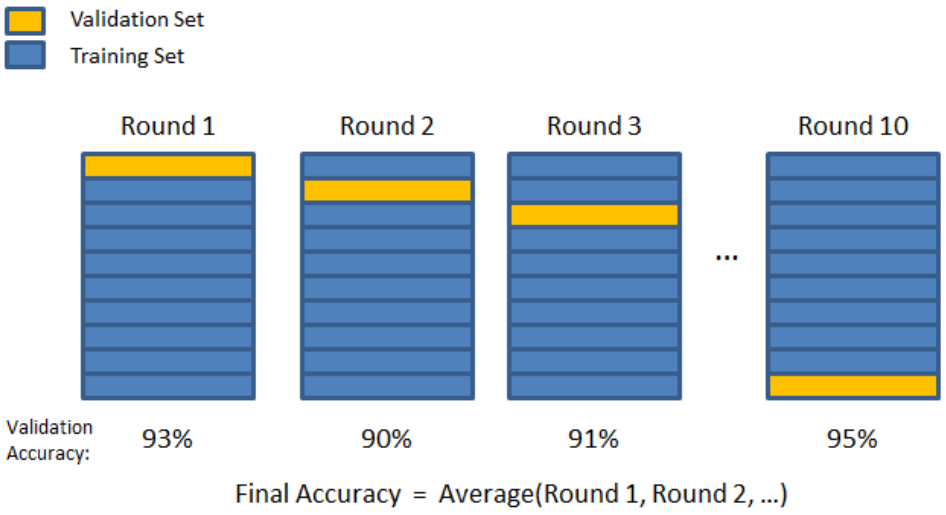
Evaluación: Métodos

Validación cruzada
[k-CV: k-fold Cross-Validation]

- Se divide aleatoriamente el conjunto de datos en k subconjuntos de intersección vacía (más o menos del mismo tamaño). Típicamente, k=10.
- En la iteración i, se usa el subconjunto i como conjunto de prueba y los k-1 restantes como conjunto de entrenamiento.
- Como medida de evaluación del método de clasificación se toma la media aritmética de las k iteraciones realizadas.

Evaluación: Métodos

Validación cruzada
[k-CV: k-fold Cross-Validation]



https://chrisjmccormick.wordpress.com/2013/07/31/k-fold-cross-validation-with-matlab-code/

Evaluación: Métodos

Validación cruzada

Variantes de la validación cruzada

- **"Leave one out":**
  Se realiza una validación cruzada con k particiones del conjunto de datos, donde k coincide con el número de ejemplos disponibles.

- **Validación cruzada estratificada:**
  Las particiones se realizan intentando mantener en todas ellas la misma proporción de clases que aparece en el conjunto de datos completo.

Evaluación: Métodos

Bootstraping

Muestreo uniforme con reemplazo de los ejemplos disponibles (esto es, una vez que se escoge un ejemplo, se vuelve a dejar en el conjunto de entrenamiento y puede que se vuelva a escoger).

NOTA: Método utilizado en "ensembles".

Evaluación: Métodos

Bootstraping

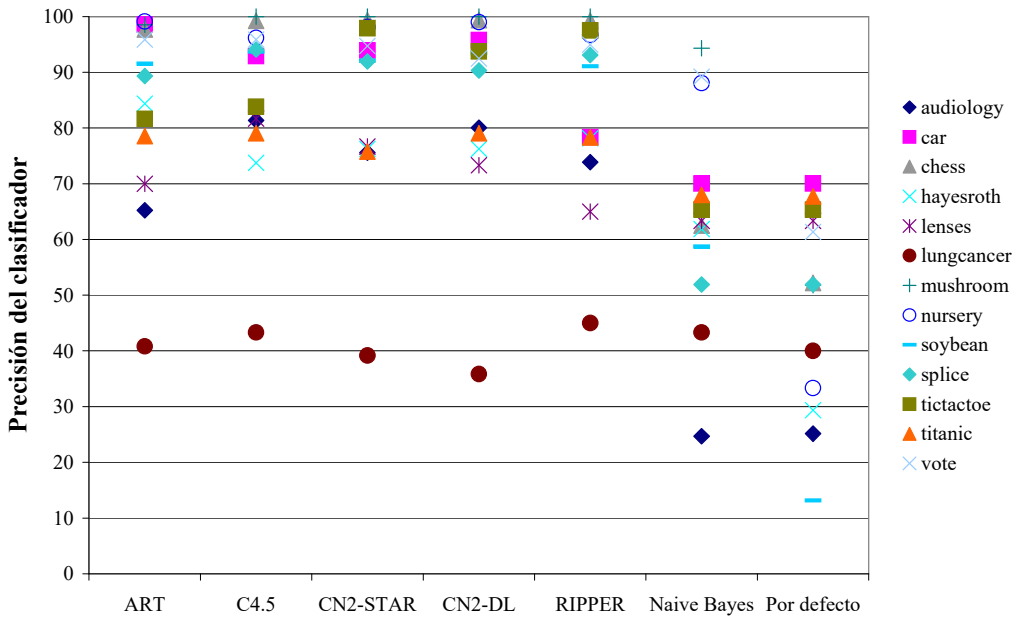
0.632 bootstrap

- Dado un conjunto de d datos, se toman d muestras. Los datos que no se escojan formarán parte del conjunto de prueba.
- En torno al 63.2% de las muestras estarán en el "bootstrap" (el conjunto de entrenamiento) y el 36.8% caerá en el conjunto de prueba, ya que $(1-1/d)^d \approx e^{-1} = 0.368$
- Si repetimos el proceso k veces, tendremos:

$$acc(M) = \frac{1}{k}\sum_{i=1}^{k}(0.632 \times acc(M_i)_{test\_set} + 0.368 \times acc(M_i)_{train\_set})$$

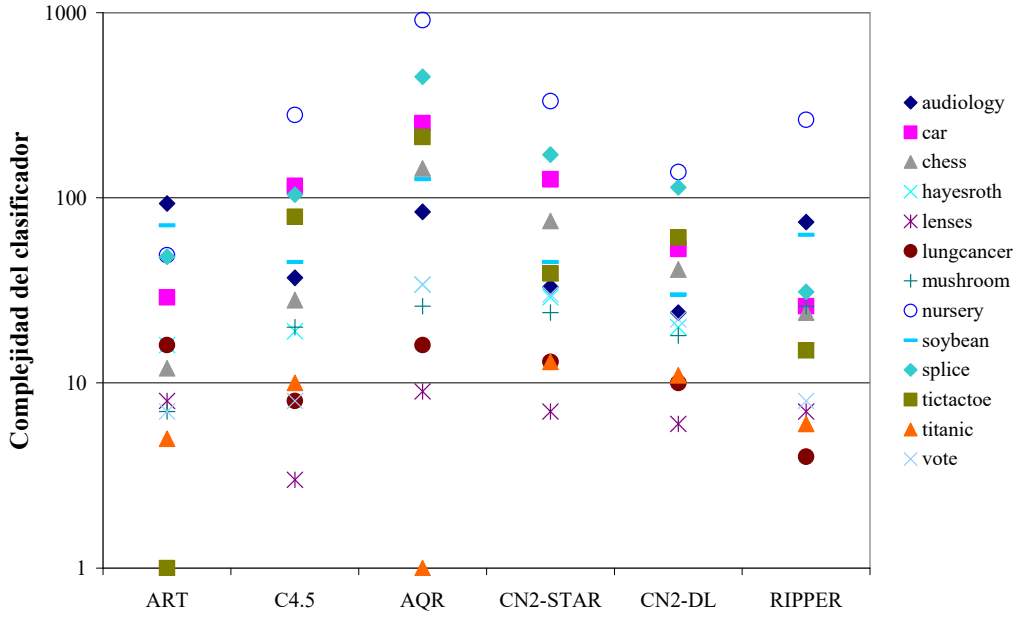
Evaluación: Comparación

Precisión [accuracy]



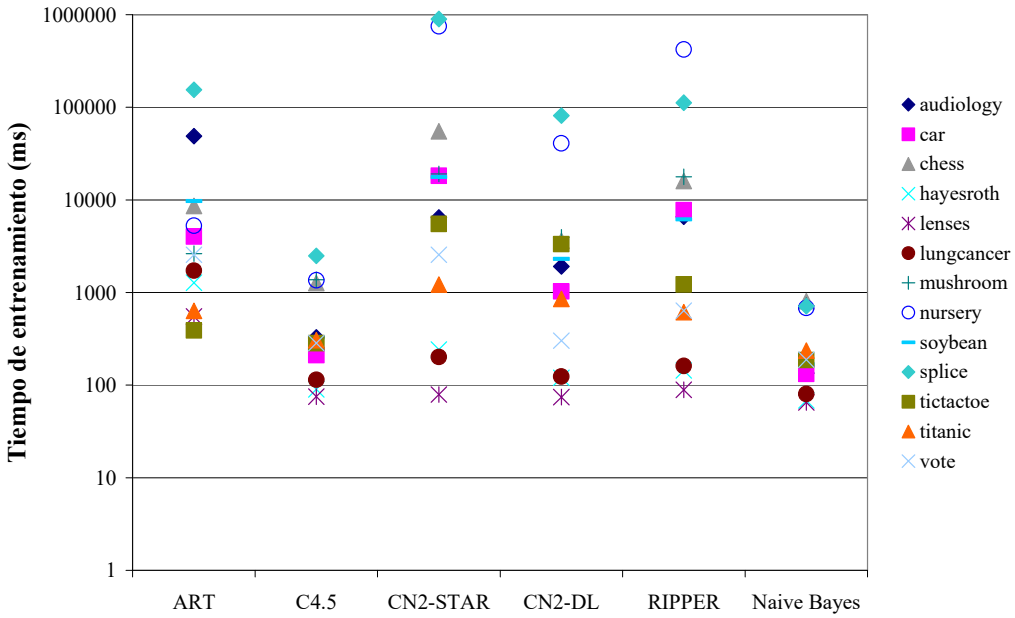
Evaluación: Comparación

Complejidad del clasificador



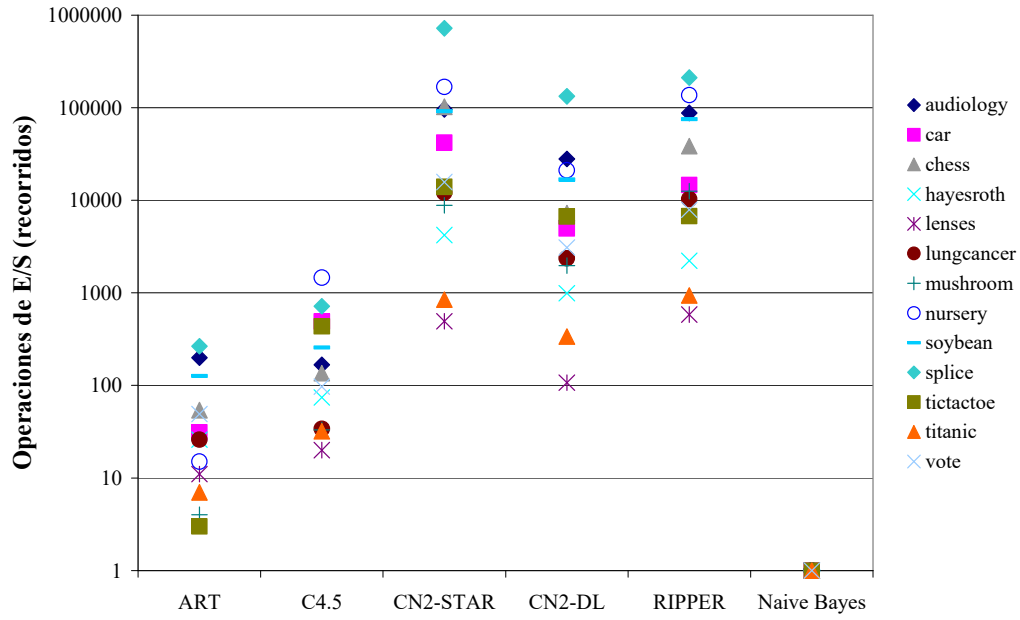
Evaluación: Comparación

Tiempo de entrenamiento



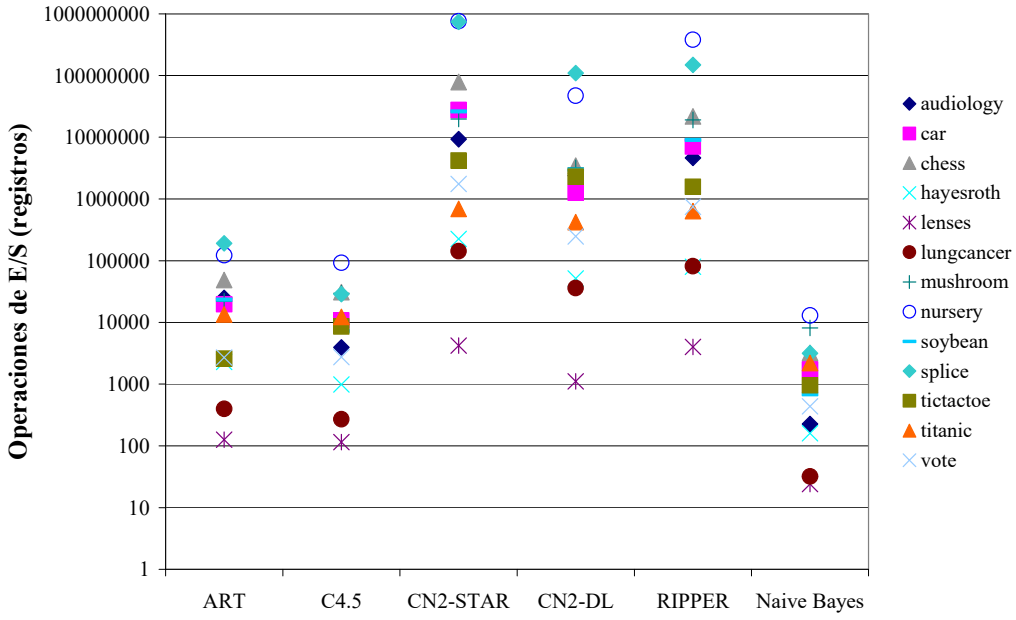
Evaluación: Comparación

Operaciones de E/S: Recorridos



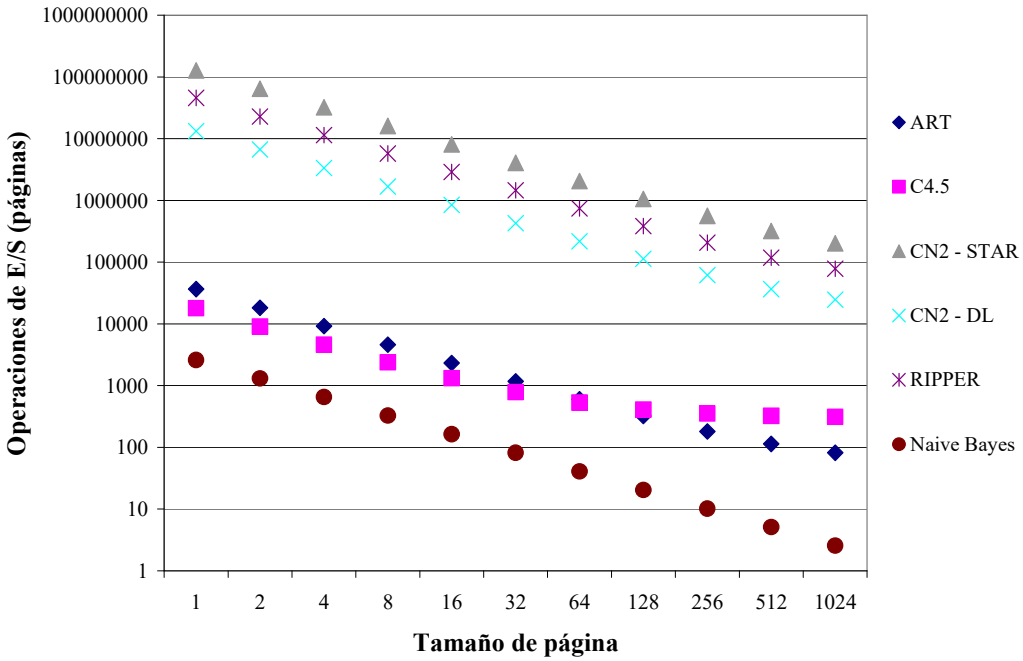
Evaluación: Comparación

Operaciones de E/S: Registros



Evaluación: Comparación

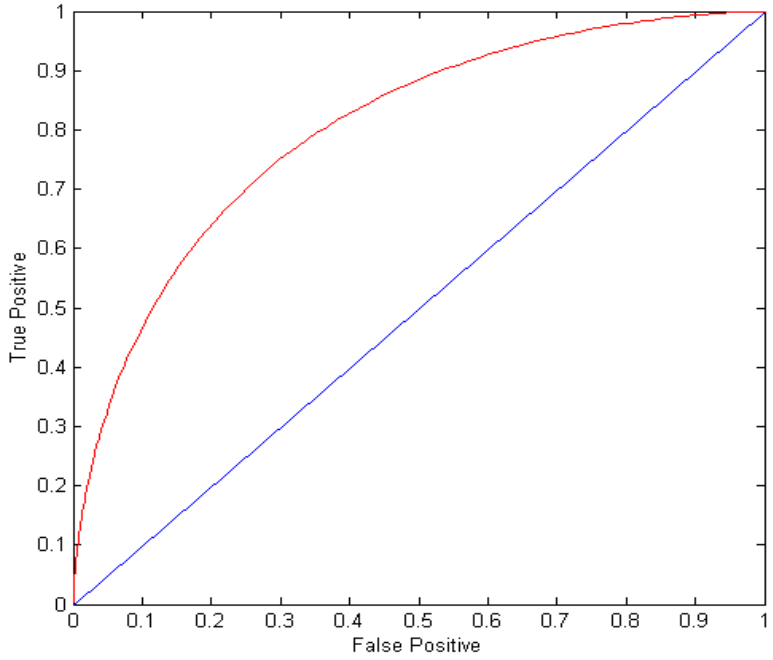
Operaciones de E/S: Páginas de disco



Evaluación: Comparación

Curvas ROC

Receiver Operating Characteristics



TPR = TP/(TP+FN) Eje vertical: "true positive rate"

FPR = FP/(FP+TN) Eje horizontal: "false positive rate"

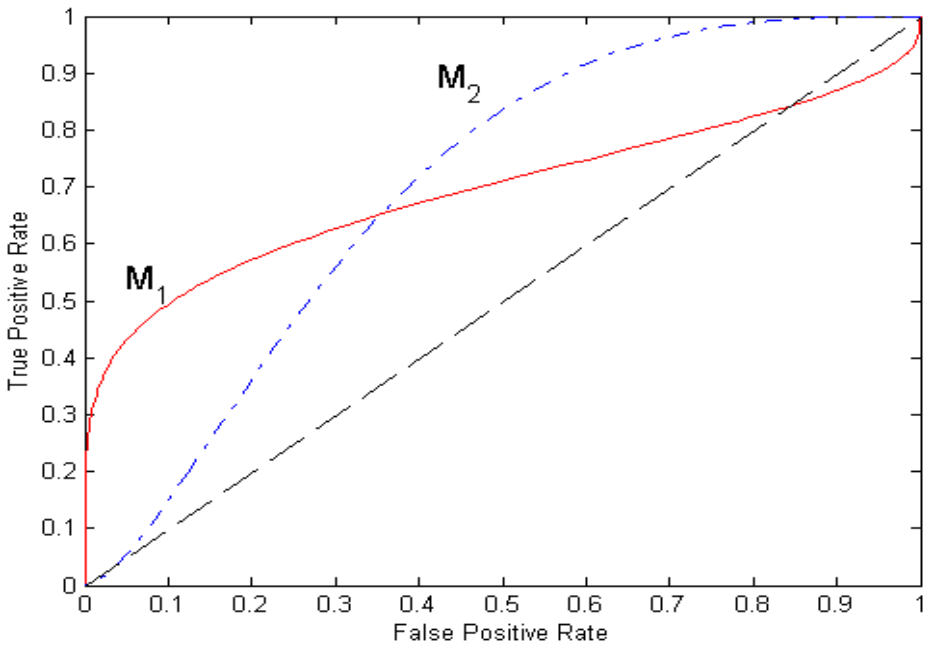
Evaluación: Comparación

Curvas ROC

- Desarrolladas en los años 50 para analizar señales con ruido: caracterizar el compromiso entre aciertos y falsas alarmas.
- El área que queda bajo la curva [**AUC**] es una medida de la precisión [accuracy] del clasificador:
  - Cuanto más cerca estemos de la diagonal (área cercana a 0.5), menos preciso será el modelo.
  - Un modelo "perfecto" tendrá área 1.
- Permiten comparar visualmente distintos modelos de clasificación…

Evaluación: Comparación

Curvas ROC



Ningún modelo es consistentemente mejor que el otro: M_1 es mejor para FPR bajos, M_2 para FPR altos.

Evaluación: Comparación

Curvas ROC

¿Cómo se construye la curva ROC?

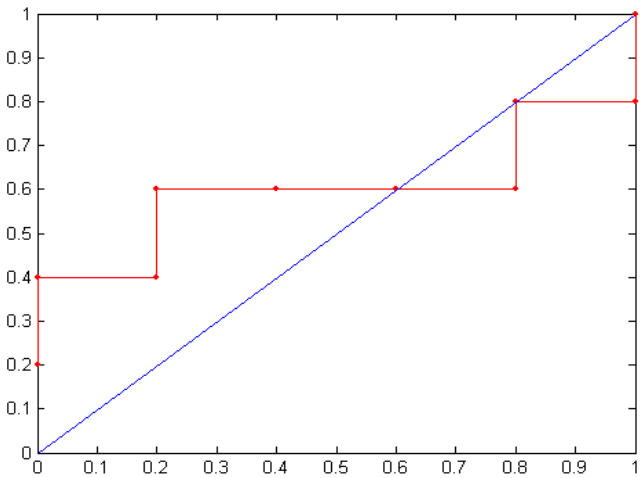
- Se usa un clasificador que prediga la probabilidad de que un ejemplo E pertenezca a la clase positiva P(+|E)
- Se ordenan los ejemplos en orden decreciente del valor estimado P(+|E)
- Se aplica un umbral para cada valor distinto de P(+|E), para el que se cuenta el número de TP, FP, TN y FN.

TPR = TP/(TP+FN)

FPR = FP/(FP+TN)

Evaluación: Comparación

Curvas ROC

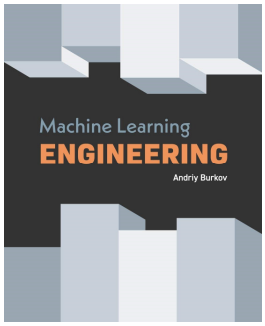


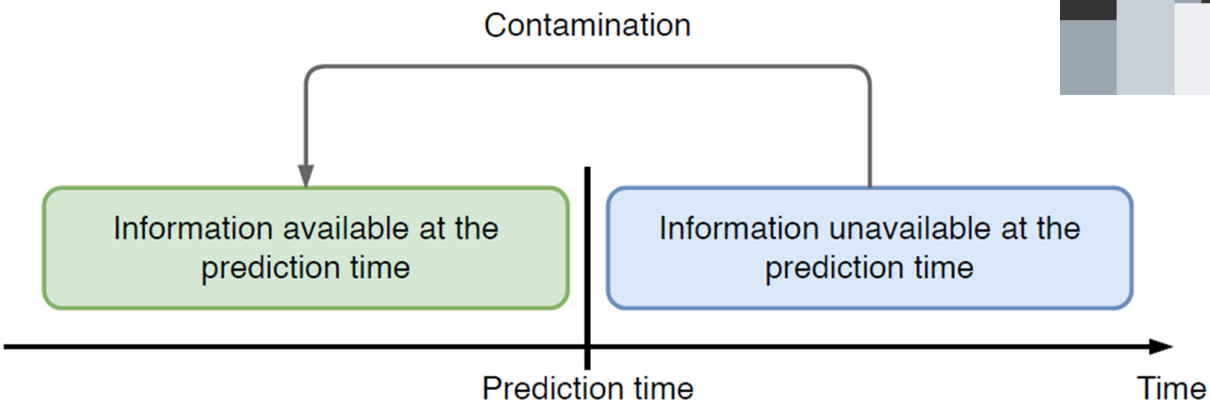
Ejemplo	P(+|E)	Clase
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

Clase	+	-	+	-	-	-	+	-	+	+	
	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

Evaluación

Cuidado con la posible contaminación de los datos [a.k.a. data leakage]

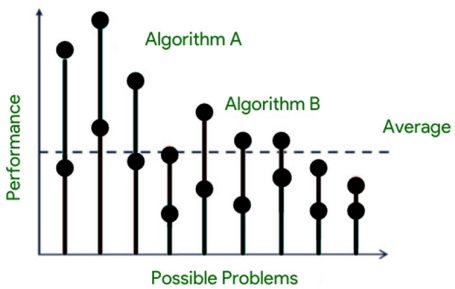




Evaluación

Teorema de Wolpert

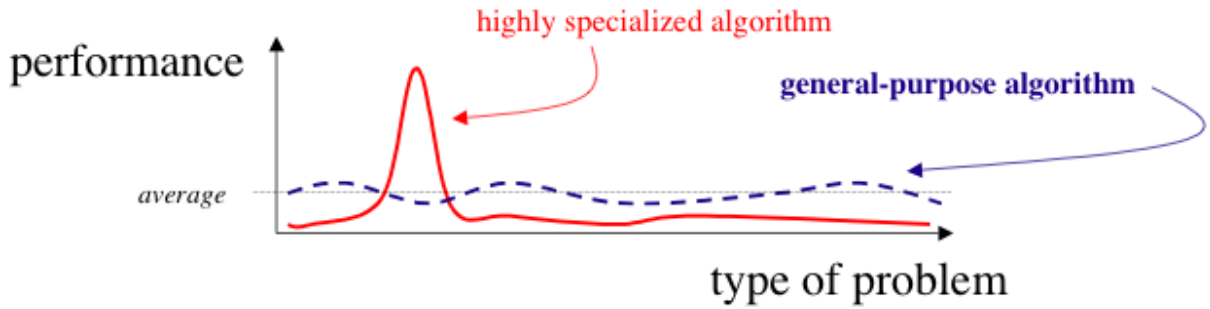
a.k.a. "No free lunch" theorem



Evaluación

Teorema de Wolpert

a.k.a. "No free lunch" theorem



No existe un algoritmo que sea siempre mejor que otro…

Ajuste de hiperparámetros

Hiperparámetros (o meta-parámetros)

Una de las principales dificultades prácticas del uso de técnicas de aprendizaje automático es la destreza que requiere establecer todos sus parámetros:

- **Parámetros** del modelo
  (los parámetros que ajusta el algoritmo de aprendizaje)

- **Hiperparámetros** del modelo
  (los parámetros que ajusta el que decide cómo ejecutar el algoritmo de aprendizaje).

Ajuste de hiperparámetros

Hiperparámetros (o meta-parámetros)

¿Cómo elegir los hiperparámetros de un modelo?

Método incorrecto: Se prueban montones de alternativas para ver cuál funciona mejor en el conjunto de test.

- Fácil de hacer, pero nos da una impresión engañosa de lo bien que funcionará la red en la práctica:
- La configuración que funcione mejor sobre el conjunto de prueba puede que no sea la que funcione mejor en otros conjuntos de prueba (o los nuevos casos sobre los que queramos aplicar la red neuronal).

Ajuste de hiperparámetros

Hiperparámetros (o meta-parámetros)

¿Cómo elegir los hiperparámetros de un modelo?

Un método mejor: **Conjunto de validación.**

Se divide el conjunto de datos disponible en tres partes:

- Conjunto de **entrenamiento** (para aprender los parámetros del modelo, e.g. pesos de una red neuronal).
- Conjunto de **validación** (no se utiliza en el entrenamiento, sino para decidir qué hiperparámetros resultan más adecuados)
- Conjunto de **prueba** (para obtener una estimación no sesgada de lo bien que funciona el modelo).

Ajuste de hiperparámetros

Hiperparámetros (o meta-parámetros)

¿Cómo elegir los hiperparámetros de un modelo?

Validación cruzada

- Dividimos el conjunto de datos en N subconjuntos.
- Utilizamos N-1 subconjuntos de conjunto de entrenamiento y el subconjunto restante de conjunto de prueba para obtener N estimaciones del error.

Ajuste de hiperparámetros

Hiperparámetros (o meta-parámetros)

¿Cómo elegir los hiperparámetros de un modelo?

AutoML

Aprendizaje automático [Machine Learning]

En vez de probar todas las combinaciones posibles de parámetros, podemos muestrear el espacio de posibles combinaciones.

p.ej. Metaheurísticas (algoritmos genéticos)

Optimización bayesiana (procesos gaussianos)

Ajuste de hiperparámetros

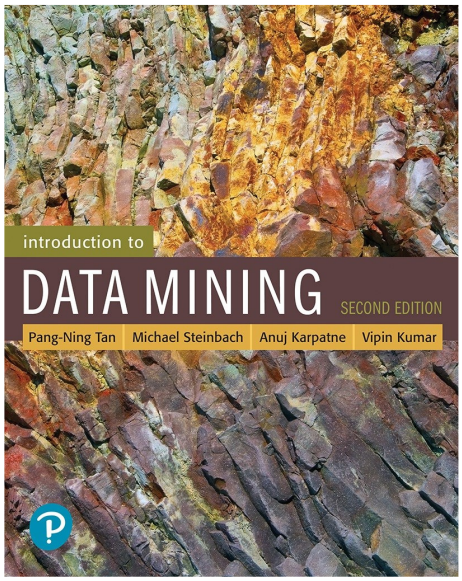
Hiperparámetros (o meta-parámetros)
¿Cómo elegir los hiperparámetros de un modelo?

AutoML
Aprendizaje automático [Machine Learning]

- Mucho mejor que ir haciendo pruebas manualmente (no es el tipo de tarea que los humanos hacemos bien).
- Evita sesgos psicológicos no deseados: método menos propenso a funcionar mejor con el método que nos gusta y peor con el que no (las personas no podemos evitarlo ;-)

Bibliografía

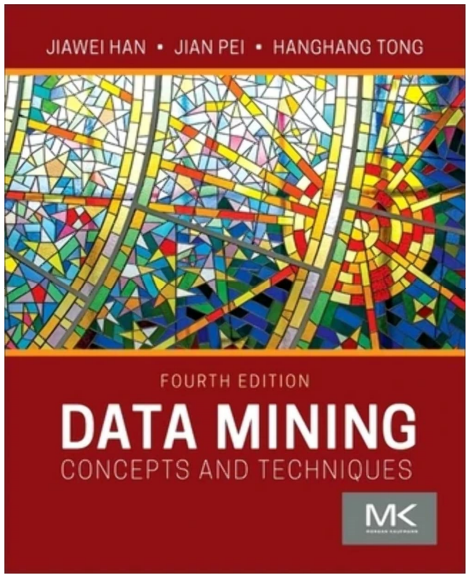
Pang-Ning Tan,
Michael Steinbach,
Vipin Kumar &
Anuj Karpatne:
Introduction to Data Mining,
2nd edition, Addison Wesley, 2018.
ISBN 0133128903



3 Classification: Basic Concepts and Techniques
6 Classification: Alternative Techniques
10.3 Statistical Testing for Classification

Bibliografía

Jiawei Han,
Jian Pei &
Hanghang Tong:
Data Mining:
Concepts and Techniques,
4th edition, Morgan Kaufmann, 2022.
ISBN 0128117605



6 Classification: Basic concepts and methods
7 Classification: Advanced methods

Bibliografía

- F. Berzal, J.C. Cubero, D. Sánchez & J.M. Serrano: **ART: A hybrid classification method**. Machine Learning, 2004
- L. Breiman, J. Friedman, R. Olshen & C. Stone: **Classification and Regression Trees**. Wadsworth International Group, 1984.
- W. Cohen: **Fast effective rule induction**. ICML'95
- R. O. Duda, P. E. Hart & D. G. Stork: **Pattern Classification**, 2ed. John Wiley and Sons, 2001
- U. M. Fayyad: **Branching on attribute values in decision tree generation**. AAAI'94
- Y. Freund & R. E. Schapire: **A decision-theoretic generalization of on-line learning and an application to boosting**. J. Computer and System Sciences, 1997.
- J. Gehrke, V. Gant, R. Ramakrishnan & W.-Y. Loh: **BOAT -- Optimistic Decision Tree Construction**. SIGMOD'99.
- J. Gehrke, R. Ramakrishnan & V. Ganti: **Rainforest: A framework for fast decision tree construction of large datasets**. VLDB'98.

Bibliografía

- T.-S. Lim, W.-Y. Loh & Y.-S. Shih: **A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms.** Machine Learning, 2000.
- S. K. Murthy: **Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey**, Data Mining and Knowledge Discovery 2(4): 345-389, 1998
- J. R. Quinlan: **Induction of decision trees**. *Machine Learning*, 1:81-106, 1986.
- J. R. Quinlan & R. M. Cameron-Jones: **FOIL: A midterm report**. ECML'93.
- J. R. Quinlan:
  C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.
- J. R. Quinlan: **Bagging, boosting, and c4.5**. AAAI'96.
- R. Rastogi & K. Shim: **Public: A decision tree classifier that integrates building and pruning**. VLDB'98
- H. Yu, J. Yang & J. Han: **Classifying large data sets using SVM with hierarchical clusters**. KDD'03.

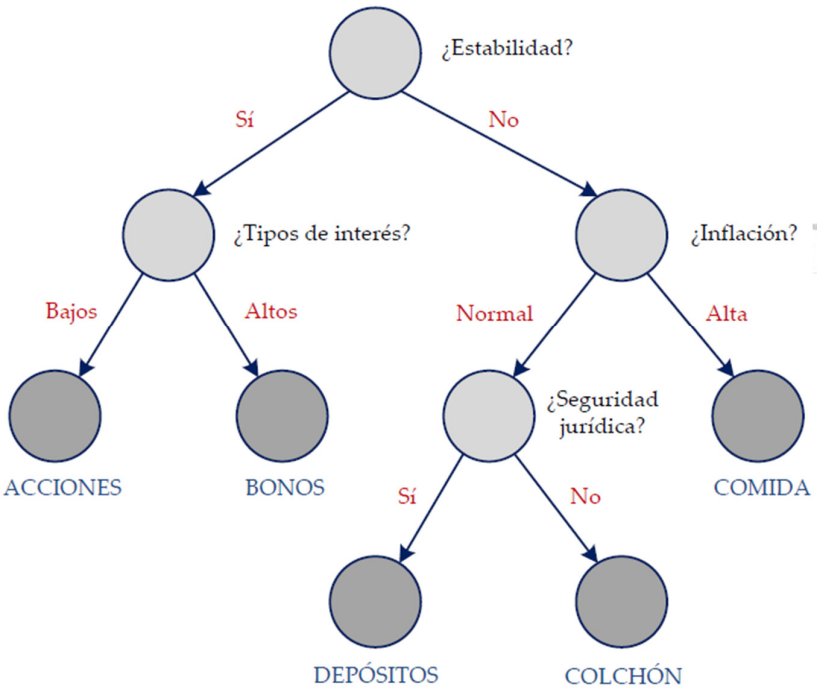
Apéndice

Modelos de clasificación

- Modelos simbólicos
  - Árboles de decisión
  - Inducción de reglas (p.ej. listas de decisión)
- Modelos "estadísticos"
  - Clasificadores paramétricos
  - Modelos bayesianos, p.ej. redes bayesianas
- Modelos analógicos
  - Clasificadores basados en casos
  - SVMs (Support Vector Machines)
- Modelos conexionistas: Redes neuronales

Modelos de clasificación

Modelos simbólicos: Árboles de decisión



Modelos de clasificación

Modelos basados en reglas de asociación

¿Por qué?

Buscando entre las mejores reglas de asociación, se superan algunas limitaciones de los árboles de decisión (p.ej. sólo consideran los atributos de uno en uno).

Apéndice

Modelos de clasificación

Modelos basados en reglas de asociación

- Modelos de clasificación parcial
  Bayardo, KDD'1997
- Modelos "asociativos" de clasificación
  CBA (Liu, Hsu & Ma, KDD'1998)
  RCBT (Cong et al., SIGMOD'2005)
- Patrones emergentes
  CAEP (Dong et al., ICDS'1999)
- Árboles de reglas
  Wang et al., KDD'2000
- Reglas con excepciones
  Liu et al., AAAI'2000

Apéndice

Modelos de clasificación

Modelos basados en reglas de asociación

- CMAR
  Classification based on Multiple Association Rules
  Li, Han & Pei, ICDM'2001
- CPAR
  Classification based on Predictive Association Rules
  Yin & Han, SDM'2003
- ART
  Association Rule Trees
  Berzal et al., Machine Learning, 2004

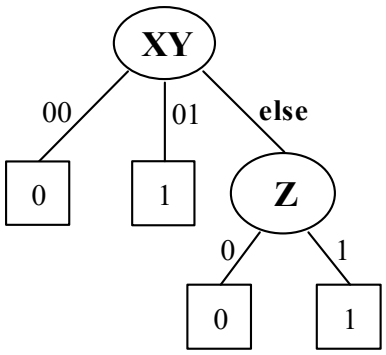
Modelos basados en reglas de asociación

ART [Association Rule Trees]

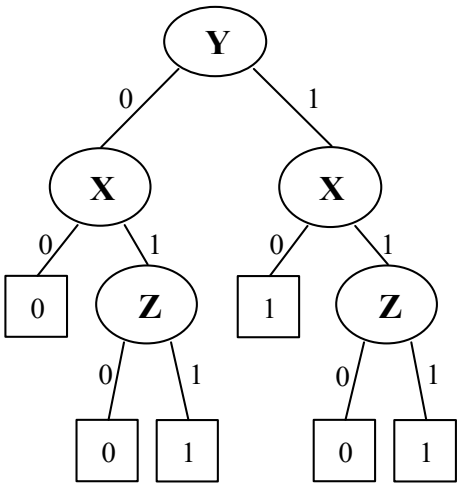

```
P30 = A : TYPE = N (473|62)
P30 = C : TYPE = N (441|24)
P30 = T : TYPE = N (447|57)
else
  P28 = A and P32 = T : TYPE = EI (235|33)
  P28 = G and P32 = T : TYPE = EI (130|20)
  P28 = C and P32 = A : TYPE = IE (160|31)
  P28 = C and P32 = C : TYPE = IE (167|35)
  P28 = C and P32 = G : TYPE = IE (179|36)
  else
    P28 = A : TYPE = N (106|14)
    P28 = G : TYPE = N (94|4)
    else
      P29 = C and P31 = G : TYPE = EI (40|5)
      P29 = A and P31 = A : TYPE = IE (86|4)
      P29 = A and P31 = C : TYPE = IE (61|4)
      P29 = A and P31 = T : TYPE = IE (39|1)
      else
        P25 = A and P35 = G : TYPE = EI (54|5)
        P25 = G and P35 = G : TYPE = EI (63|7)
        else
          P23 = G and P35 = G : TYPE = EI (40|8)
          P23 = T and P35 = C : TYPE = IE (37|7)
          else
            P21 = G and P34 = A : TYPE = EI (41|5)
            else
              P28 = T and P29 = A : TYPE = IE (66|8)
              else
                P31 = G and P33 = A : TYPE = EI (62|9)
                else
                  P28 = T : TYPE = N (49|6)
                  else
                    P24 = C and P29 = A : TYPE = IE (39|8)
                    else
                      TYPE = IE (66|39)
```


Modelos basados en reglas de asociación

ART



TDIDT



DEMO

ART
Association Rule Trees

Clasificadores bayesianos

Idea básica

Si no puede asegurarse a qué clase pertenece una instancia, se le asigna la clase a la que tiene mayor probabilidad de pertenecer.

Dada una instancia,
¿cómo se estima la clase más probable?
Teorema de Bayes

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

Clasificadores bayesianos

Idea básica

¿Cómo se determinan las clases más probables de forma eficiente, i.e. P(X|y)? Hipótesis simplificadoras:

- Independencia entre atributos:
  Métodos **Naive Bayes**.
- Distribución normal conjunta para atributos continuos:
  Análisis discriminante y sus variantes.
- Dependencias conocidas entre variables:
  Redes bayesianas.

Apéndice
Modelos de clasificación

Clasificadores bayesianos

Naïve Bayes

Aplicando el Teorema de Bayes, se maximiza:

$$P(y_k|\mathbf{X}) = P(\mathbf{X}|y_k)P(y_k)$$

$$P(\mathbf{X}|y_k) = \prod_{i=1}^{n} P(x_i|y_k) = P(x_1|y_k) \times P(x_2|y_k) \times \cdots \times P(x_n|y_k)$$

Ventaja

- Basta con recorrer los datos una sola vez.

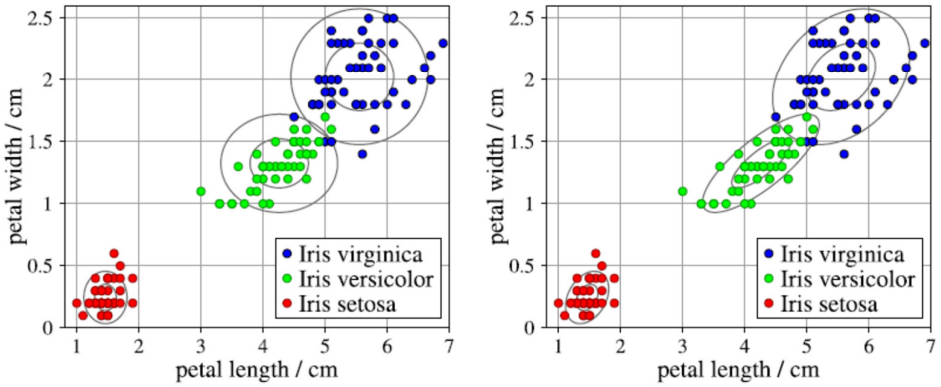
Desventajas

- Interpretabilidad del modelo.
- Supone que las variables son independientes.

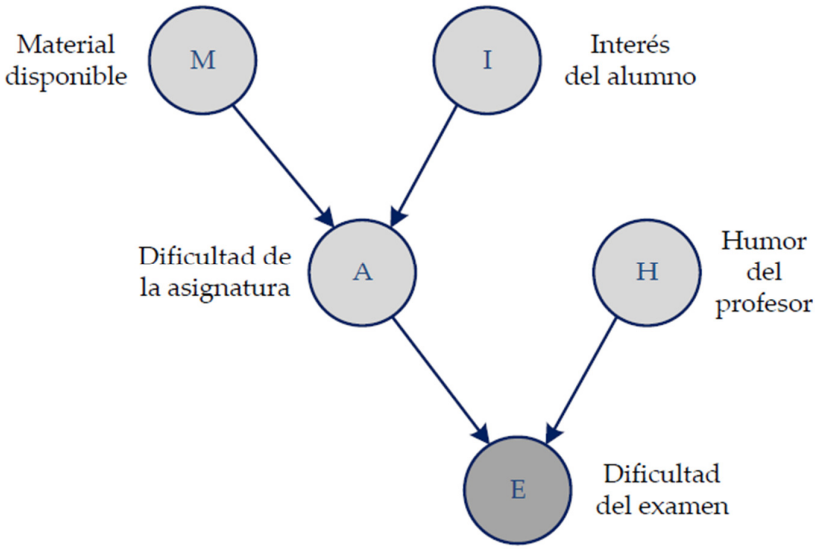
Clasificadores bayesianos

con valores numéricos (hipótesis de normalidad)

Iris type	Iris setosa	Iris versicolor	Iris virginica
Prior probability	0.333	0.333	0.333
Petal length	1.46 ± 0.17	4.26 ± 0.46	5.55 ± 0.55
Petal width	0.24 ± 0.11	1.33 ± 0.20	2.03 ± 0.27



Redes bayesianas



Modelos de clasificación

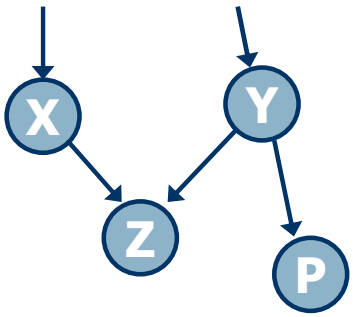
Clasificadores bayesianos

Redes Bayesianas

Representan mediante un grafo dirigido acíclico dependencias entre variables, especificando sus distribuciones de probabilidad conjuntas.

Nodos: Variables

Enlaces: Dependencias



Apéndice

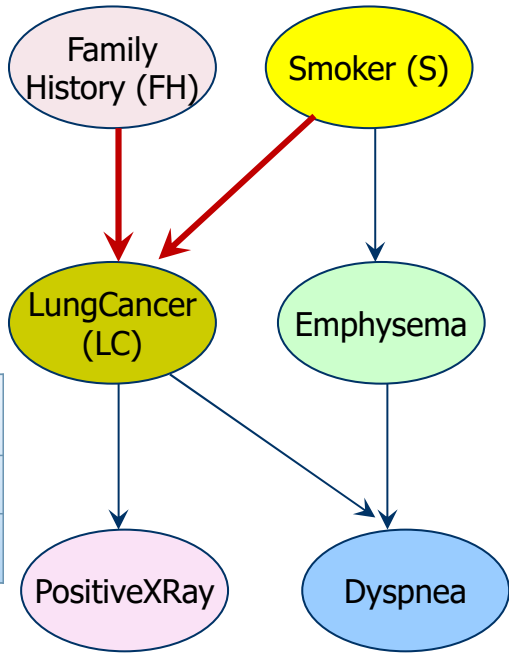
Modelos de clasificación

Clasificadores bayesianos

Redes Bayesianas

CPT [Conditional Probability Table] para la variable LungCancer:

P(LC|...)	(FH,S)	(FH, ~S)	(~FH,S)	(~FH, ~S)
LC	0.8	0.5	0.7	0.1
~ LC	0.2	0.5	0.3	0.9



Muestra la probabilidad condicional de que alguien desarrolle cáncer de pulmón para combinación de las variables que lo "causan".

Apéndice

Modelos de clasificación

Clasificadores bayesianos

Redes Bayesianas

Entrenamiento de las redes bayesianas:

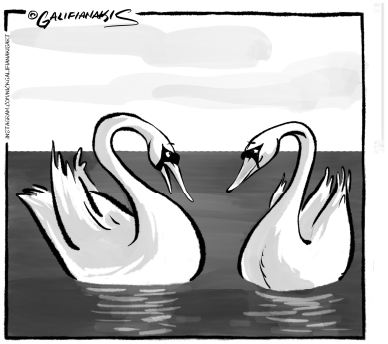
- Dada la estructura de la red, calcular CPTs (sencillo, como en Naïve Bayes).
- Dada la estructura de la red, con algunas variables "ocultas" (desconocidas), buscar una configuración adecuada de la red que encaje con nuestro conjunto de entrenamiento (usando técnicas de optimización como el gradiente descendente).
- Dadas las variables observables, determinar la topología óptima de la red (muy ineficiente).

Apéndice

Modelos de clasificación

Clasificadores basados en casos [lazy learners]

Almacenan todo el conjunto de entrenamiento (o parte de él) y lo utilizan directamente para clasificar nuevos datos buscando los casos de entrenamiento más parecidos.



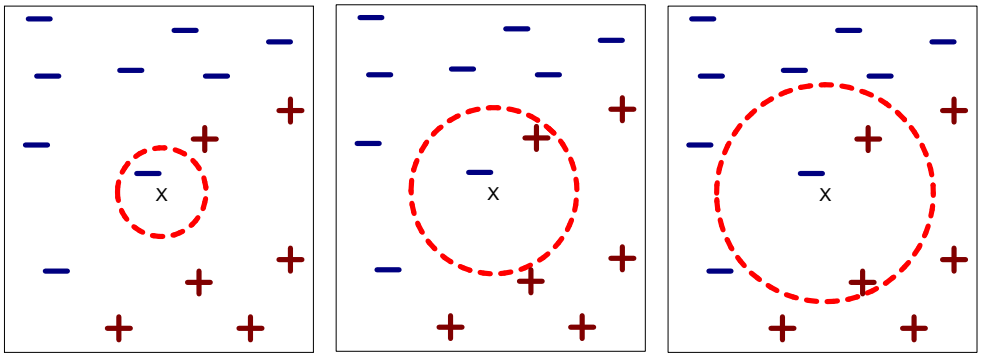
Ejemplos

- k-NN (k Nearest Neighbors)
- Razonamiento basado en casos (CBR)

Modelos de clasificación

Clasificadores basados en casos

k-NN



(a) 1-nearest neighb□r (b) 2-nearest neighb□r (c) 3-nearest neighb□r

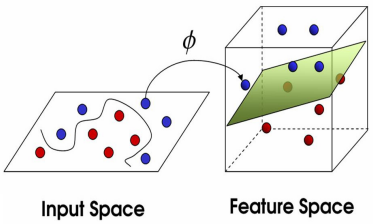
Se escoge la clase más común entre los k vecinos más cercanos:

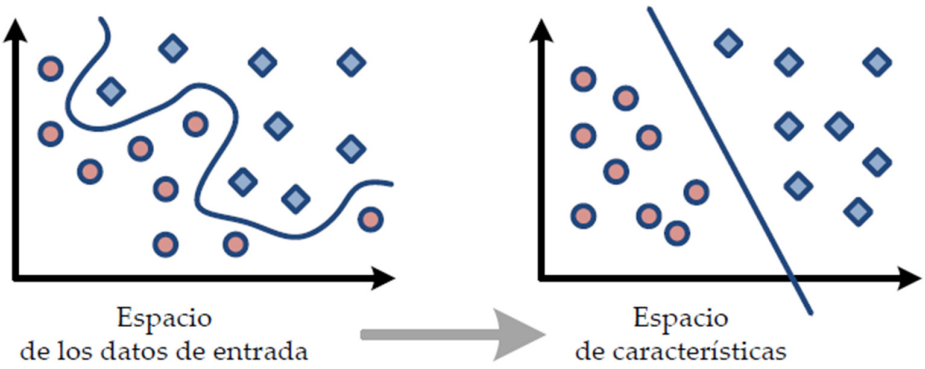
- k demasiado pequeño
  → Sensible a ruido.
- k demasiado grande
  → El vecindario puede incluir puntos de otras clases.

Apéndice

Modelos de clasificación

SVMs [Support Vector Machines]

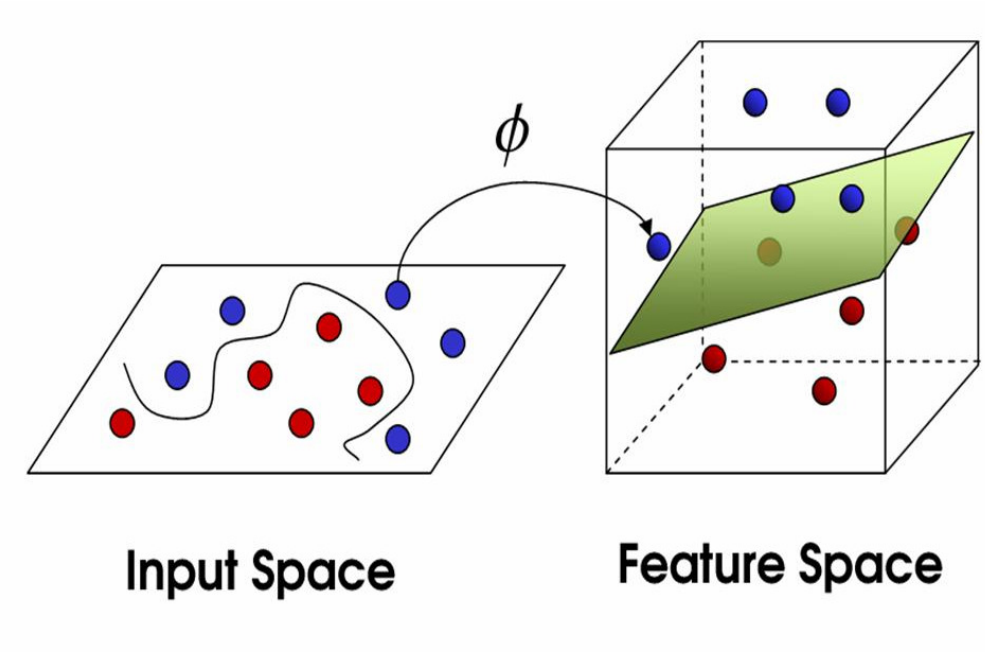




De moda hasta la llegada del Deep Learning…

Modelos de clasificación

SVMs [Support Vector Machines]

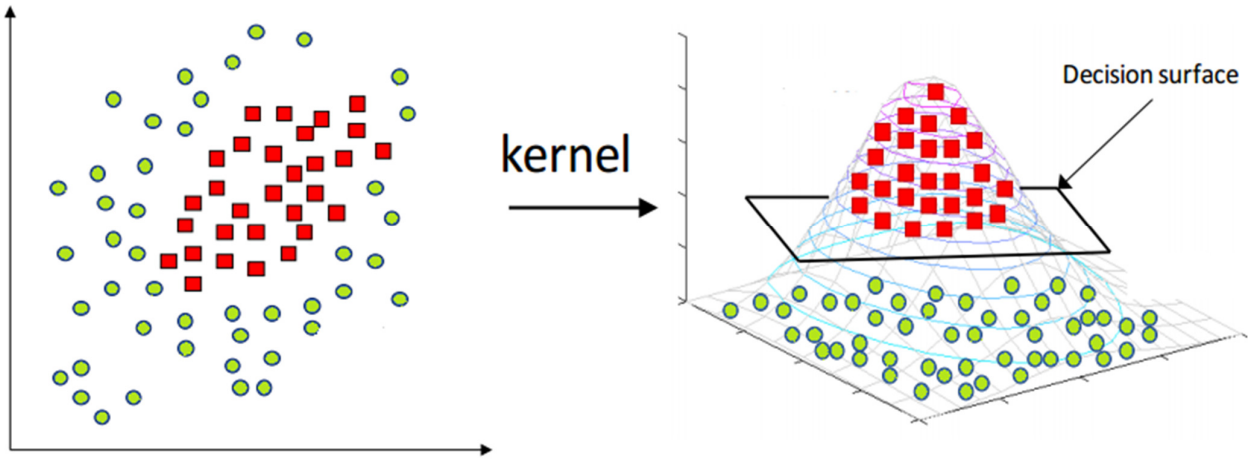


Apéndice

Modelos de clasificación

SVMs [Support Vector Machines]

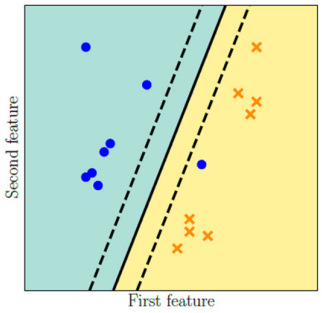
Proyectar puntos en más dimensiones
los hace linealmente separables…



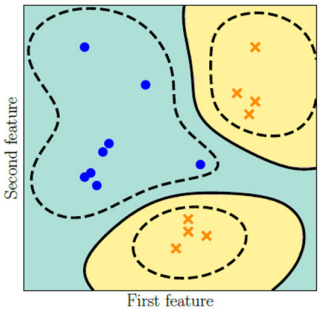
El truco del kernel: Se representan los datos mediante la similitud entre pares de observaciones **x**, en lugar de aplicar explícitamente las transformaciones $\phi(\mathbf{x})$.

SVMs [Support Vector Machines]

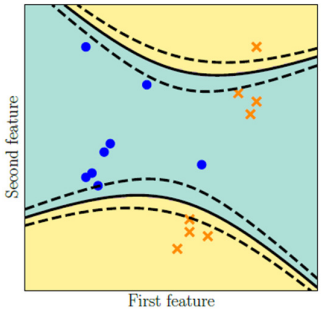
con distintos tipos de kernel



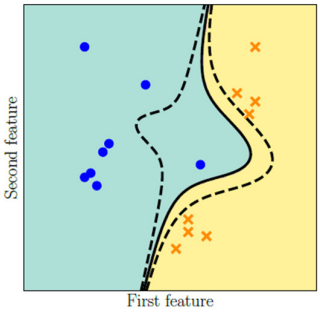
(a) SVM with linear kernel



(b) SVM with RBF kernel



(c) SVM with polynomial (degree 2) kernel

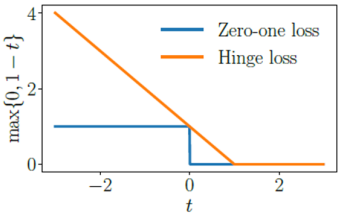


(d) SVM with polynomial (degree 3) kernel

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle - \sum_{i=1}^{N} \alpha_i$$

$$\text{subject to} \quad \sum_{i=1}^{N} y_i \alpha_i = 0$$

$$0 \leqslant \alpha_i \leqslant C \quad \text{for all} \quad i = 1, \ldots, N\,.$$



SVMs [Support Vector Machines]

Ventajas

- Precisión generalmente alta.
- Robustez frente a ruido.

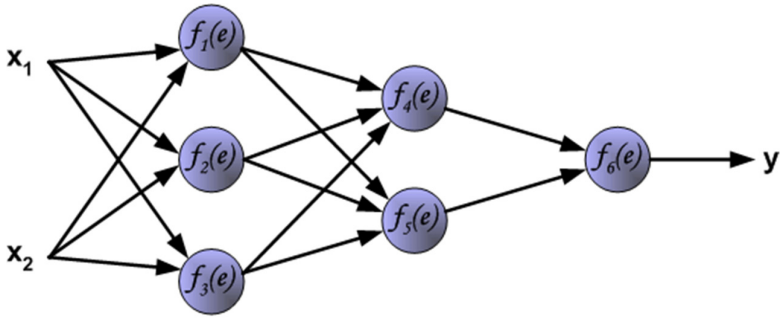
Desventajas

- Costosos de entrenar (eficiencia y escalabilidad).
- Difíciles de interpretar (basados en transformaciones matemáticas para conseguir que las clases sean linealmente separables).

Modelos de clasificación

Redes neuronales

p.ej. Backpropagation

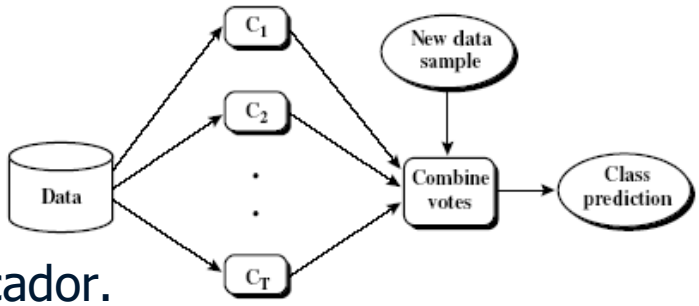


- Como "aproximadores universales", pueden aplicarse para predecir el valor de un atributo (tanto nominal como numérico).
- Ejemplo de modelo predictivo pero no descriptivo (podemos verlo como una caja negra).

Modelos de clasificación

Ensembles

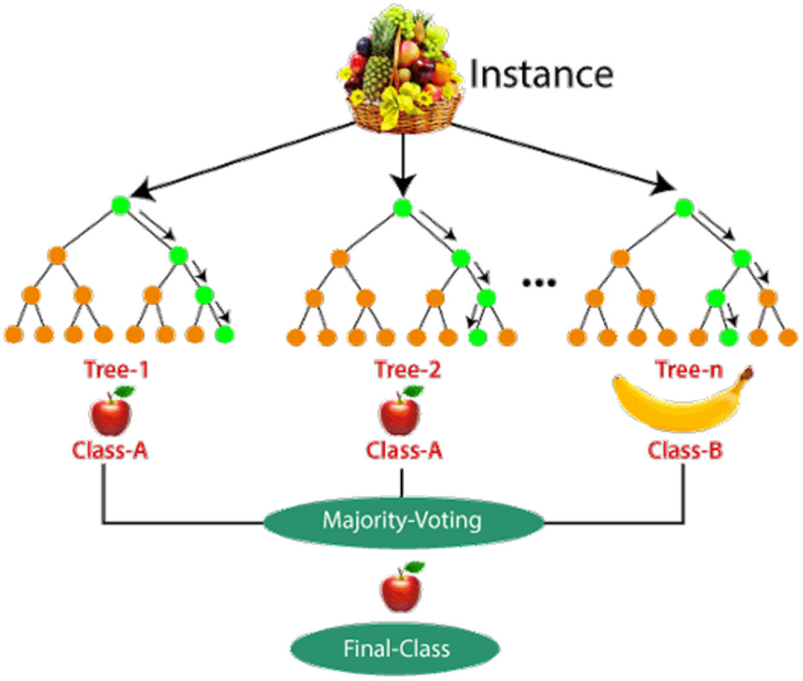
Combinan varios modelos con el objetivo de mejorar la precisión final del clasificador.

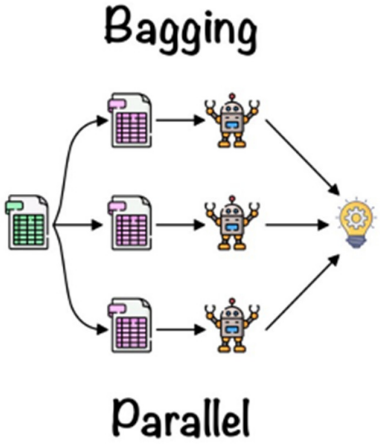


- **Bagging**: Votación por mayoría.
  Varios clasificadores diferentes votan para decidir la clase de un caso de prueba (usa bootstrapping).
- **Boosting**: Votación ponderada.
  Los clasificadores tienen distintos pesos en la votación (en función de su precisión), vg: AdaBoost.

Modelos de clasificación

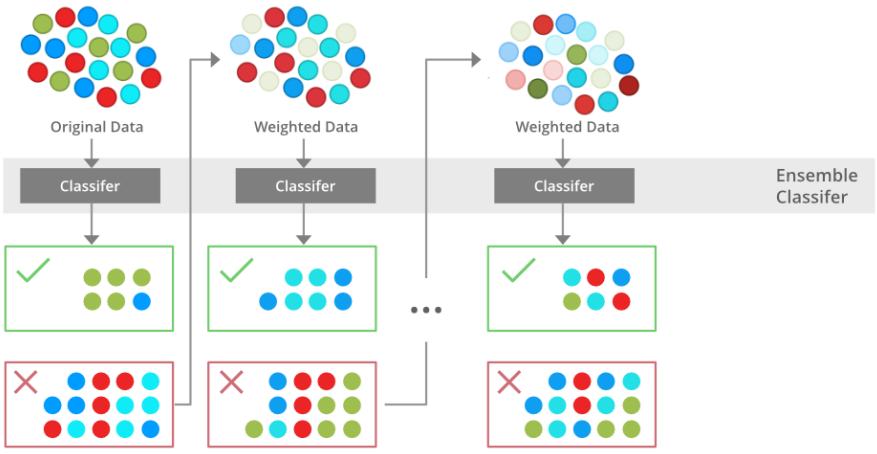
Ensembles: Bagging [bootstrap aggregating], e.g. Random Forests

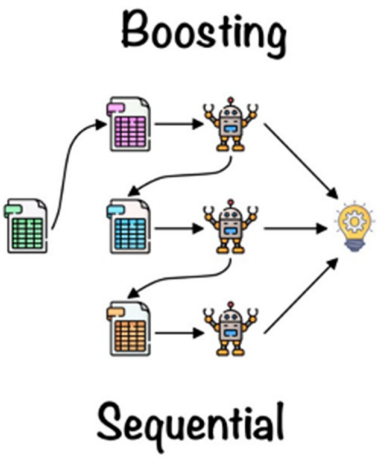


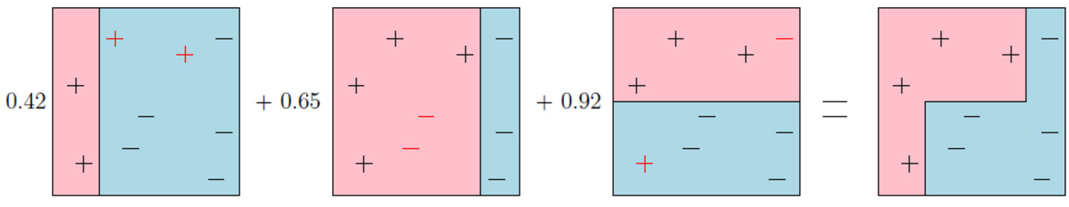


Modelos de clasificación

Ensembles: Boosting







Modelos de clasificación

Ensembles: Stacking