Técnicas de Inteligencia Artificial

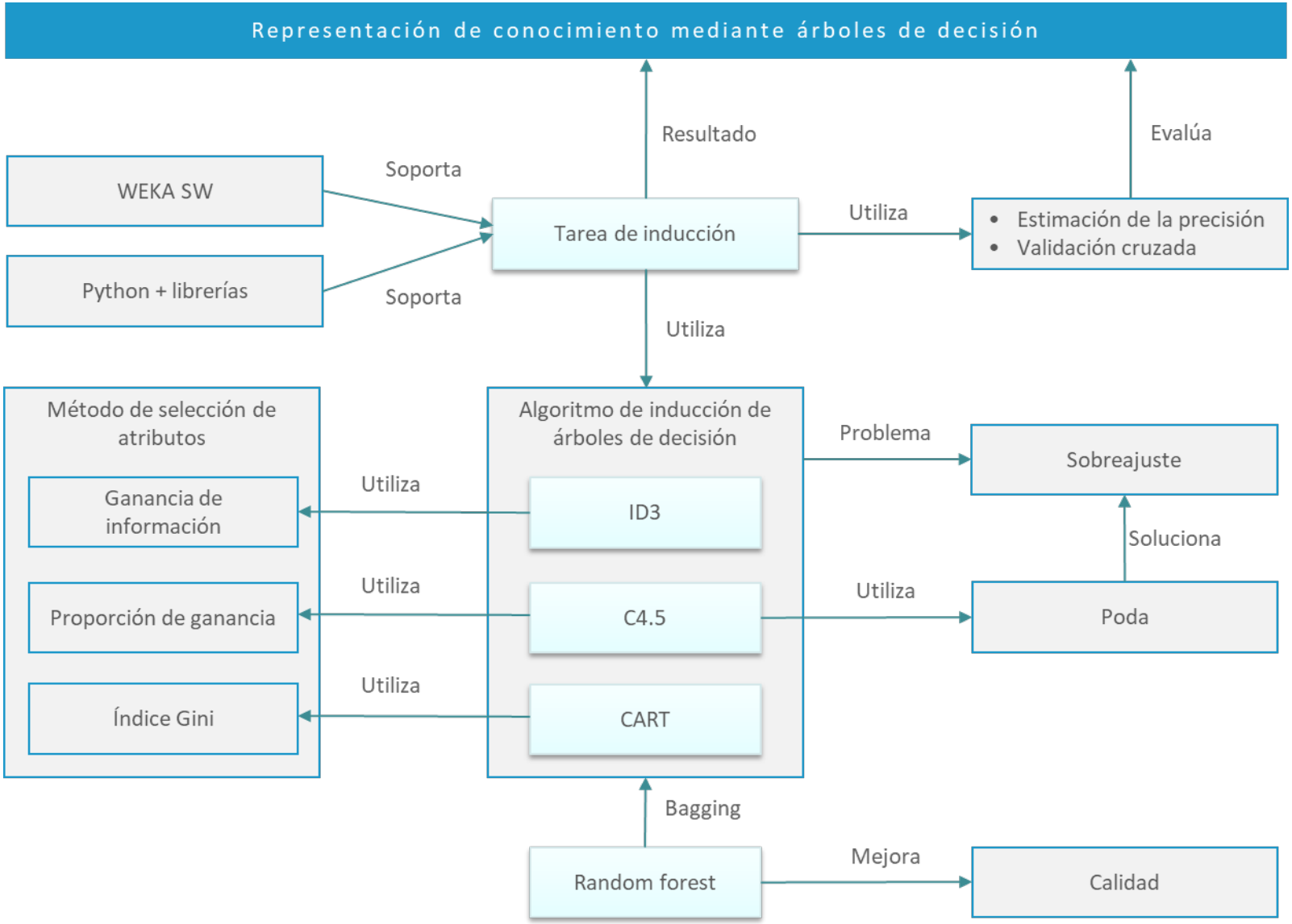
Óscar García

Tema 3. Árboles de Decisión

UNIVERSIDAD INTERNACIONAL DE LA RIOJA

La Universidad en Internet

¿Cómo estudiar este tema?



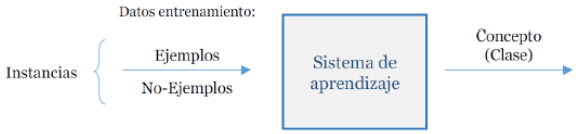
Introducción

Definición de aprendizaje automático

- **Un programa de ordenador aprende** de la experiencia E con respecto a una clase de tareas T y una medida de rendimiento P, **si su rendimiento en las tareas T, medido en base a la medida P, mejora con la experiencia E**. (Mitchell, 1997).

- *Aprender a detectar robos de tarjetas de crédito.*
  - *T: detectar robos de tarjetas de crédito.*
  - *P: porcentaje de robos detectados.*
  - *E: base de datos de hábitos de compra con la tarjeta de crédito.*

Elementos del aprendizaje de un concepto



- **Concepto (clase):** conjunto de objetos, símbolos o eventos agrupados porque comparten ciertas características y que pueden ser referenciados por un nombre en particular o un símbolo
- **Instancia:** ilustración específica de un objeto, símbolo, evento, proceso o procedimiento (Merrill, 1994)
- **Ejemplo:** instancia miembro del concepto en consideración
- **No-ejemplo:** instancia no-miembro del concepto en consideración
- **Datos de entrenamiento:** conjunto de instancias que forman parte de la experiencia que utiliza el sistema para aprender la tarea
- **Atributos:** características de las instancias

Aprendizaje de conceptos

- Tareas que resuelve el aprendizaje
  - Descripción de conceptos
    - en base a los atributos de ejemplos bien conocidos de una clase se aprender el concepto (esa clase)
  - Formación de nuevos conceptos
    - en base a unos ejemplos se quiere aprender un nuevo concepto desconocido que los describa

- Tipos de aprendizaje
  - Aprendizaje supervisado
    - Técnica para la descripción de conceptos
    - Pretende caracterizar un concepto a partir de instancias del mismo
  - Aprendizaje no-supervisado
    - Técnica para la formación de nuevos conceptos
    - Pretende caracterizar un concepto desconocido a partir de instancias del mismo

Aprendizaje Automático

- Aprendizaje supervisado (clase conocida)

ÁRBOLES DE DECISIÓN

Descripción del concepto	Clasificar instancias no clasificadas
Construcción de un modelo descriptivo a partir de un conjunto de instancias, conteniendo ejemplos y no-ejemplos del concepto a aprender.	Una vez construido el modelo podemos determinar si una instancia nuevamente encontrada se puede clasificar como ejemplo de esa clase.

Atributos — Clase

Datos de entrenamiento

Número de paciente	Fiebre	Dolor de garganta	Congestión	Dolor de cabeza	Diagnóstico
1	Sí	Sí	No	Sí	Infección de garganta
2	No	No	Sí	No	Alergia
3	No	No	Sí	Sí	Resfriado
4	No	No	Sí	No	Alergia
5	Sí	Sí	Sí	Sí	Infección de garganta
6	No	No	Sí	No	Resfriado
7	Sí	No	Sí	Sí	Resfriado
8	Sí	Sí	No	Sí	Infección de garganta
9	No	No	Sí	Sí	Resfriado

Introducción. Representación del conocimiento mediante árboles de decisión

- Ejemplo: Problema "Jugar al aire libre", J.R. Quinlan (1986)

Id	Ambiente	Temperatura	Humedad	Viento	*Clase*
E1	soleado	Alta	Alta	Falso	No
E2	soleado	Alta	Alta	Verdadero	No
E3	nublado	Alta	Alta	Falso	Sí
E4	Lluvioso	Media	Alta	Falso	Sí
E5	Lluvioso	Baja	Normal	Falso	Sí
E6	Lluvioso	Baja	Normal	Verdadero	No
E7	Nublado	Baja	Normal	Verdadero	Sí
E8	Soleado	Media	Alta	Falso	No
E9	Soleado	Baja	Normal	Falso	Sí
E10	Lluvioso	Media	Normal	Falso	Sí
E11	Soleado	Media	Normal	Verdadero	Sí
E12	Nublado	Media	Alta	Verdadero	Sí
E13	Nublado	Alta	Normal	Falso	Sí
E14	lluvioso	Media	alta	Verdadero	No

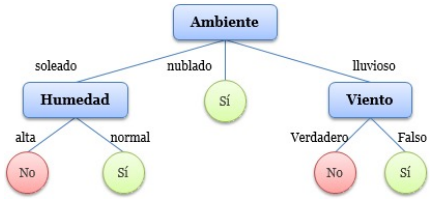
⇩

Aprender la función objetivo

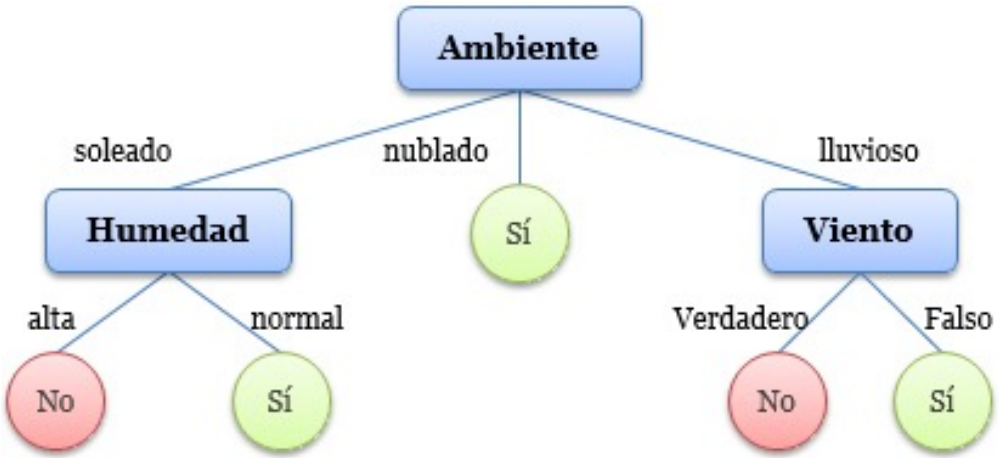
Entradas: atributos de las instancias

Salidas: clases o concepto a aprender

ÁRBOL DE DECISIÓN



Introducción. Representación del conocimiento mediante árboles de decisión



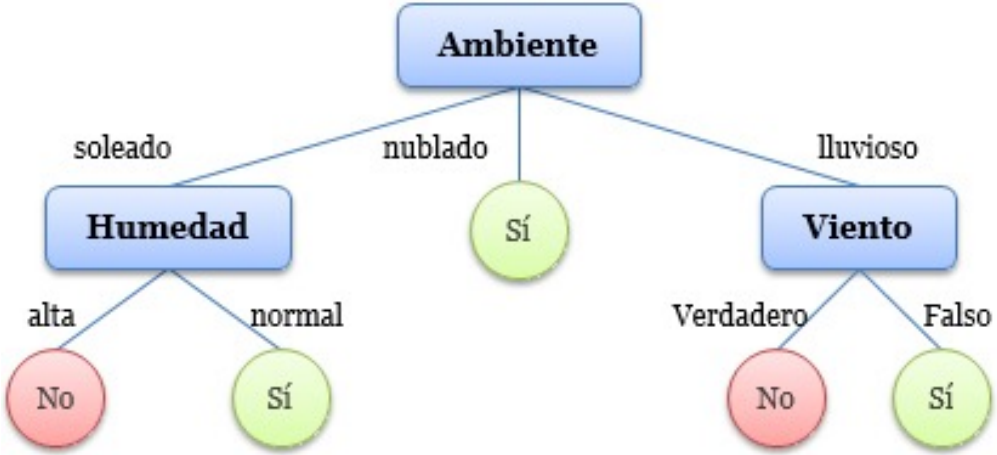
Función objetivo – Mapeo a reglas

SI ambiente es soleado
AND humedad es normal
OR ambiente es nublado
OR ambiente es lluvioso
AND viento es falso
ENTONCES jugar = Sí

SI ambiente es soleado
AND humedad es alta
OR ambiente es lluvioso
AND viento es verdadero
ENTONCES jugar = No

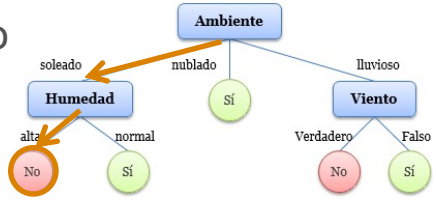
- **Árbol de decisión:** representación de la función objetivo como una serie de condiciones consecutivas.
  - Nodos: atributos (ej. Ambiente, Humedad, Viento).
  - Arcos: valores de los atributos (ej. soleado, nublado o lluvioso para el atributo Ambiente).
  - Hojas: clases (ej. Sí o No).
  - Rama: condiciones desde la raíz a la hoja unidas a través de conjunciones (AND) y entre ramas a través de disyunciones (OR).

Introducción. Representación del conocimiento mediante árboles de decisión

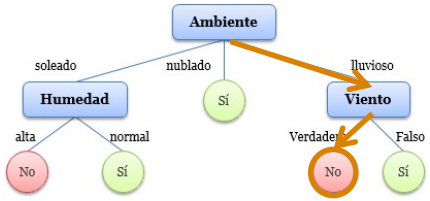


- **Clasificación con el árbol de decisión:**
  - Comparar los valores de los atributos de una instancia cuya clase es desconocida con las ramas del árbol de decisión.

Ambiente = soleado
Temperatura = alta
Humedad = alta
Viento = falso
¿Jugar al aire libre? No

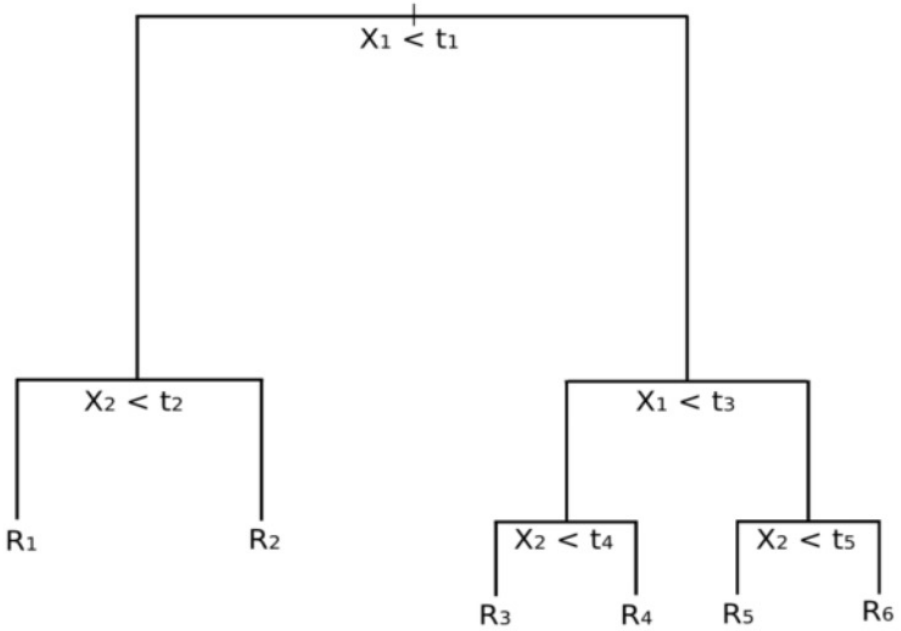


Ambiente = lluvioso
Temperatura = baja
Humedad = alta
Viento = verdadero
¿Jugar al aire libre? No

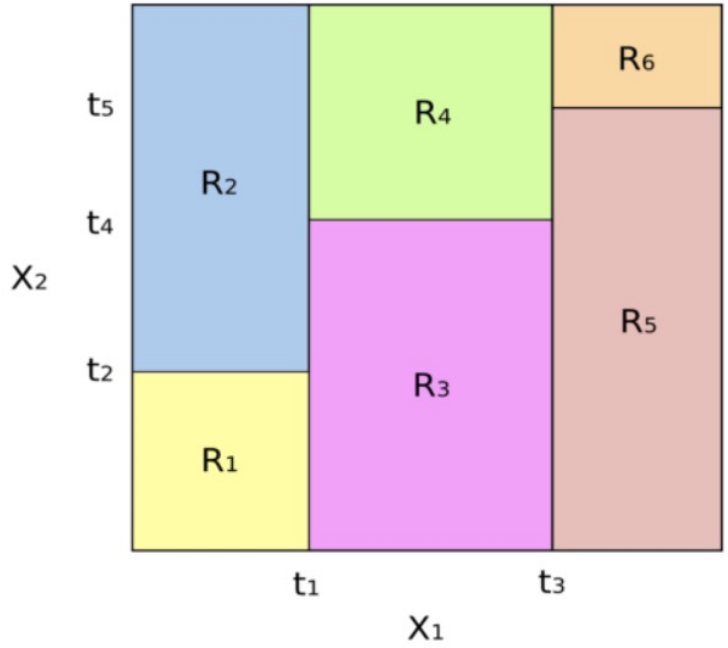


Introducción. Representación del conocimiento mediante árboles de decisión

- Divide el espacio de variables independientes en regiones distintas y no superpuestas



A Decision Tree with six separate regions



The resulting partition of the subset of $\mathbb{R}^2$ into six regional "blocks"

Fuente: https://bookdown.org/content/2031/arboles-de-decision-parte-i.html

Introducción. Representación del conocimiento mediante árboles de decisión

- ¿Cuándo son adecuados?
  - Instancias representadas por pares atributos-valores.
  - Atributos de entrada con valores nominales o numéricos.
  - Valores de salida de la función objetivo discretos (nominales).
  - Existen errores o valores de atributos desconocidos en datos de entrenamiento.
- Árboles de Clasificación
  - Variable dependiente categórica.
  - Valores de nodos se reducen a la moda de las observaciones en esa región.
- Árboles de Regresión
  - Variable dependiente continua.
  - Valores de nodos se reducen a la media de las observaciones en esa región.

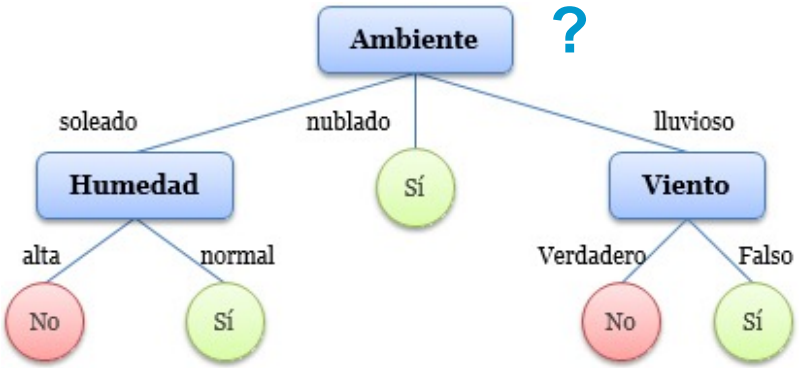
Introducción. Representación del conocimiento mediante árboles de decisión

- Ventajas
  - Fáciles de comprender y mapear a reglas.
  - Trabajan con conjuntos de datos tanto numéricos como nominales.
  - Trabajan con datos multidimensionales.
  - No requieren conocimiento en un dominio dado ni establecer parámetros.
  - Útiles para exploración de datos: identifica variables importantes.
  - Método no paramétrico: sin suposiciones.
- Desventajas
  - Los atributos de salida deben ser categorías: pérdida de información al categorizar variables continuas.
  - Sobreajuste.
  - No se permiten múltiples atributos de salida.
  - Si los datos son numéricos, los árboles pueden resultar muy complejos.
  - Inestables: pequeños cambios en los datos pueden cambiar el árbol notablemente.

Descripción de la tarea de inducción

Hipótesis del aprendizaje inductivo de conceptos:
Cualquier hipótesis que encaje «suficientemente» bien con un conjunto «suficientemente» grande de ejemplos de entrenamiento también encajará bien con instancias nuevas.

- Tarea de inducción del árbol de decisión: encontrar el árbol que mejor encaje con los datos de ejemplo disponibles y ya clasificados.
- Espacio de hipótesis: conjunto de todos los árboles de decisión posibles.
- Método de selección de atributos: criterio utilizado para generar las diferentes ramas del árbol.



Descripción de la tarea de inducción

¿Cómo decide un árbol dónde ramificar?

- Decisión importante pues afecta altamente la precisión del árbol.
- Los criterios de decisión son diferentes para árboles de clasificación y regresión.
- La creación de subnodos incrementa la homogeneidad de los subnodos resultantes.
- Se prueba la división con todas las variables y se escoge la que produce subnodos más homogéneos.
- Varios algoritmos para decidir la ramificación: ID3, CART, C4.5

Descripción de la tarea de inducción

Índice de Gini

Si seleccionamos aleatoriamente dos instancias de una población, entonces estos deben ser de la misma clase y la probabilidad de esto es 1 si la población es pura.

- Variable objetivo categórica.
- Divisiones binaria.
- Mayor índice Gini, mayor homogeneidad.
- CART (*Classification and Regression Tree*) usa el método de Gini para la división binaria.

Descripción de la tarea de inducción

Chi Cuadrado

Significancia estadística de las diferencias entre subnodos y un nodo padre.

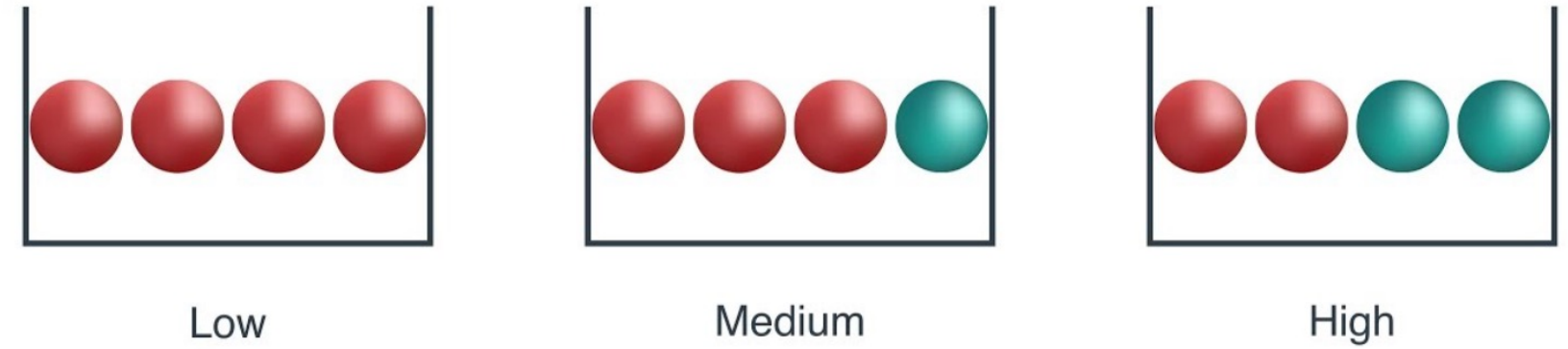
- Variable objetivo categórica.
- Dos o más divisiones.
- A más alto valor de Chi-Cuadrado, más alta la significancia estadística de las diferencias entre cada nodo y el nodo padre.

Descripción de la tarea de inducción

Entropía

Un nodo menos impuro requiere menos información para ser descrito mientras un nodo más impuro necesita más información.

- Mide el grado de desinformación de una sistema.
- Muestra completamente homogénea = entropía 0.
- Muestra igualmente dividida (50% – 50%) = entropía 1.



Descripción de la tarea de inducción

Reducción de la varianza

Se utiliza para variables continuas.

- Problemas de regresión.
- Utiliza la varianza para escoger el criterio de división.
- La población se divide en función de la varianza más baja.

Descripción de la tarea de inducción

- Algoritmo para construir el árbol de decisión si los ejemplos tienen atributos de entrada nominales

```
    PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
    COMIENZO
P1  Crear un nodo N;
P2  SI todos los elementos de E pertenecen a la misma clase, C
    ENTONCES Retornar N como nodo hoja etiquetado con la clase C
P3  SI_NO SI la lista de atributos (Lista_Atributos) está vacía
    ENTONCES Retornar N como nodo hoja etiquetado con la clase más numerosa en los ejemplos
P4  SI_NO    Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el
         atributo A que mejor particiona E
P5       Borrar Atributo A de la lista de Atributos Lista_Atributos
P6       Etiquetar N con el atributo seleccionado
P7       PARA CADA valor V de A
         Siendo Ev el subconjunto de elementos en E con valor V en el atributo A.
P8       SI Ev está vacío
         ENTONCES unir al nodo N una hoja etiquetada con la clase mayoritaria en E.
P9       SINO unir al nodo N el nodo retornado de Inducir_Arbol (Ev, Lista_Atributos,
         Método_Selección_Atributos)
         FIN PARA CADA
    FIN SI-SI_NO
    FIN
```

Descripción de la tarea de inducción

```
   PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
   COMIENZO
P1 Crear un nodo N;
P2 SI todos los elementos de E pertenecen a la misma clase, C
   ENTONCES Retornar N como nodo hoja etiquetado con la clase C
P3 SINO SI la lista de atributos (Lista_Atributos) está vacia
   ENTONCES Retornar N como nodo hoja etiquetado con la clase más numerosa en los ejemplos
P4 SINO Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el
        atributo A que mejor particiona E
P5      Borrar Atributo A de la lista de Atributos Lista_Atributos
P6      Etiquetar N con el atributo seleccionado
P7      PARA CADA valor V de A
        Siendo Ev el subconjunto de elementos en E con valor V en el atributo A.
P8      SI Ev está vacio
        ENTONCES unir al nodo N una hoja etiquetada con la clase mayoritaria en E.
P9      SINO unir al nodo N el nodo retornado de Inducir_Arbol (Ev, Lista_Atributos,
        Método_Selección_Atributos)
        FIN PARA CADA
   FIN SI-SINO
   FIN
```

- Algoritmo para construir el árbol de decisión

```
PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
COMIENZO
```

E = {E1, E2, … Ei} Lista_Atributos = {A1, A2, … Aj} Método_Selección_Atributos = m

Id	Atributo A1	Atributo A2	…	Atributo Aj	Clase
E1	$V_{A1} \in$ {a, b…,e}	$V_{A2} \in$ {a, b…,z}		$V_{aj} \in$ {a, b…,m}	C ∈ {C1, C2…, Cn}
E2	$V_{A1} \in$ {a, b…,e}	$V_{A2} \in$ {a, b…,z}		$V_{aj} \in$ {a, b…,m}	C ∈ {C1, C2…, Cn}
…	$V_{A1} \in$ {a, b…,e}	$V_{A2} \in$ {a, b…,z}		$V_{aj} \in$ {a, b…,m}	C ∈ {C1, C2…, Cn}
Ei	$V_{A1} \in$ {a, b…,e}	$V_{A2} \in$ {a, b…,z}		$V_{aj} \in$ {a, b…,m}	C ∈ {C1, C2…, Cn}

Clase:
C ∈ {C1, C2…, Cn}

P1 `Crear un nodo N;`

Nodo N

Descripción de la tarea de inducción

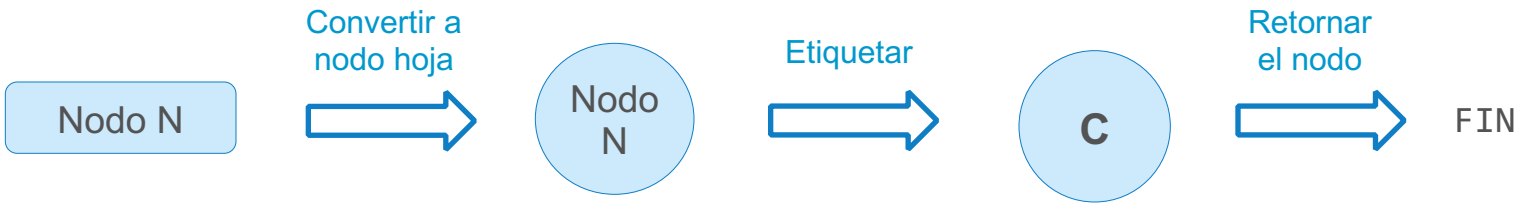
```
   PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
   COMIENZO
P1 Crear un nodo N;
P2 SI todos los elementos de E pertenecen a la misma clase, C
   ENTONCES Retornar N como nodo hoja etiquetado con la clase C
P3 SINO SI la lista de atributos (Lista_Atributos) está vacia
   ENTONCES Retornar N como nodo hoja etiquetado con la clase más numerosa en los ejemplos
P4 SINO Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el
        atributo A que mejor particiona E
P5      Borrar Atributo A de la lista de Atributos Lista_Atributos
P6      Etiquetar N con el atributo seleccionado
P7      PARA CADA valor V de A
        Siendo Ev el subconjunto de elementos en E con valor V en el atributo A.
P8      SI Ev está vacio
        ENTONCES unir al nodo N una hoja etiquetada con la clase mayoritaria en E.
P9      SINO unir al nodo N el nodo retornado de Inducir_Arbol (Ev, Lista_Atributos,
        Método_Selección_Atributos)
        FIN PARA CADA
   FIN SI-SINO
   FIN
```

- Algoritmo para construir el árbol de decisión

```
P2  SI todos los elementos de E pertenecen a la misma clase, C
    ENTONCES Retornar N como nodo hoja etiquetado con la clase C
```

E = {E1, E2, … Ei}

Id	Atributo A1	Atributo A2	…	Atributo Aj	Clase
E1	$V_{A1} \in \{a, b\ldots,e\}$	$V_{A2} \in \{a, b\ldots,z\}$		$V_{aj} \in \{a, b\ldots,m\}$	C
E2	$V_{A1} \in \{a, b\ldots,e\}$	$V_{A2} \in \{a, b\ldots,z\}$		$V_{aj} \in \{a, b\ldots,m\}$	C
…	$V_{A1} \in \{a, b\ldots,e\}$	$V_{A2} \in \{a, b\ldots,z\}$		$V_{aj} \in \{a, b\ldots,m\}$	C
Ei	$V_{A1} \in \{a, b\ldots,e\}$	$V_{A2} \in \{a, b\ldots,z\}$		$V_{aj} \in \{a, b\ldots,m\}$	C



Descripción de la tarea de inducción

```
   PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
   COMIENZO
P1 Crear un nodo N;
P2 SI todos los elementos de E pertenecen a la misma clase, C
   ENTONCES Retornar N como nodo hoja etiquetado con la clase C
P3 SINO SI la lista de atributos (Lista_Atributos) está vacía
   ENTONCES Retornar N como nodo hoja etiquetado con la clase más numerosa en los ejemplos
P4 SINO Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el
        atributo A que mejor particiona E
P5      Borrar Atributo A de la lista de Atributos Lista_Atributos
P6      Etiquetar N con el atributo seleccionado
P7      PARA CADA valor V de A
        Siendo Ev el subconjunto de elementos en E con valor V en el atributo A.
P8      SI Ev está vacío
        ENTONCES unir al nodo N una hoja etiquetada con la clase mayoritaria en E.
P9      SINO unir al nodo N el nodo retornado de Inducir_Arbol (Ev, Lista_Atributos,
        Método_Selección_Atributos)
        FIN PARA CADA
   FIN SI-SINO
   FIN
```

- Algoritmo para construir el árbol de decisión

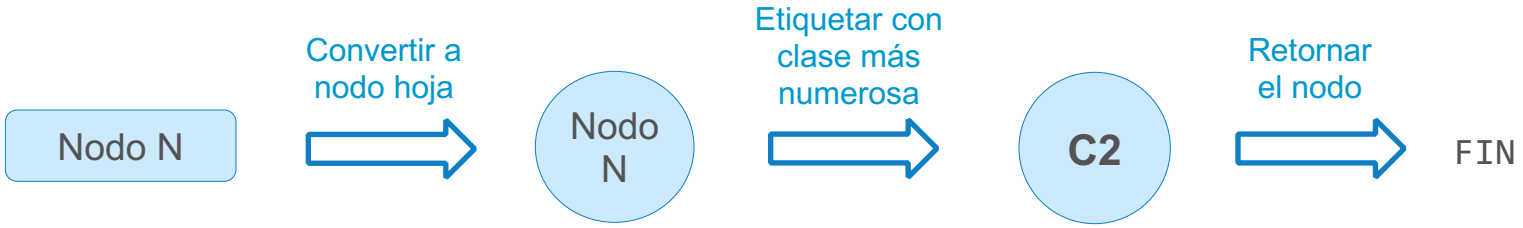
```
P3  SI_NO SI la lista de atributos (Lista_Atributos) está vacía
    ENTONCES Retornar N como nodo hoja etiquetado con la clase más numerosa en los ejemplos
```

E = {E1, E2, … Ei} Lista_Artributos = { } → *Todos los atributos A1, A2, … Aj ya se han utilizado en nodos superiores del árbol*

Id	Atributo A1	Atributo A2	…	Atributo Aj	Clase
E1	$V_{A1} \in$ {a, b…,e}	$V_{A2} \in$ {a, b…,z}		$V_{aj} \in$ {a, b…,m}	C2
E2	$V_{A1} \in$ {a, b…,e}	$V_{A2} \in$ {a, b…,z}		$V_{aj} \in$ {a, b…,m}	C1
…	$V_{A1} \in$ {a, b…,e}	$V_{A2} \in$ {a, b…,z}		$V_{aj} \in$ {a, b…,m}	…
Ei	$V_{A1} \in$ {a, b…,e}	$V_{A2} \in$ {a, b…,z}		$V_{aj} \in$ {a, b…,m}	C2

Calcular cuántas veces aparecen las clases (C1, C2, … Cn) en el conjunto de ejemplos (E) → C1: x veces C2: y veces … Cn: z veces

y = max[x, y, … z] → *C2 es la clase que más aparece en los ejemplos*



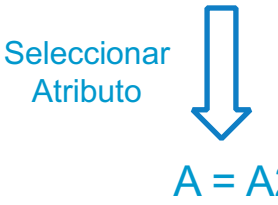
Descripción de la tarea de inducción

```
   PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
   COMIENZO
P1 Crear un nodo N;
P2 SI todos los elementos de E pertenecen a la misma clase, C
   ENTONCES Retornar N como nodo hoja etiquetado con la clase C
P3 SINO SI la lista de atributos (Lista_Atributos) está vacía
   ENTONCES Retornar N como nodo hoja etiquetado con la clase más numerosa en los ejemplos
P4 SINO Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el
        atributo A que mejor particiona E
P5      Borrar Atributo A de la lista de Atributos Lista_Atributos
P6      Etiquetar N con el atributo seleccionado
P7      PARA CADA valor V de A
        Siendo Ev el subconjunto de elementos en E con valor V en el atributo A.
P8      SI Ev está vacío
        ENTONCES unir al nodo N una hoja etiquetada con la clase mayoritaria en E.
P9      SINO unir al nodo N el nodo retornado de Inducir_Arbol (Ev, Lista_Atributos,
        Método_Selección_Atributos)
        FIN PARA CADA
   FIN SI-SINO
   FIN
```

- Algoritmo para construir el árbol de decisión

P4 `SI_NO Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el atributo A que mejor particiona E`

E = {E1, E2, … Ei} Lista_Atributos = {A1, A2, … Aj} Método_Selección_Atributos = m



P5 `Borrar Atributo A de la lista de Atributos Lista_Atributos`

P6 `Etiquetar N con el atributo seleccionado`

Descripción de la tarea de inducción

```
   PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
   COMIENZO
P1 Crear un nodo N;
P2 SI todos los elementos de E pertenecen a la misma clase, C
   ENTONCES Retornar N como nodo hoja etiquetado con la clase C
P3 SINO SI la lista de atributos (Lista_Atributos) está vacia
   ENTONCES Retornar N como nodo hoja etiquetado con la clase más numerosa en los ejemplos
P4 SINO Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el
        atributo A que mejor particiona E
P5      Borrar Atributo A de la lista de Atributos Lista_Atributos
P6      Etiquetar N con el atributo seleccionado
P7      PARA CADA valor V de A
        Siendo Ev el subconjunto de elementos en E con valor V en el atributo A.
P8      SI Ev está vacio
        ENTONCES unir al nodo N una hoja etiquetada con la clase mayoritaria en E.
P9      SINO unir al nodo N el nodo retornado de Inducir_Arbol (Ev, Lista_Atributos,
        Método_Selección_Atributos)
        FIN PARA CADA
   FIN SI-SINO
   FIN
```

- Algoritmo para construir el árbol de decisión

```
PARA CADA valor V de A
```

A = A2

Id	Atributo A1	Atributo A2	…	Atributo Aj	Clase
E1	$V_{A1} \in \{a, b\ldots,e\}$	$V_{A2} \in \{a, b\ldots,z\}$		$V_{aj} \in \{a, b\ldots,m\}$	$C \in \{C1, C2\ldots, Cn\}$
E2	$V_{A1} \in \{a, b\ldots,e\}$	$V_{A2} \in \{a, b\ldots,z\}$		$V_{aj} \in \{a, b\ldots,m\}$	$C \in \{C1, C2\ldots, Cn\}$
…	$V_{A1} \in \{a, b\ldots,e\}$	$V_{A2} \in \{a, b\ldots,z\}$		$V_{aj} \in \{a, b\ldots,m\}$	$C \in \{C1, C2\ldots, Cn\}$
Ei	$V_{A1} \in \{a, b\ldots,e\}$	$V_{A2} \in \{a, b\ldots,z\}$		$V_{aj} \in \{a, b\ldots,m\}$	$C \in \{C1, C2\ldots, Cn\}$

$V_{A2} \in \{a, b\ldots,z\}$

Iterar en V_{A2} ⇩

PARA V=a

Descripción de la tarea de inducción

```
PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
COMIENZO
P1  Crear un nodo N;
P2  SI todos los elementos de E pertenecen a la misma clase, C
    ENTONCES Retornar N como nodo hoja etiquetado con la clase C
P3  SINO SI la lista de atributos (Lista_Atributos) está vacia
    ENTONCES Retornar N como nodo hoja etiquetado con la clase más numerosa en los ejemplos
P4  SINO Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el
         atributo A que mejor particiona E
P5       Borrar Atributo A de la lista de Atributos Lista_Atributos
P6       Etiquetar N con el atributo seleccionado
P7       PARA CADA valor V de A
         Siendo Ev el subconjunto de elementos en E con valor V en el atributo A.
P8       SI Ev está vacio
         ENTONCES unir al nodo N una hoja etiquetada con la clase mayoritaria en E.
P9       SINO unir al nodo N el nodo retornado de Inducir_Arbol (Ev, Lista_Atributos,
         Método_Selección_Atributos)
         FIN PARA CADA
FIN SI-SINO
FIN
```

- Algoritmo para construir el árbol de decisión

PARA CADA valor V de A

PARA V=a A = A2 $V_{A2} \in \{a, b\ldots,z\}$

Siendo Ev el subconjunto de elementos en E con valor V en el atributo A.

E = {E1, E2, … Ei}

Id	Atributo A1	Atributo A2	…	Atributo Aj	Clase
E1	$V_{A1} \in \{a, b\ldots,e\}$	g		$V_{aj} \in \{a, b\ldots,m\}$	C ∈ {C1, C2…, Cn}
E2	$V_{A1} \in \{a, b\ldots,e\}$	a		$V_{aj} \in \{a, b\ldots,m\}$	C ∈ {C1, C2…, Cn}
…	$V_{A1} \in \{a, b\ldots,e\}$	…		$V_{aj} \in \{a, b\ldots,m\}$	C ∈ {C1, C2…, Cn}
Ei	$V_{A1} \in \{a, b\ldots,e\}$	a		$V_{aj} \in \{a, b\ldots,m\}$	C ∈ {C1, C2…, Cn}

¿Qué ejemplos tienen en el atributo A2 el valor V=a? E2, Ei

Ev = {E2, Ei}

Descripción de la tarea de inducción

```
   PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
   COMIENZO
P1 Crear un nodo N;
P2 SI todos los elementos de E pertenecen a la misma clase, C
   ENTONCES Retornar N como nodo hoja etiquetado con la clase C
P3 SINO SI la lista de atributos (Lista_Atributos) está vacía
   ENTONCES Retornar N como nodo hoja etiquetado con la clase más numerosa en los ejemplos
P4 SINO Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el
        atributo A que mejor particiona E
P5      Borrar Atributo A de la lista de Atributos Lista_Atributos
P6      Etiquetar N con el atributo seleccionado
P7      PARA CADA valor V de A
        Siendo Ev el subconjunto de elementos en E con valor V en el atributo A.
P8      SI Ev está vacío
        ENTONCES unir al nodo N una hoja etiquetada con la clase mayoritaria en E.
P9      SINO unir al nodo N el nodo retornado de Inducir_Arbol (Ev, Lista_Atributos,
        Método_Selección_Atributos)
        FIN PARA CADA
   FIN SI-SINO
   FIN
```

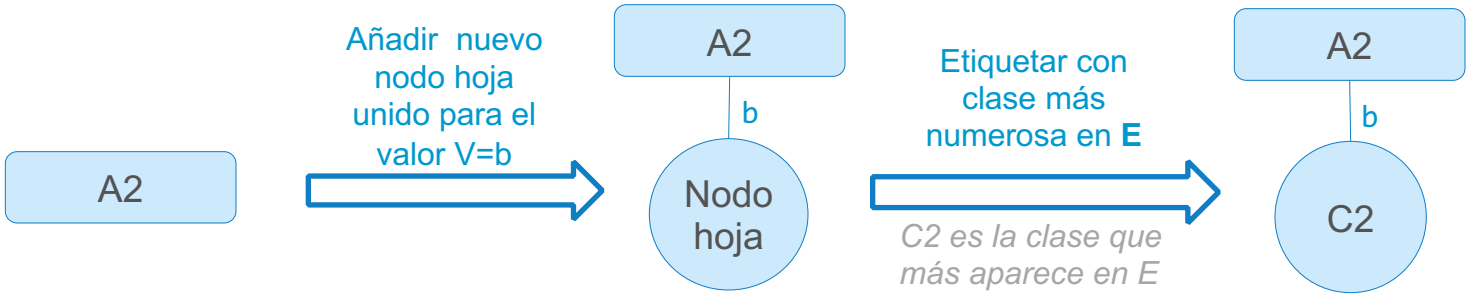
- Algoritmo para construir el árbol de decisión

P8 `SI Ev está vacío`
`ENTONCES unir al nodo N una hoja etiquetada con la clase mayoritaria en E.`

$E = \{E1, E2, \ldots Ei\}$ $A = A2$ $V_{A2} \in \{a, b\ldots,z\}$ PARA V=b → Ev = { }

Id	Atributo A1	Atributo A2	…	Atributo Aj	Clase
E1	$V_{A1} \in \{a, b\ldots,e\}$	g		$V_{aj} \in \{a, b\ldots,m\}$	C2
E2	$V_{A1} \in \{a, b\ldots,e\}$	a		$V_{aj} \in \{a, b\ldots,m\}$	C1
…	$V_{A1} \in \{a, b\ldots,e\}$	…		$V_{aj} \in \{a, b\ldots,m\}$	…
Ei	$V_{A1} \in \{a, b\ldots,e\}$	a		$V_{aj} \in \{a, b\ldots,m\}$	C2

Ev = { } → *Ningún ejemplo tiene en el atributo A2 el valor V=b*



Descripción de la tarea de inducción

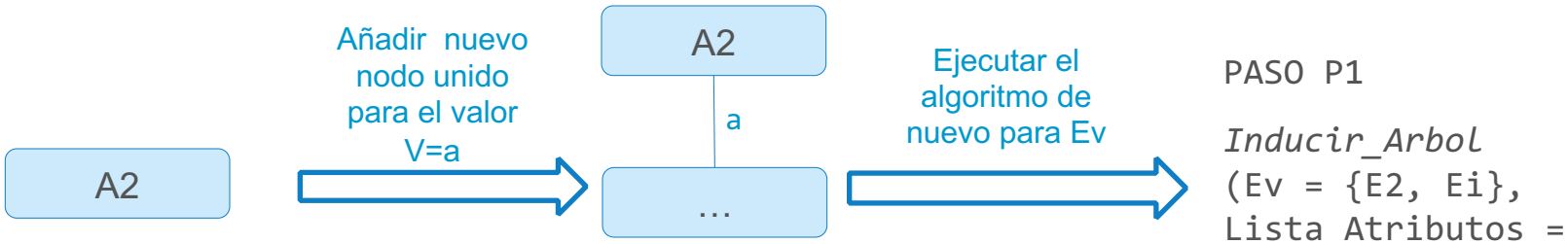
```
   PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
   COMIENZO
P1 Crear un nodo N;
P2 SI todos los elementos de E pertenecen a la misma clase, C
   ENTONCES Retornar N como nodo hoja etiquetado con la clase C
P3 SINO SI la lista de atributos (Lista_Atributos) está vacia
   ENTONCES Retornar N como nodo hoja etiquetado con la clase más numerosa en los ejemplos
P4 SINO Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el
        atributo A que mejor particiona E
P5      Borrar Atributo A de la lista de Atributos Lista_Atributos
P6      Etiquetar N con el atributo seleccionado
P7      PARA CADA valor V de A
        Siendo Ev el subconjunto de elementos en E con valor V en el atributo A.
P8      SI Ev está vacio
        ENTONCES unir al nodo N una hoja etiquetada con la clase mayoritaria en E.
P9      SINO unir al nodo N el nodo retornado de Inducir_Arbol (Ev, Lista_Atributos,
        Método_Selección_Atributos)
        FIN PARA CADA
   FIN SI-SINO
   FIN
```

- Algoritmo para construir el árbol de decisión

P9 SI_NO unir al nodo N el nodo retornado de *Inducir_Arbol* (Ev, Lista_Atributos, Método_Selección_Atributos)

E = {E1, E2, … Ei} A = A2 $V_{A2} \in \{a, b\ldots,z\}$ PARA V=a → Ev = {E2, Ei}

Id	Atributo A1	Atributo A2	…	Atributo Aj	Clase
E1	$V_{A1} \in \{a, b\ldots,e\}$	g		$V_{aj} \in \{a, b\ldots,m\}$	C ∈ {C1, C2…, Cn}
E2	$V_{A1} \in \{a, b\ldots,e\}$	a		$V_{aj} \in \{a, b\ldots,m\}$	C ∈ {C1, C2…, Cn}
…	$V_{A1} \in \{a, b\ldots,e\}$	…		$V_{aj} \in \{a, b\ldots,m\}$	C ∈ {C1, C2…, Cn}
Ei	$V_{A1} \in \{a, b\ldots,e\}$	a		$V_{aj} \in \{a, b\ldots,m\}$	C ∈ {C1, C2…, Cn}



PASO P1

Inducir_Arbol (Ev = {E2, Ei}, Lista_Atributos = {A1, A3, … Aj}, método)

Descripción de la tarea de inducción

- Ejemplo: Problema "Jugar al aire libre", J.R. Quinlan (1986)

Conjunto de datos de entrenamiento

Id	Ambiente	Temperatura	Humedad	Viento	*Clase*
E1	soleado	Alta	Alta	Falso	No
E2	soleado	Alta	Alta	Verdadero	No
E3	nublado	Alta	Alta	Falso	Sí
E4	Lluvioso	Media	Alta	Falso	Sí
E5	Lluvioso	Baja	Normal	Falso	Sí
E6	Lluvioso	Baja	Normal	Verdadero	No
E7	Nublado	Baja	Normal	Verdadero	Sí
E8	Soleado	Media	Alta	Falso	No
E9	Soleado	Baja	Normal	Falso	Sí
E10	Lluvioso	Media	Normal	Falso	Sí
E11	Soleado	Media	Normal	Verdadero	Sí
E12	Nublado	Media	Alta	Verdadero	Sí
E13	Nublado	Alta	Normal	Falso	Sí
E14	lluvioso	Media	alta	Verdadero	No

14 instancias

E = {E1, E2, … E14}

4 Atributos de entrada:
Ambiente {soleado, nublado, lluvioso}
Temperatura {alta, media, baja}
Humedad {alta, normal}
Viento {verdadero, falso}

Clase (o atributo de salida):
{Sí, No}

Lista_Atributos = {Ambiente, Temperatura, Humedad, Viento}

Descripción de la tarea de inducción

- Aplicación del algoritmo para construir el árbol de decisión

E = {E1, E2, … E14}

Lista_Atributos = {Ambiente, Temperatura, Humedad, Viento}

```
    PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
    COMIENZO
P1  Crear un nodo N;
```

N

FALSO {Sí, No}

```
P2  SI todos los elementos de E pertenecen a la misma clase, C
    ENTONCES Retornar N como nodo hoja etiquetado con la clase C,
```

Id	Ambiente	Temperatura	Humedad	Viento	*Clase*
E1	soleado	Alta	Alta	Falso	No
E2	soleado	Alta	Alta	Verdadero	No
E3	nublado	Alta	Alta	Falso	Sí
E4	Lluvioso	Media	Alta	Falso	Sí
E5	Lluvioso	Baja	Normal	Falso	Sí
E6	Lluvioso	Baja	Normal	Verdadero	No
E7	Nublado	Baja	Normal	Verdadero	Sí
E8	Soleado	Media	Alta	Falso	No
E9	Soleado	Baja	Normal	Falso	Sí
E10	Lluvioso	Media	Normal	Falso	Sí
E11	Soleado	Media	Normal	Verdadero	Sí
E12	Nublado	Media	Alta	Verdadero	Sí
E13	Nublado	Alta	Normal	Falso	Sí
E14	lluvioso	Media	alta	Verdadero	No

FALSO Lista_Atributos = {Ambiente, Temperatura, Humedad, Viento}

```
P3  SI_NO SI la lista de atributos (Lista_Atributos) está vacía
    ENTONCES Retornar N como nodo hoja etiquetado con la clase más numerosa en los ejemplos

P4  SI_NO Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el
        atributo A que mejor particiona E
```

A = Ambiente

Descripción de la tarea de inducción

- Aplicación del algoritmo para construir el árbol de decisión

P4 SI_NO Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el atributo A que mejor particiona E

A = Ambiente

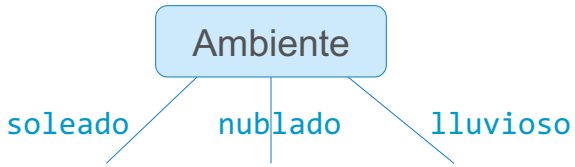
P5 Borrar Atributo A de la lista de Atributos Lista_Atributos

Lista_Atributos = {Temperatura, Humedad, Viento}

P6 Etiquetar N con el atributo seleccionado

Ambiente

PARA CADA valor V de A ← V(A) = {soleado, nublado, lluvioso}



Descripción de la tarea de inducción

- Aplicación del algoritmo para construir el árbol de decisión

PARA CADA valor V de A V = soleado

P7 Siendo Ev el subconjunto de elementos en E con valor V en el atributo A.

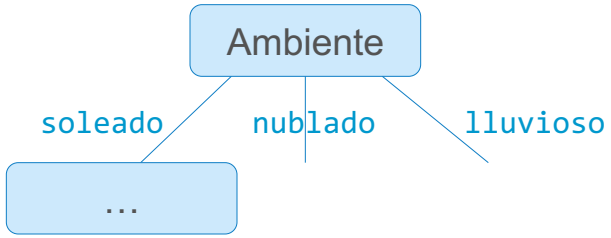
Id	Ambiente	Temperatura	Humedad	Viento	*Clase*
E1	soleado	Alta	Alta	Falso	No
E2	soleado	Alta	Alta	Verdadero	No
E3	nublado	Alta	Alta	Falso	Sí
E4	Lluvioso	Media	Alta	Falso	Sí
E5	Lluvioso	Baja	Normal	Falso	Sí
E6	Lluvioso	Baja	Normal	Verdadero	No
E7	Nublado	Baja	Normal	Verdadero	Sí
E8	Soleado	Media	Alta	Falso	No
E9	Soleado	Baja	Normal	Falso	Sí
E10	Lluvioso	Media	Normal	Falso	Sí
E11	Soleado	Media	Normal	Verdadero	Sí
E12	Nublado	Media	Alta	Verdadero	Sí
E13	Nublado	Alta	Normal	Falso	Sí
E14	lluvioso	Media	alta	Verdadero	No

Ev = {E1, E2, E8, E9, E11}

P8 SI Ev está vacío ← FALSO
ENTONCES unir al nodo N una hoja etiquetada con la clase mayoritaria en E.

P9 SI_NO unir al nodo N el nodo retornado de *Inducir_Arbol* (Ev, Lista_Atributos, Método_Selección_Atributos)

Volver al *P1 con E* = Ev = {E1, E2, E8, E9, E11} y Lista_Atributos = {Temperatura, Humedad, Viento} *y continuar…*



Descripción de la tarea de inducción

- Aplicación del algoritmo para construir el árbol de decisión

E = Ev = {E1, E2, E8, E9, E11}

Lista_Atributos = {Temperatura, Humedad, Viento}

```
PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
COMIENZO
P1  Crear un nodo N;
```

N

```
P2  SI todos los elementos de E pertenecen a la misma clase, C   FALSO
    ENTONCES Retornar N como nodo hoja etiquetado con la clase C
P3  SI_NO SI la lista de atributos (Lista_Atributos) está vacía  FALSO
    ENTONCES Retornar N como nodo hoja etiquetado con la clase más numerosa en los ejemplos
P4  SI_NO    Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el
         atributo A que mejor particiona E
```

Humedad

```
P5       Borrar Atributo A de la lista de Atributos Lista_Atributos
```

Lista_Atributos = {Temperatura, Viento}

```
P6       Etiquetar N con el atributo seleccionado
```

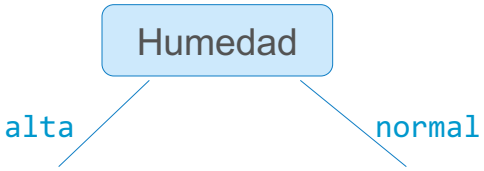
Humedad

Id	Ambiente	Temperatura	Humedad	Viento	Clase
E1	soleado	Alta	Alta	Falso	No
E2	soleado	Alta	Alta	Verdadero	No
E8	Soleado	Media	Alta	Falso	No
E9	Soleado	Baja	Normal	Falso	Sí
E11	Soleado	Media	Normal	Verdadero	Sí

Descripción de la tarea de inducción

- Aplicación del algoritmo para construir el árbol de decisión

PARA CADA valor V de A



PARA V = alta

P7 Siendo Ev el subconjunto de elementos en E con valor V en el atributo A.

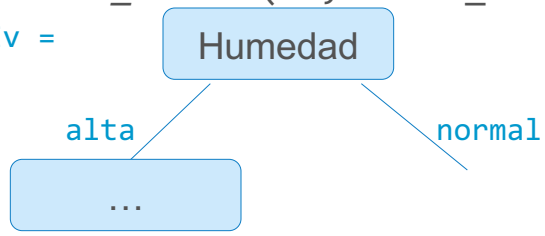
Id	Ambiente	Temperatura	Humedad	Viento	*Clase*
E1	soleado	Alta	Alta	Falso	No
E2	soleado	Alta	Alta	Verdadero	No
E3	Nublado	Alta	Alta	Falso	Sí
E4	Lluvioso	Media	Alta	Falso	Sí
E5	Lluvioso	Baja	Normal	Falso	Sí
E6	Lluvioso	Baja	Normal	Verdadero	No
E7	Nublado	Baja	Normal	Verdadero	Sí
E8	Soleado	Media	Alta	Falso	No
E9	Soleado	Baja	Normal	Falso	Sí
E10	Lluvioso	Media	Normal	Falso	Sí
E11	Soleado	Media	Normal	Verdadero	Sí
E12	Nublado	Media	Alta	Verdadero	Sí
E13	Nublado	Alta	Normal	Falso	Sí
E14	Lluvioso	Media	alta	Verdadero	No

Ev = {E1, E2, E8}

P8 SI Ev está vacío ← FALSO
ENTONCES unir al nodo N una hoja etiquetada con la clase mayoritaria en E.

P9 SI_NO unir al nodo N el nodo retornado de *Inducir_Arbol* (Ev, Lista_Atributos, Método_Selección_Atributos)

Volver al *P1 con E* = Ev = {E1, E2, E8} y Lista_Atributos = {Temperatura, Viento} *y continuar…*



Descripción de la tarea de inducción

- Aplicación del algoritmo para construir el árbol de decisión

E = Ev = {E1, E2, E8}

Lista_Atributos = {Temperatura, Viento}

```
    PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
    COMIENZO
P1  Crear un nodo N;

P2  SI todos los elementos de E pertenecen a la misma clase, C
    ENTONCES Retornar N como nodo hoja etiquetado con la clase C,
```

N

VERDADERO
C = No

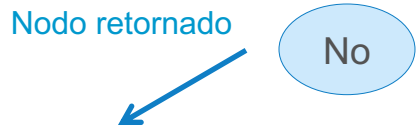
Id	Ambiente	Temperatura	Humedad	Viento	*Clase*
E1	soleado	Alta	Alta	Falso	No
E2	soleado	Alta	Alta	Verdadero	No
E3	Nublado	Alta	Alta	Falso	Sí
E4	Lluvioso	Media	Alta	Falso	Sí
E5	Lluvioso	Baja	Normal	Falso	Sí
E6	Lluvioso	Baja	Normal	Verdadero	No
E7	Nublado	Baja	Normal	Verdadero	Sí
E8	Soleado	Media	Alta	Falso	No
E9	Soleado	Baja	Normal	Falso	Sí
E10	Lluvioso	Media	Normal	Falso	Sí
E11	Soleado	Media	Normal	Verdadero	Sí
E12	Nublado	Media	Alta	Verdadero	Sí
E13	Nublado	Alta	Normal	Falso	Sí
E14	Lluvioso	Media	Alta	Verdadero	No

Retornar

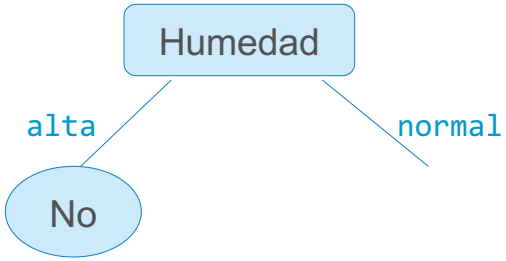
No

Descripción de la tarea de inducción

- Aplicación del algoritmo para construir el árbol de decisión



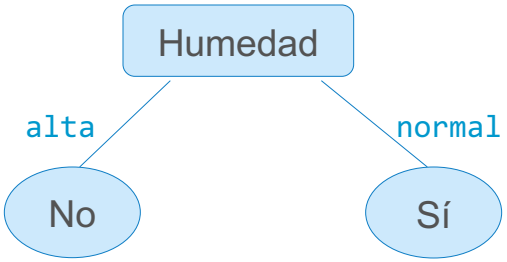
P9 SI_NO unir al nodo N el nodo retornado de *Inducir_Arbol* (Ev, Lista_Atributos, Método_Selección_Atributos)



Continuamos en PARA CADA con V= normal …

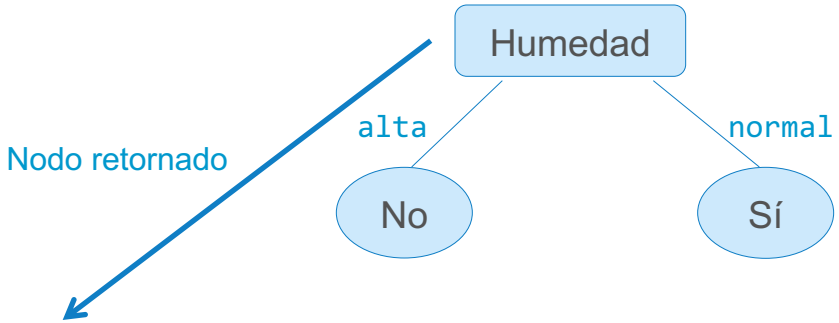
…

Retornar

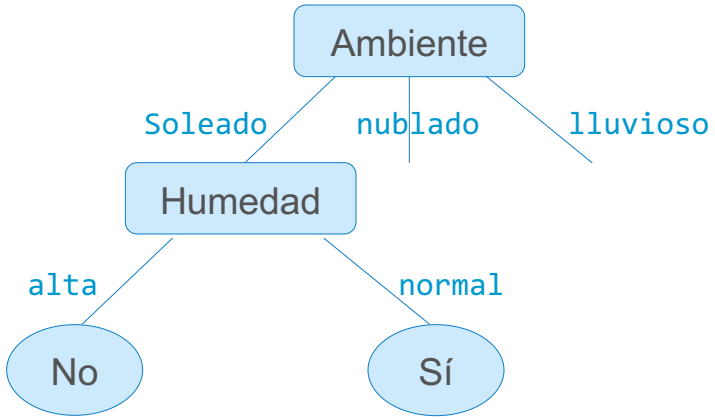


Descripción de la tarea de inducción

- Aplicación del algoritmo para construir el árbol de decisión



P9 SI_NO unir al nodo N el nodo retornado de *Inducir_Arbol* (Ev, Lista_Atributos, Método_Selección_Atributos)



Continuamos en PARA CADA con V= nublado …

Descripción de la tarea de inducción

- Aplicación del algoritmo para construir el árbol de decisión

PARA CADA valor V de A V = nublado

P7 Siendo Ev el subconjunto de elementos en E con valor V en el atributo A.

Id	Ambiente	Temperatura	Humedad	Viento	*Clase*
E1	soleado	Alta	Alta	Falso	No
E2	soleado	Alta	Alta	Verdadero	No
E3	nublado	Alta	Alta	Falso	Sí
E4	Lluvioso	Media	Alta	Falso	Sí
E5	Lluvioso	Baja	Normal	Falso	Sí
E6	Lluvioso	Baja	Normal	Verdadero	No
E7	Nublado	Baja	Normal	Verdadero	Sí
E8	Soleado	Media	Alta	Falso	No
E9	Soleado	Baja	Normal	Falso	Sí
E10	Lluvioso	Media	Normal	Falso	Sí
E11	Soleado	Media	Normal	Verdadero	Sí
E12	Nublado	Media	Alta	Verdadero	Sí
E13	Nublado	Alta	Normal	Falso	Sí
E14	lluvioso	Media	alta	Verdadero	No

Ev = {E3, E7, E12, E13}

P8 SI Ev está vacío ← FALSO
ENTONCES unir al nodo N una hoja etiquetada con la clase mayoritaria en E.

…

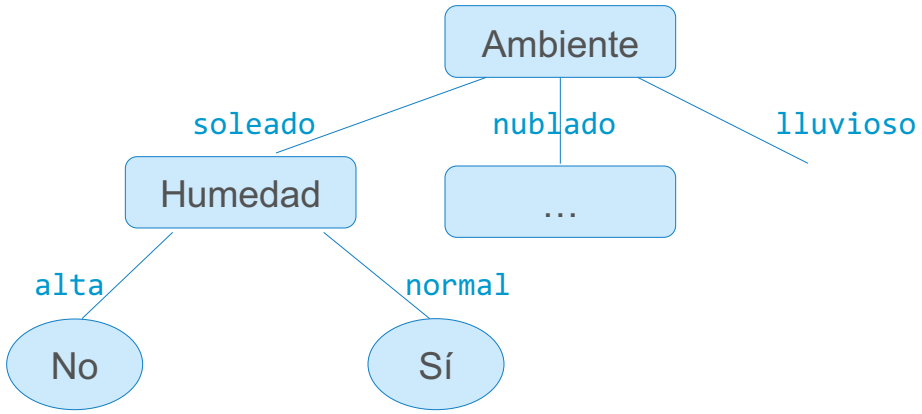
Descripción de la tarea de inducción

- Aplicación del algoritmo para construir el árbol de decisión

…

P9 SI_NO unir al nodo N el nodo retornado de *Inducir_Arbol* (Ev, Lista_Atributos, Método_Selección_Atributos)

Volver al *P1* con *E* = Ev = {E3, E7, E12, E13} y Lista_Atributos = {Temperatura, Humedad, Viento} *y continuar…*



Descripción de la tarea de inducción

- Aplicación del algoritmo para construir el árbol de decisión

E = Ev = {E3, E7, E12, E13}

Lista_Atributos = {Temperatura, Humedad, Viento}

```
    PROCEDIMIENTO Inducir_Arbol (Ejemplos E, Lista_Atributos, Método_Selección_Atributos)
    COMIENZO
P1  Crear un nodo N;
```

N

```
P2  SI todos los elementos de E pertenecen a la misma clase, C
    ENTONCES Retornar N como nodo hoja etiquetado con la clase C,
```

VERDADERO
C = Sí

Id	Ambiente	Temperatura	Humedad	Viento	*Clase*
E3	nublado	Alta	Alta	Falso	Sí
E6	Lluvioso	Baja	Normal	Verdadero	No
E7	Nublado	Baja	Normal	Verdadero	Sí
E12	Nublado	Media	Alta	Verdadero	Sí
E13	Nublado	Alta	Normal	Falso	Sí

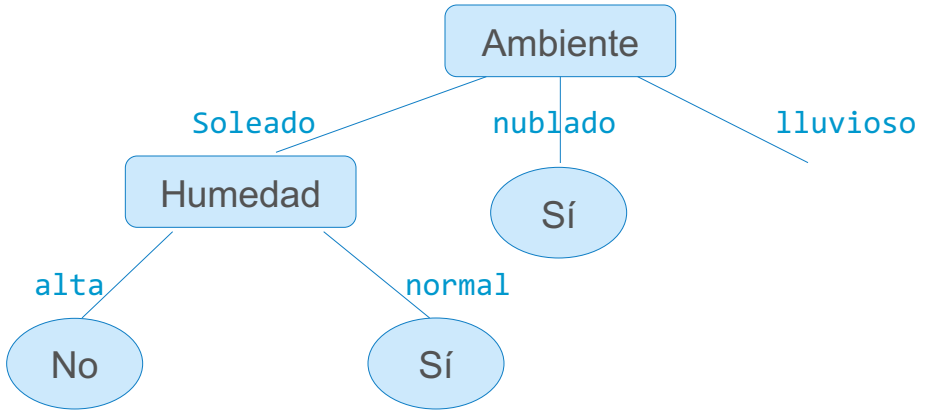
Retornar

Sí

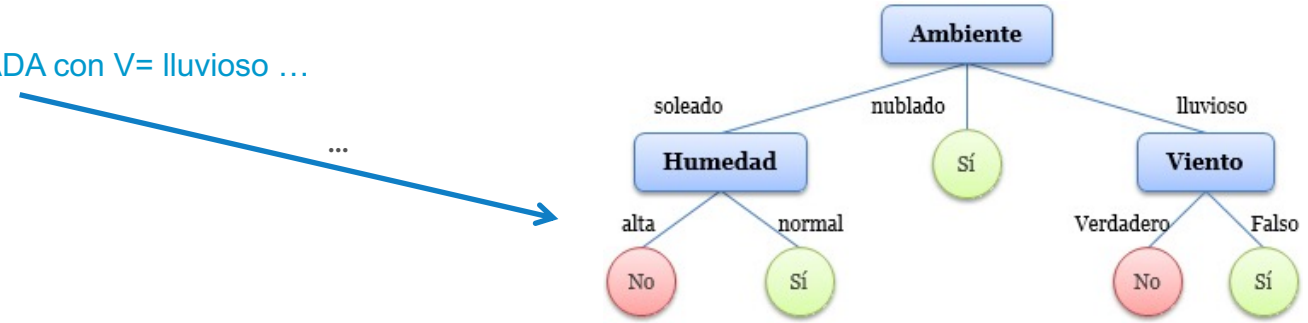
Descripción de la tarea de inducción

- Aplicación del algoritmo para construir el árbol de decisión

P9 `SI_NO unir al nodo N el nodo retornado de` *`Inducir_Arbol`* `(Ev, Lista_Atributos, Método_Selección_Atributos)`



Continuamos en PARA CADA con V= lluvioso …



Algoritmo básico de aprendizaje de árboles de decisión: ID3

- Algoritmo del tipo «**divide-y-vencerás»**
- **Método codicioso** (***greedy***): sin retroceso
- Construcción del árbol de arriba abajo
- Método de selección de atributos **basado en la teoría de la información**
  - El atributo cuyo conocimiento aporta mayor información en la clasificación es el más útil.
  - Método de selección de atributos: **Ganancia de Información** (métrica basada en la **entropía -** Indice Gini, Ganancia de la información).

Algoritmo básico de aprendizaje de árboles de decisión: ID3

- **Entropía:** caracteriza la heterogeneidad de un conjunto de ejemplos.
  - Entropía del conjunto de ejemplos E respecto a la clase C

$$Entropía(E) = \sum_{i=1}^{n} -p_i log_2 p_i$$

n número de valores que puede tomar la clase C

p_i la proporción de ejemplos de E que pertenecen a la clase i

  - $Entropía(E) = 0$ si todos los miembros del conjunto E pertenecen a la misma clase.
  - $Entropía(E) = 1$ si se tiene un mismo número de ejemplos positivos y negativos (clasificación binaria)

Algoritmo básico de aprendizaje de árboles de decisión: ID3

- **Ganancia de información:** *mide* la efectividad de un atributo para clasificar ejemplos.
  - Específicamente mide *la reducción de entropía al distribuir los ejemplos de acuerdo a los valores de un atributo*
  - Siendo un atributo A con V_a posibles valores y un conjunto de ejemplos E

$$Ganancia\ (E, A) = \ Entropía\ (E) - \sum_{v \in Va} \frac{|E_v|}{E} Entropia(E_v)$$

Algoritmo básico de aprendizaje de árboles de decisión: ID3

- Ejemplo: Método de selección de atributos

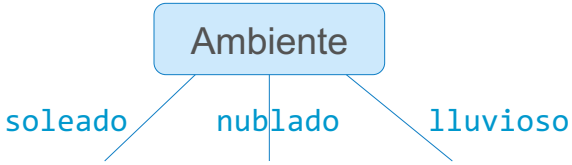
E = {E1, E2, … E14}

Lista_Atributos = {Ambiente, Temperatura, Humedad, Viento}

```
P4  SI_NO Aplicar Método_Selección_Atributos(E, Lista_Atributos) para seleccionar el
        atributo A que mejor particiona E
```

A = Ambiente

Id	Ambiente	Temperatura	Humedad	Viento	*Clase*
E1	soleado	Alta	Alta	Falso	No
E2	soleado	Alta	Alta	Verdadero	No
E3	nublado	Alta	Alta	Falso	Sí
E4	Lluvioso	Media	Alta	Falso	Sí
E5	Lluvioso	Baja	Normal	Falso	Sí
E6	Lluvioso	Baja	Normal	Verdadero	No
E7	Nublado	Baja	Normal	Verdadero	Sí
E8	Soleado	Media	Alta	Falso	No
E9	Soleado	Baja	Normal	Falso	Sí
E10	Lluvioso	Media	Normal	Falso	Sí
E11	Soleado	Media	Normal	Verdadero	Sí
E12	Nublado	Media	Alta	Verdadero	Sí
E13	Nublado	Alta	Normal	Falso	Sí
E14	lluvioso	Media	alta	Verdadero	No



$$Ganancia\ (E, A) = Entropía\ (E) - \sum_{v \in Va} \frac{|E_v|}{E} Entropia(E_v)$$

```
Ganancia (E, Ambiente) = max [
     Ganancia (E, Ambiente),
     Ganancia (E, Temperatura),
     Ganancia (E, Humedad),
     Ganancia (E, Viento) ]
```

Algoritmo básico de aprendizaje de árboles de decisión: ID3

- Ejemplo: Método de selección de atributos

Id	Ambiente	Temperatura	Humedad	Viento	*Clase*
E1	soleado	Alta	Alta	Falso	No
E2	soleado	Alta	Alta	Verdadero	No
E3	nublado	Alta	Alta	Falso	Sí
E4	Lluvioso	Media	Alta	Falso	Sí
E5	Lluvioso	Baja	Normal	Falso	Sí
E6	Lluvioso	Baja	Normal	Verdadero	No
E7	Nublado	Baja	Normal	Verdadero	Sí
E8	Soleado	Media	Alta	Falso	No
E9	Soleado	Baja	Normal	Falso	Sí
E10	Lluvioso	Media	Normal	Falso	Sí
E11	Soleado	Media	Normal	Verdadero	Sí
E12	Nublado	Media	Alta	Verdadero	Sí
E13	Nublado	Alta	Normal	Falso	Sí
E14	lluvioso	Media	alta	Verdadero	No

$$Entropía(E) = \sum_{i=1}^{n} -p_i log_2 p_i$$

9 veces la clase Sí en E = {E1, E2, … E13}

5 veces la clase No en E = {E1, E2, … E13}

$$Entropía(E) = -\frac{9}{14} log_2 \frac{9}{14} - \frac{5}{14} log_2 \frac{5}{14} = +0{,}94$$

$$Ganancia\ (E, A) = Entropía\ (E) - \sum_{v \in Va} \frac{|E_v|}{E} Entropia(E_v)$$

Algoritmo básico de aprendizaje de árboles de decisión: ID3

- Ejemplo: Método de selección de atributos

$$Ganancia\ (E, A) = Entropía\ (E) - \sum_{v \in Va} \frac{|E_v|}{E} Entropia(E_v)$$

A = ambiente Va = {soleado, nublado, lluvioso}

$$\text{Ganancia (E, ambiente)} = \text{Entropía(E)} - \frac{5}{14}\text{Entropia}(\text{E}_{\text{soleado}}) - \frac{4}{14}\text{Entropia}(\text{E}_{\text{nublado}}) - \frac{5}{14}\text{Entropia}(\text{E}_{\text{lluvioso}})$$

v = soleado

Ev = $E_{soleado}$= {E1, E2, E8, E9, E11}

Id	Ambiente	Temperatura	Humedad	Viento	Clase
E1	soleado	Alta	Alta	Falso	No
E2	soleado	Alta	Alta	Verdadero	No
E3	nublado	Alta	Alta	Falso	Sí
E4	Lluvioso	Media	Alta	Falso	Sí
E5	Lluvioso	Baja	Normal	Falso	Sí
E6	Lluvioso	Baja	Normal	Verdadero	No
E7	Nublado	Baja	Normal	Verdadero	Sí
E8	Soleado	Media	Alta	Falso	No
E9	Soleado	Baja	Normal	Falso	Sí
E10	Lluvioso	Media	Normal	Falso	Sí
E11	Soleado	Media	Normal	Verdadero	Sí
E12	Nublado	Media	Alta	Verdadero	Sí
E13	Nublado	Alta	Normal	Falso	Sí
E14	lluvioso	Media	alta	Verdadero	No

2 Sí en Ev 3 No en Ev

$$\text{Entropia}(E_{soleado}) = -\ 2/5 \log 2/5 - 3/5 \log 3/5 = 0{,}9709$$

Algoritmo básico de aprendizaje de árboles de decisión: ID3

- Ejemplo: Método de selección de atributos

$$Ganancia\ (E, A) = Entropía\ (E) - \sum_{v \in Va} \frac{|E_v|}{E} Entropia(E_v)$$

A = ambiente Va = {soleado, nublado, lluvioso}

$$\text{Ganancia (E, ambiente)} = \text{Entropía(E)} - \frac{5}{14}\text{Entropia}(E_{\text{soleado}}) - \frac{4}{14}\text{Entropia}(E_{\text{nublado}}) - \frac{5}{14}\text{Entropia}(E_{\text{lluvioso}})$$

v = nublado

Ev = $E_{nublado}$= {E3, E7, E12, E13}

Id	Ambiente	Temperatura	Humedad	Viento	*Clase*
E1	soleado	Alta	Alta	Falso	No
E2	soleado	Alta	Alta	Verdadero	No
E3	nublado	Alta	Alta	Falso	Sí
E4	Lluvioso	Media	Alta	Falso	Sí
E5	Lluvioso	Baja	Normal	Falso	Sí
E6	Lluvioso	Baja	Normal	Verdadero	No
E7	Nublado	Baja	Normal	Verdadero	Sí
E8	Soleado	Media	Alta	Falso	No
E9	Soleado	Baja	Normal	Falso	Sí
E10	Lluvioso	Media	Normal	Falso	Sí
E11	Soleado	Media	Normal	Verdadero	Sí
E12	Nublado	Media	Alta	Verdadero	Sí
E13	Nublado	Alta	Normal	Falso	Sí
E14	lluvioso	Media	alta	Verdadero	No

4 Sí en Ev 0 No en Ev

$$\text{Entropia}(E_{\text{nublado}}) = -\ 4/4 \log 4/4 - 0 \log 0 = 0$$

Algoritmo básico de aprendizaje de árboles de decisión: ID3

- Ejemplo: Método de selección de atributos

$$Ganancia\ (E, A) = Entropía\ (E) - \sum_{v \in Va} \frac{|E_v|}{E} Entropia(E_v)$$

A = ambiente Va = {soleado, nublado, lluvioso}

$$\text{Ganancia (E, ambiente)} = \text{Entropía(E)} - \frac{5}{14}\text{Entropia}(E_{\text{soleado}}) - \frac{4}{14}\text{Entropia}(E_{\text{nublado}}) - \frac{5}{14}\text{Entropia}(E_{\text{lluvioso}})$$

v = lluvioso

Ev = $E_{lluvioso}$= {E4, E5, E6, E10, E14}

Id	Ambiente	Temperatura	Humedad	Viento	*Clase*
E1	soleado	Alta	Alta	Falso	No
E2	soleado	Alta	Alta	Verdadero	No
E3	nublado	Alta	Alta	Falso	Sí
E4	Lluvioso	Media	Alta	Falso	Sí
E5	Lluvioso	Baja	Normal	Falso	Sí
E6	Lluvioso	Baja	Normal	Verdadero	No
E7	Nublado	Baja	Normal	Verdadero	Sí
E8	Soleado	Media	Alta	Falso	No
E9	Soleado	Baja	Normal	Falso	Sí
E10	Lluvioso	Media	Normal	Falso	Sí
E11	Soleado	Media	Normal	Verdadero	Sí
E12	Nublado	Media	Alta	Verdadero	Sí
E13	Nublado	Alta	Normal	Falso	Sí
E14	lluvioso	Media	alta	Verdadero	No

3 Sí en Ev 2 No en Ev

$$\text{Entropia}(E_{\text{lluvioso}}) = -\ 3/5 \log 3/5 - 2/5 \log 2/5 = 0{,}9709$$

Algoritmo básico de aprendizaje de árboles de decisión: ID3

- Ejemplo: Método de selección de atributos

$$Ganancia\ (E, A) = Entropía\ (E) - \sum_{v \in Va} \frac{|E_v|}{E} Entropia(E_v)$$

A = ambiente Va = {soleado, nublado, lluvioso}

$$\text{Ganancia (E, ambiente)} = \text{Entropía(E)} - \frac{5}{14}\text{Entropia}(E_{soleado}) - \frac{4}{14}\text{Entropia}(E_{nublado}) - \frac{5}{14}\text{Entropia}(E_{lluvioso})$$

$\text{Entropia(E)} = 0{,}9402$

$\text{Entropia}(E_{soleado}) = 0{,}9709$

$\text{Entropia}(E_{nublado}) = 0$

$\text{Entropia}(E_{lluvioso}) = 0{,}9709$

$$\text{Ganancia (E, ambiente)} = +0{,}9402 - \frac{5}{14}0{,}9709 - \frac{4}{14}0 - \frac{5}{14}0{,}9709 = 0{,}247$$

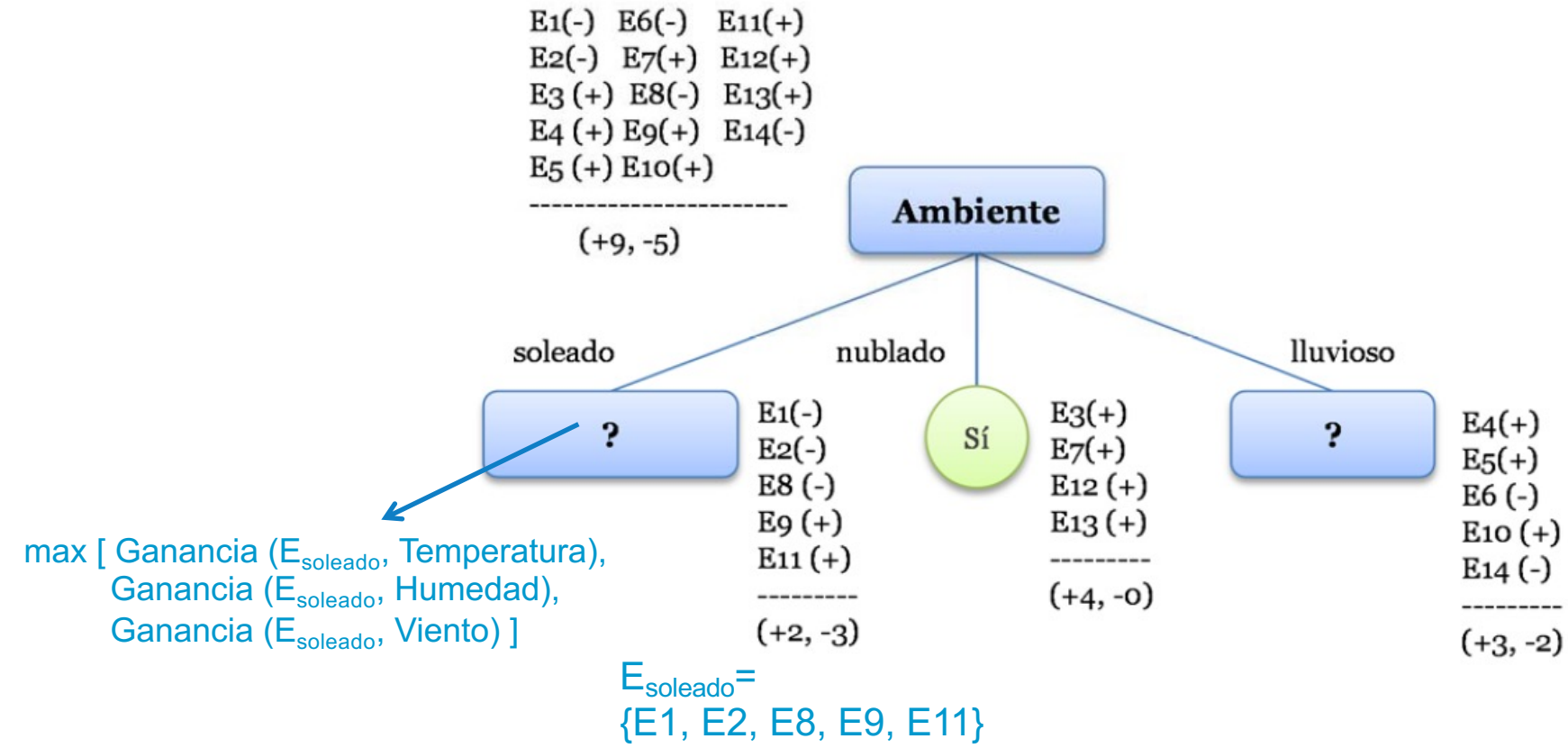
A = temperatura Va = {alta, media, baja}
→ Ganancia (E, Temperatura) = 0,029
A = humedad Va = {alta, normal}
→ Ganancia (E, Humedad) = 0,151
A = viento Va = {verdadero, falso}
→ Ganancia (E, Viento) = 0,048

```
Ganancia (E, Ambiente) = max [
     Ganancia (E, Ambiente),
     Ganancia (E, Temperatura),
     Ganancia (E, Humedad),
     Ganancia (E, Viento) ]
```

Ambiente

Algoritmo básico de aprendizaje de árboles de decisión: ID3

- Ejemplo: Método de selección de atributos



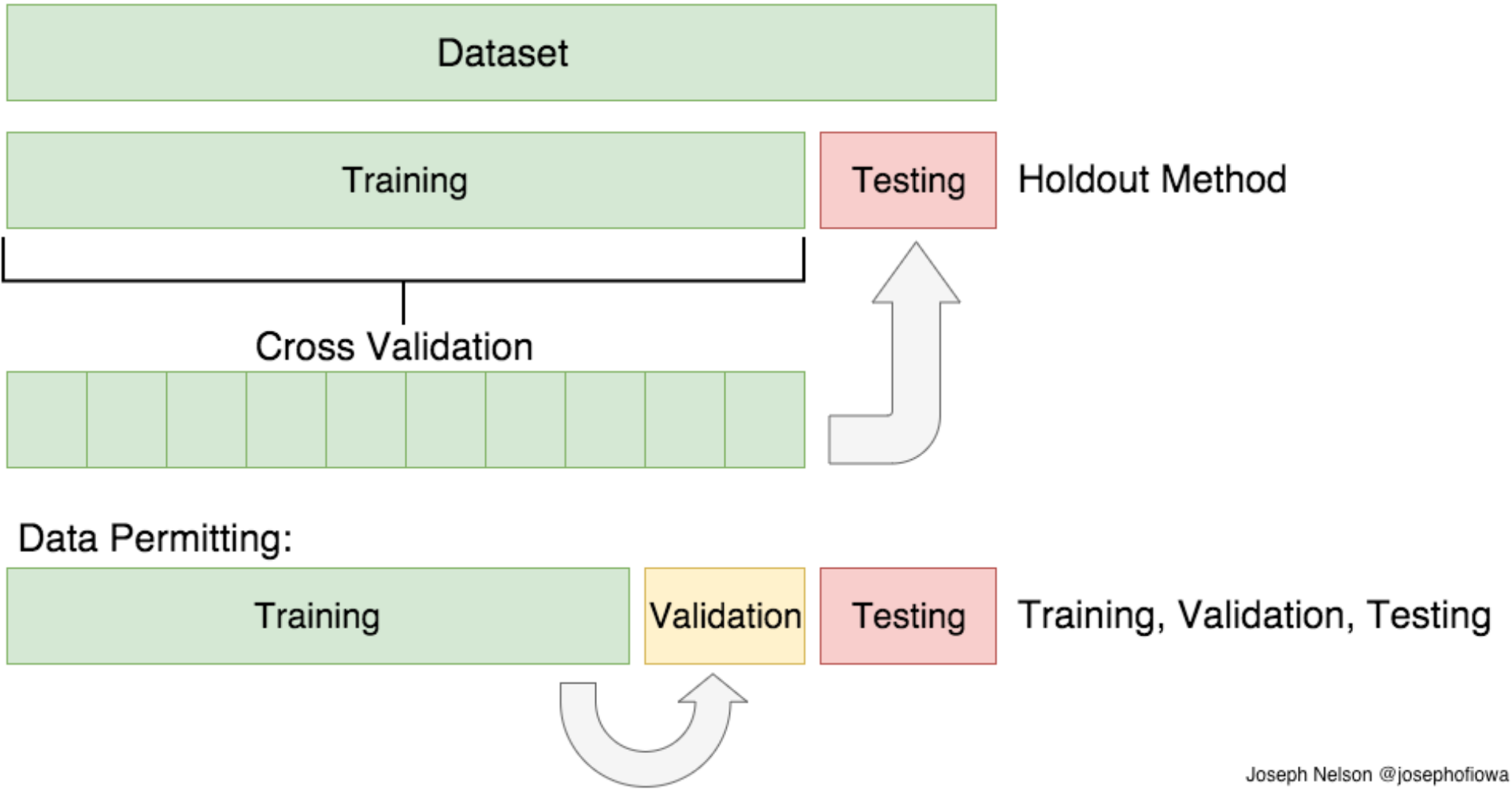
Espacio de búsqueda y *bias* inductivo

- Problema de construcción de árboles de decisión: búsqueda por el espacio de hipótesis hasta encontrar el árbol que encaja con los ejemplos de entrenamiento.
- Ventajas de ID3:
  - Trabaja en un espacio de hipótesis completo.
  - Robusto frente a errores.
- Inconvenientes de ID3:
  - No trabaja con varias soluciones simultáneamente → riesgo de construir el árbol que no es la mejor solución.
  - No da marcha atrás → riesgo de converger hacia una solución óptima local.
  - Más carga computacional que una solución incremental.

Espacio de búsqueda y *bias* inductivo

- **Bías Inductivo**
  - ¿En qué se basan los algoritmos para generalizar el árbol de decisión? Es decir, para considerar que el árbol clasificará correctamente instancias no utilizadas en la etapa de aprendizaje.
  - Conjunto de todos los Factores que permiten Realizar y Seleccionar las Hipótesis más adecuadas.
- **Bías Inductivo en ID3**
  - Preferencia por árboles cortos frente a largos.
  - Preferencia por árboles que sitúan los atributos de mayor. ganancia de información cerca de la raíz.
  - El preferir árboles cortos puede mejorar la generalización, puesto que hay hipótesis complejas que encajan muy bien con los datos de entrenamiento pero que no generalizan correctamente datos futuros.

Entrenamiento y validación

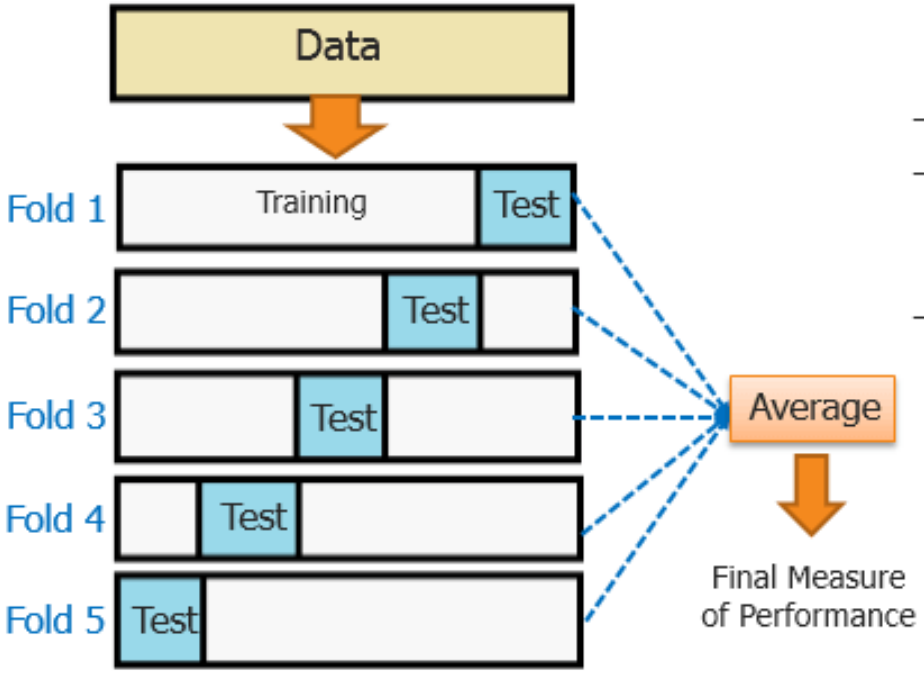


Fuente: https://bookdown.org/content/2031/arboles-de-decision-parte-i.html#conceptos-introductorios

Entrenamiento y validación

- **Validación cruzada (*cross-validation*)**
  - Estimar el ajuste del modelo a un hipotético conjunto de datos de prueba cuando no se dispone de este conjunto de datos de prueba de manera explícita.
  - Dividir el conjunto de ejemplos disponibles en un conjunto de datos de entrenamiento y un conjunto de datos de validación:
    - Datos de entrenamiento: se utilizan para generar el árbol.
    - Datos de validación: se utilizan para validar la precisión del árbol generado sobre datos futuros.
- **Validación cruzada de k iteraciones (*k-fold cross-validation*)**
  - Dividir los datos en k subconjuntos de igual tamaño
  - Utilizar un subconjunto como datos de prueba (o validación)
  - Utilizar k-1 subconjuntos como datos de entrenamiento
  - Repetir la validación cruzada k veces
  - Realizar la media de los resultados

Entrenamiento y validación

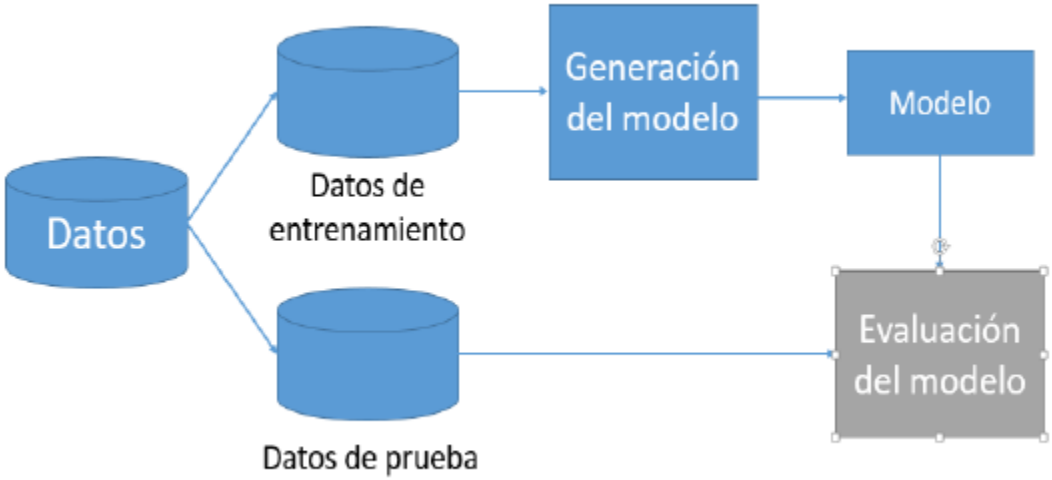


- → Technique to validate models/classifiers
- → Method to estimate how accurately the model generalizes to unseen data i.e., how well it performs/predicts
- → K-fold CV
  - » Most popular
  - » k is typically set to 10
  - » Every sample/record is used both in training and test sets

Fuente: https://bookdown.org/content/2031/arboles-de-decision-parte-i.html#conceptos-introductorios

Sobreajuste y poda de árboles

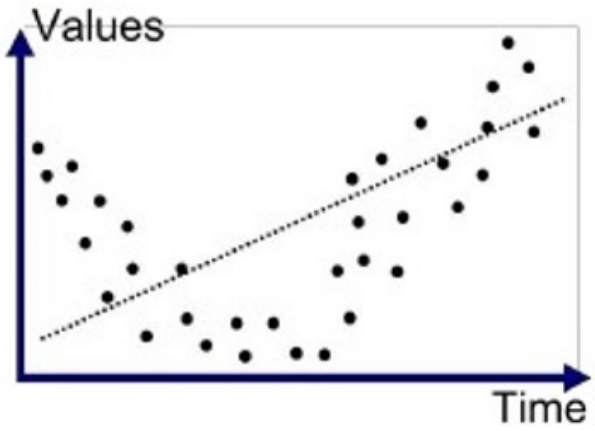
- **¿Cómo se puede saber cuál es el tamaño de árbol adecuado?**
  - La etapa de validación permite evaluar la efectividad de la poda para clasificar instancias futuras.



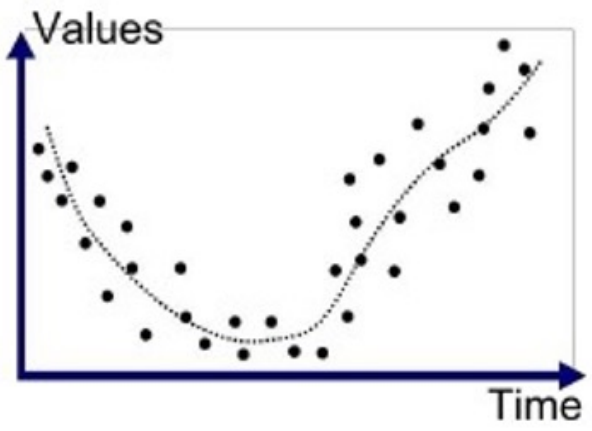
Fuente: Elena Verdú, UNIR

Sobreajuste y poda de árboles

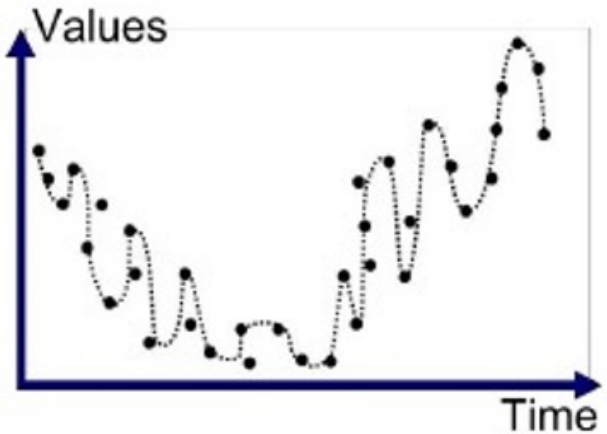
- **Sobreajuste - Overfitting**



Underfitted



Good Fit/Robust

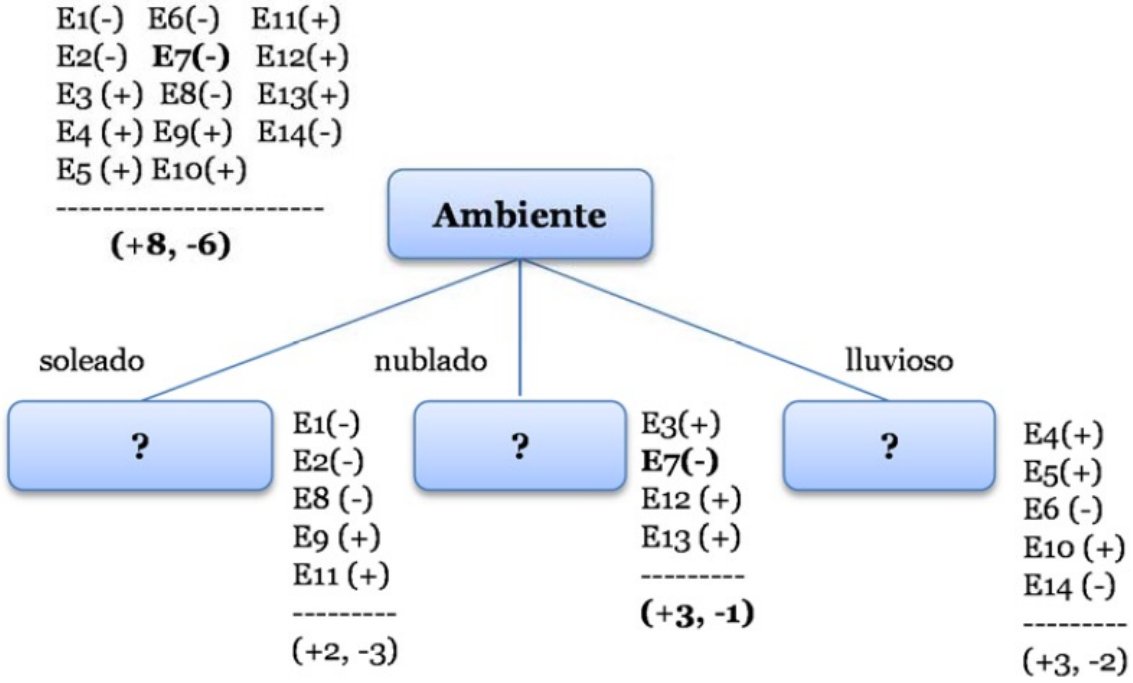


Overfitted

Fuente: https://bookdown.org/content/2031/arboles-de-decision-parte-i.html#conceptos-introductorios

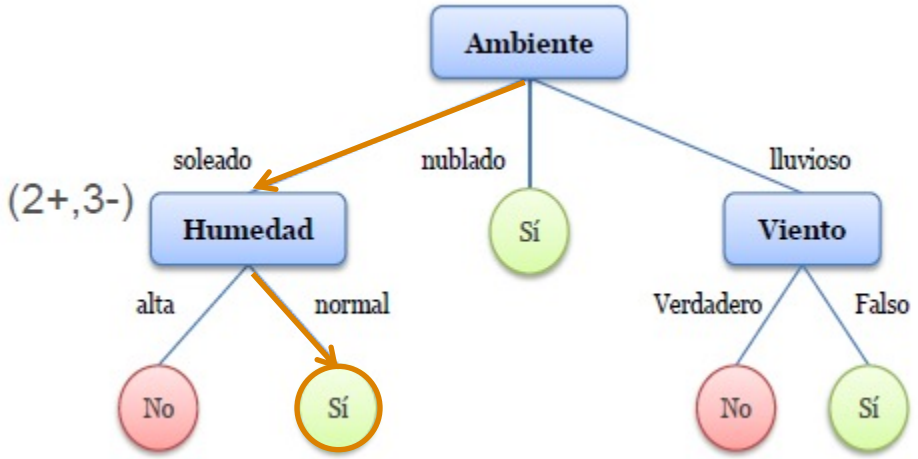
Sobreajuste y poda de árboles

- **Sobreajuste**
  - Existe una hipótesis H del espacio de hipótesis que se ajusta mejor a los datos de entrenamiento que otra hipótesis H' del espacio de hipótesis pero, sin embargo, H' se ajusta mejor a todas las instancias (comprendiendo datos de entrenamiento e instancias futuras).



Sobreajuste y poda de árboles

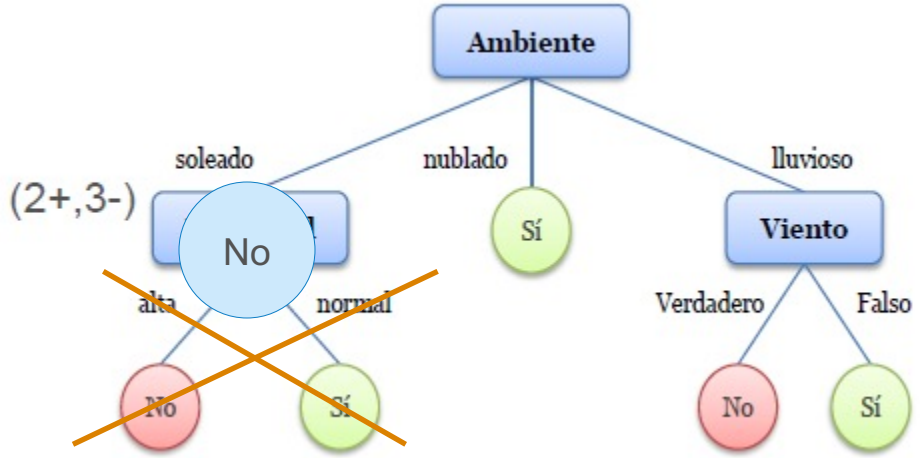
- Ejemplo:
  - Se tiene la instancia (Ambiente = soleado, Temperatura = alta, Humedad = normal, Viento = falso, jugar=no).



  - El árbol clasifica a esta instancia incorrectamente

Sobreajuste y poda de árboles

- Mediante simplificación puedo evitar el sobreajuste.
  - ¿Y si podamos el nodo “Humedad”?
  - El nodo humedad se convertiría en hoja etiquetada con la clase mayoritaria: No



  - Ahora el árbol clasificaría correctamente la instancia.

Sobreajuste y poda de árboles

- Estrategias para evitar sobreajuste

Pospoda	Prepoda
Podar el árbol una vez generado. Se pueden llegar a tener en cuenta combinaciones de atributos antes de realizar la poda.	Limitar el crecimiento del árbol. Tieen la ventaja de que ahorra los costes de procesamiento debidos a generar nodos y ramas que posteriormente serían podados (utilizando un método de pospoda).

- **Pospoda de reglas**
  - Generación del árbol de decisión a partir de los datos de entrenamiento.
  - Conversión del árbol en un conjunto de reglas.
  - Poda cada regla eliminando las condiciones en el antecedente que suponen mejorar la precisión en la clasificación.
  - Ordenar las reglas en función de la precisión estimada
    - Las futuras instancias se clasifican en función de la primera regla que satisfacen, ordenadas en ese orden.

Sobreajuste y poda de árboles

- **Poda de reglas**
  - La decisión de poda se realiza de manera aislada para cada camino que pasa por un nodo
  - Poda indiferente de si el atributo se encuentra cerca de la raíz o de la hoja

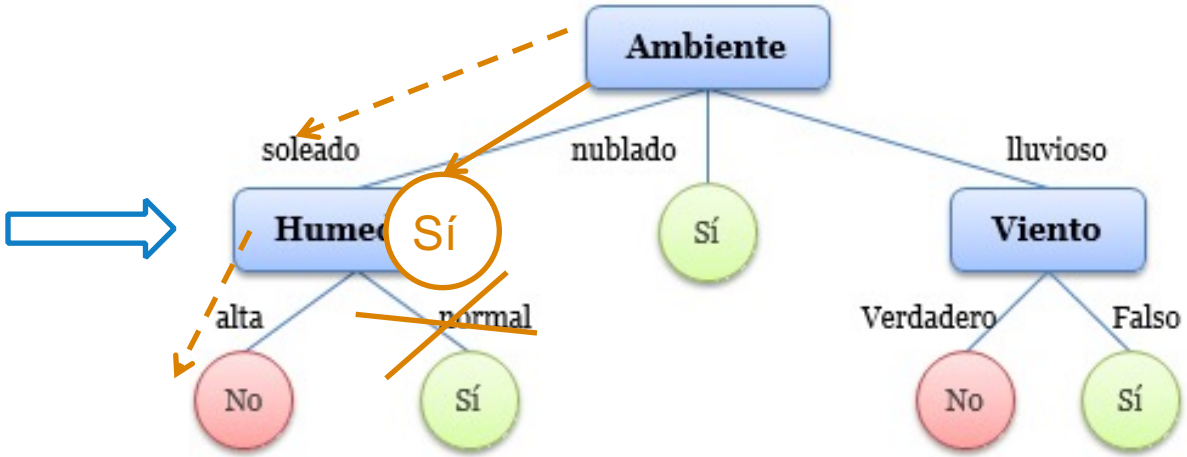
SI ambiente es soleado
~~*AND humedad es normal*~~
ENTONCES jugar = Sí

SI ambiente es nublado
ENTONCES jugar = Sí

SI ambiente es lluvioso
AND viento es falso
ENTONCES jugar = Sí

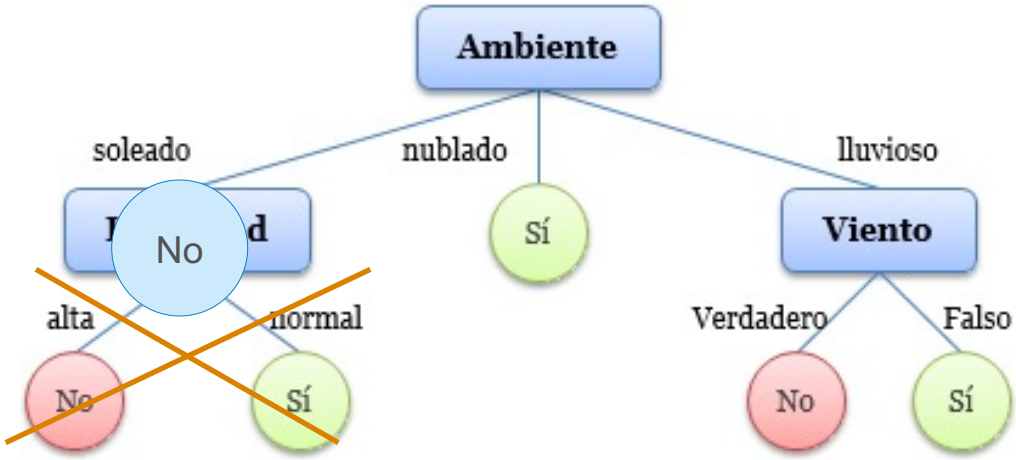
SI ambiente es soleado
AND humedad es alta
ENTONCES jugar = No

SI ambiente es lluvioso
AND viento es verdadero
ENTONCES jugar = No



Sobreajuste y poda de árboles

- **Poda de árboles**
  - La eliminación de un nodo de un árbol supone eliminar diferentes reglas



SI ambiente es soleado
AND humedad es normal
ENTONCES jugar = Sí

SI ambiente es nublado
ENTONCES jugar = Sí

SI ambiente es lluvioso
AND viento es falso
ENTONCES jugar = Sí

SI ambiente es soleado
AND humedad es alta
ENTONCES jugar = No

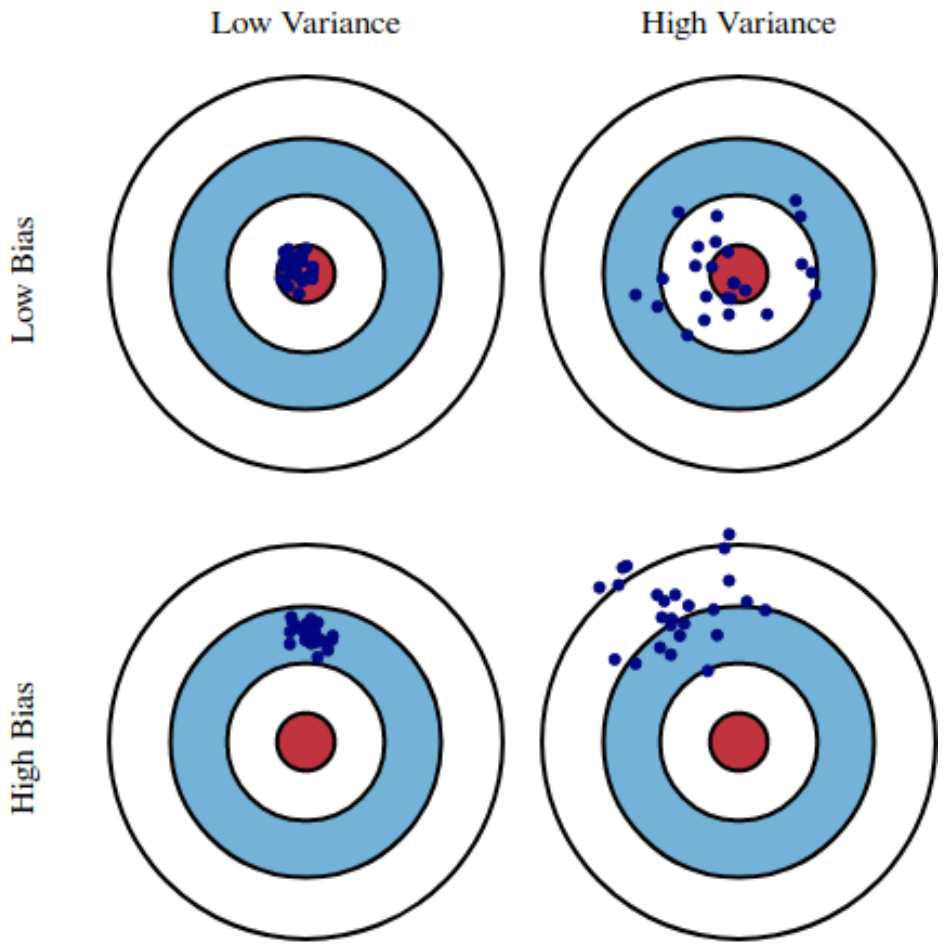
SI ambiente es lluvioso
AND viento es verdadero
ENTONCES jugar = No

SI ambiente es soleado
ENTONCES jugar = No

Simplificación de árboles de decisión mediante poda: algoritmo C4.5

- Características:
  - Método de aprendizaje de árboles de decisión basado en ID3
  - Se puede aplicar a datos con atributos de valores tanto discretos como continuos
  - Puede trabajar con datos ausentes
  - Método de selección de atributos: medida de **proporción de ganancia** (basada en la ganancia de información y proporciona mejores resultados si los atributos tienen muchos posibles valores)
- Realiza una poda tras la generación del árbol (pospoda) con el fin de mejorar la generalización del modelo: elimina nodos que al podar mejoran la precisión en la clasificación.

Sesgo (*bias*) *y* varianza



Fuente: https://bookdown.org/content/2031/arboles-de-decision-parte-i.html#conceptos-introductorios

Medidas de la precisión de la clasificación

- Se utiliza un conjunto de ejemplos de los disponibles (datos de prueba) para evaluar la hipotética efectividad de la clasificación de instancias futuras por un árbol de decisión inducido.
- **¿Cómo podemos estimar la precisión real en la clasificación de futuras instancias?**
  - Se confía más en una clasificación inducida a partir de un conjunto más numeroso de datos de entrenamiento
  - Se confía más en una validación que utiliza un numeroso conjunto de datos de prueba
  - Utilizando N=1000 ejemplos de prueba de los cuales 750 están bien clasificados, se tiene una tasa de éxito t_e de 0.75. ¿Cómo se puede extrapolar este hecho a futuras instancias?
    - Con un 80% de confianza la tasa de éxito real se encontrará en el intervalo [0.732, 0.767].

Medidas de la precisión de la clasificación

- **Matriz de confusión:**

		Predicción	
		Positivos	**Negativos**
Observación	**Positivos**	Verdaderos Positivos (VP)	Falsos Negativos (FN)
	Negativos	Falsos Positivos (FP)	Verdaderos Negativos (VN)

- En inglés: TP, FN, FP, TN

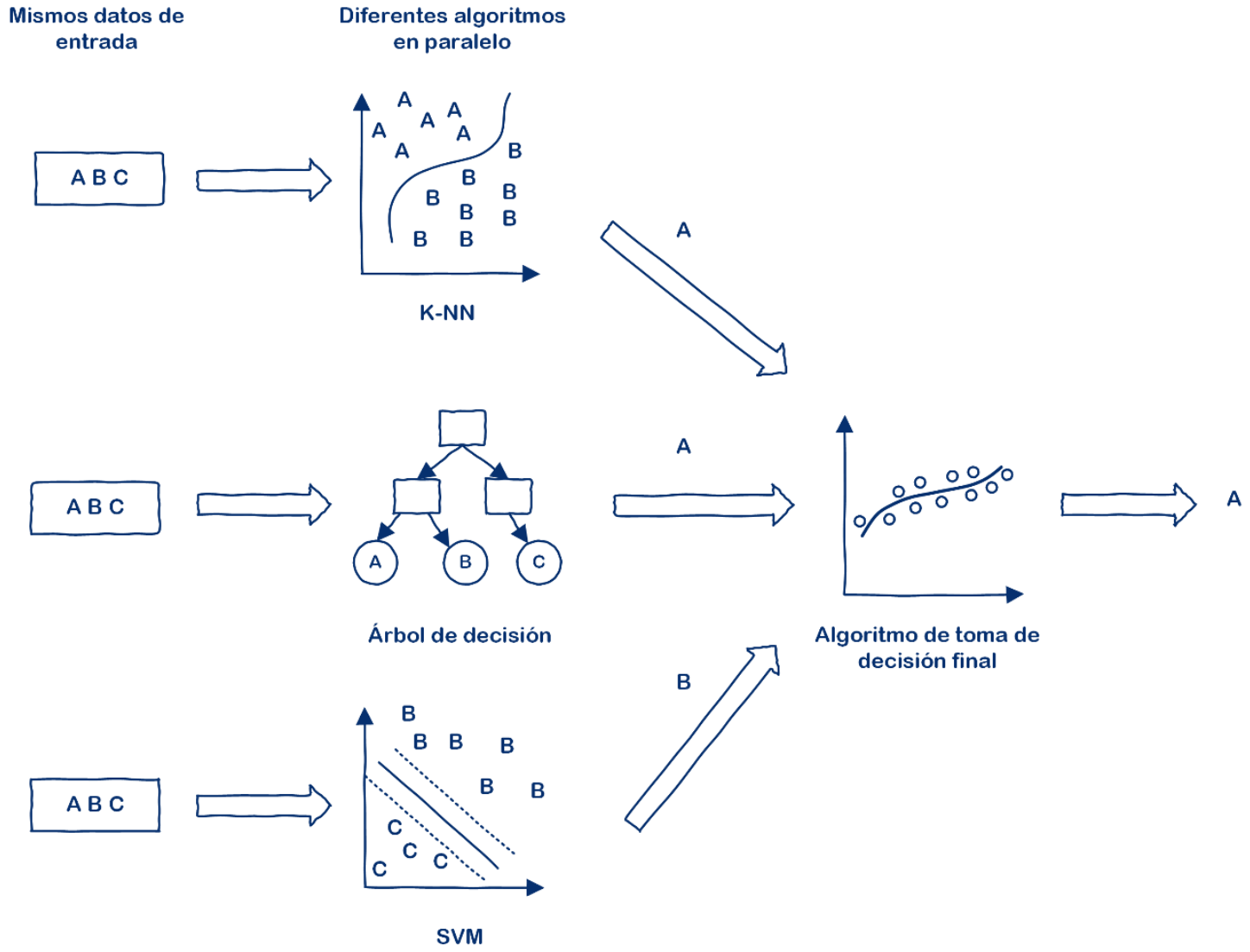
Medidas de la precisión de la clasificación

- **Exactitud (*accuracy*) ACC**
  - ACC = (TP + TN) / (P + N)
- **Sensibilidad** o Razón de Verdaderos Positivos (TPR)
  - TPR = TP / P = TP / (TP + FN)
- **Especificidad (*SPC*)** o Razón de Verdaderos Negativos (TNR)
  - SPC = TN / N = TN / (FP + TN)
- **Razón de falsos positivos (FPR) = 1 - SPC**
  - FPR = FP / N = FP / (FP + TN) = 1 - SPC

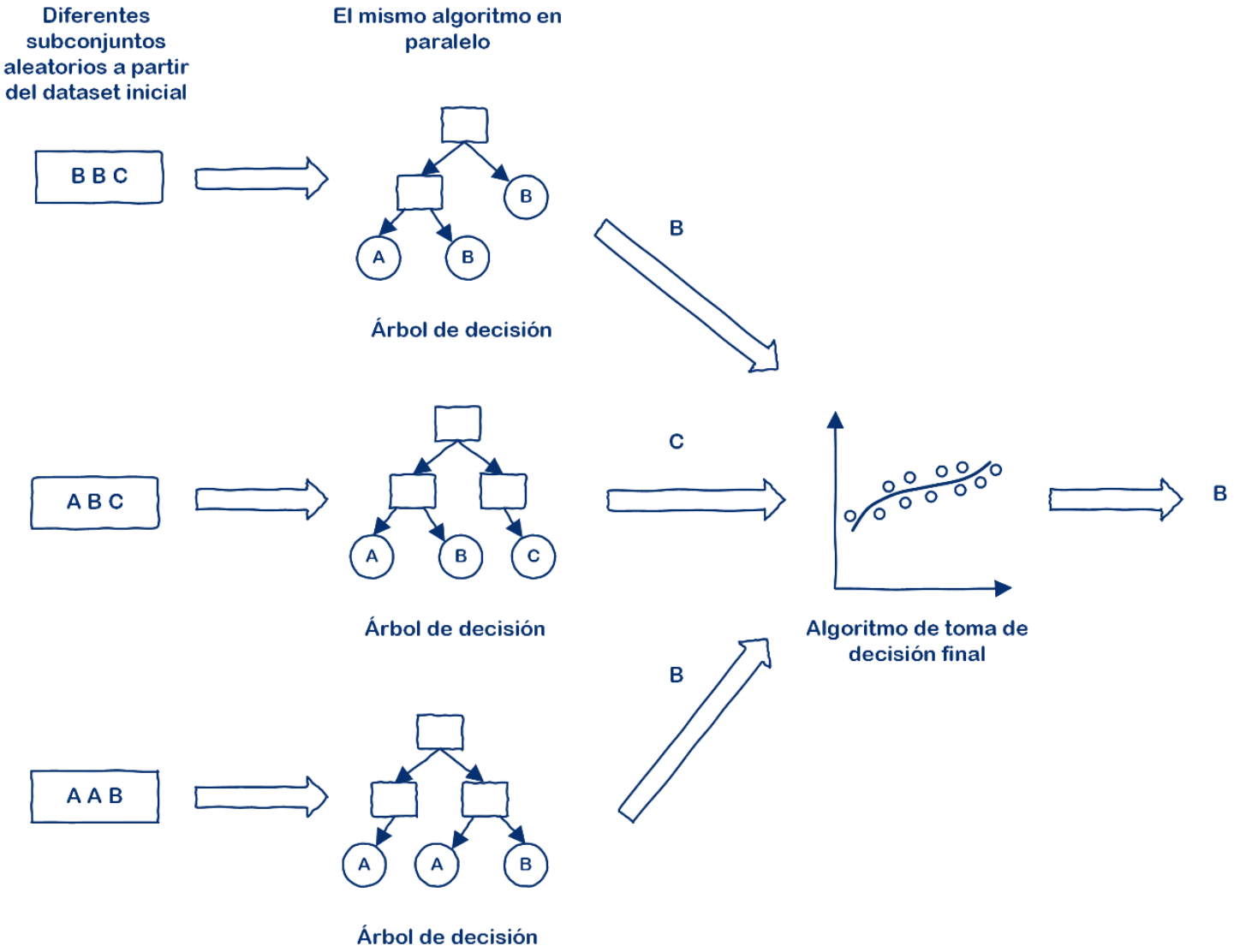
Ensemble learning o aprendizaje integrado

- Actualmente, los métodos más modernos y maduros utilizados para obtener los resultados más precisos en entornos de producción, además de las redes neuronales, son los métodos ensemble learning.
- En los métodos de aprendizaje integrado, la idea es **unir un conjunto de algoritmos ineficientes en los que cada uno colabora corrigiendo los errores del resto del conjunto.**
- De esta manera, se consigue una **calidad general más alta que la de los mejores algoritmos individuales que trabajan de forma aislada**.

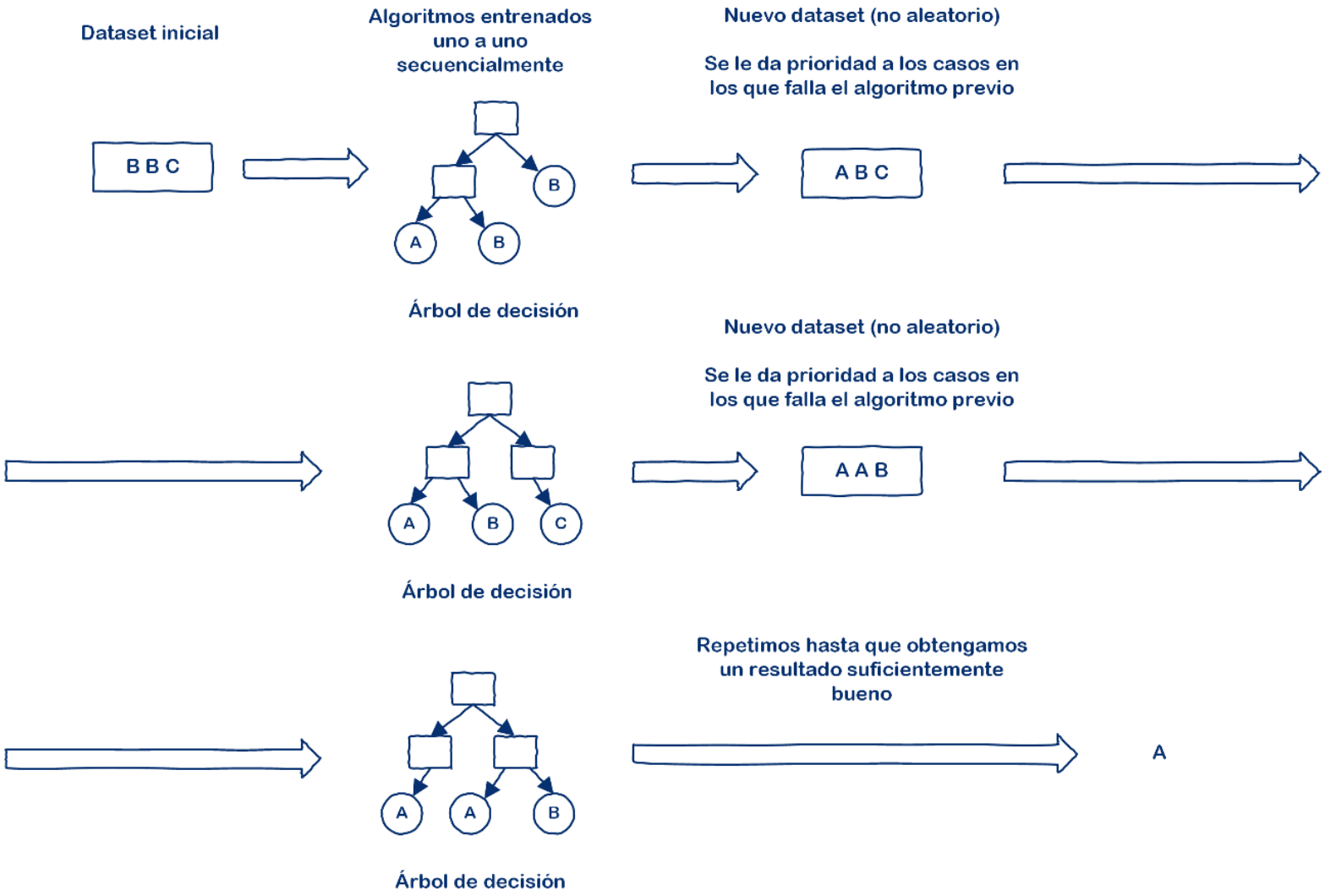
Ensemble learning: *stacking*



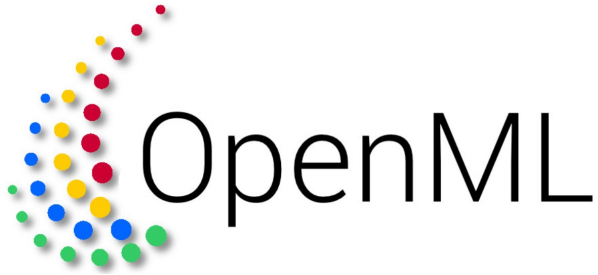
Ensemble learning: *bagging* (ej: *random forest*)



Ensemble learning: *boosting*



Ejemplos con Python



- Tomaremos el dataset **iris de Fisher**
  - https://www.openml.org/d/61
  - https://www.openml.org/data/get_csv/61/dataset_61_iris.arff
  - Cuatro características
    - Longitud y anchura de pétalo
    - Longitud y anchura de sépalo
  - Tres tipos de flores (clases)
    - *Iris setosa (50)*
    - *Iris virginica (50)*
    - *Iris versicolor (50)*

Largo de sépalo	Ancho de sépalo	Largo de pétalo	Ancho de pétalo	Especies
5.1	3.5	1.4	0.2	I. setosa
4.9	3.0	1.4	0.2	I. setosa
4.7	3.2	1.3	0.2	I. setosa

Ejemplos con Python

- **`pip install pandas`**
- **`pip install scikit-learn`**
  - CART optimizado
  - Aunque sólo para variables numéricas
- **`pip install matplotlib`**
- **`pip install graphviz`**
  - https://www.graphviz.org/download/
  - También será necesario incluir la ruta al ejecutable en nuestro PATH, por ejemplo, en Windows 10 esto se realizaría en Panel de Control → Sistema → Configuración avanzada del sistema → Variables de entorno, y ahí editar la variable PATH añadiendo, por ejemplo:
  - `C:\Program Files (x86)\Graphviz2.38\bin`
- **`pip install pydotplus`**

Ejemplos con Python

- Vamos a comparar un árbol de decisión con la regresión logística a la hora de clasificar las especies de flores
- La **regresión logística**, a pesar de su nombre, es un clasificador, no es una regresión
- La regresión logística se utiliza para dar una probabilidad entre 0 y 1 de que los datos de entrada correspondan a una clase
- En lugar de utilizar los mínimos cuadrados ordinarios que podrían dar probabilidades inferiores a 0 o superiores a 1, se utilizan ***modelos logit***, como la ***función sigmoidea***:

$$S(x) = \frac{1}{1 + e^{-x}}$$

Ejemplos con Python: explorando los datos

```
# Load libraries
from pandas import read_csv

url = "https://www.openml.org/data/get_csv/61/dataset_61_iris.arff"
# La siguiente línea no es necesaria usando esta fuente, pues ya incluye la
# cabecera
#names = ['sepallength', 'sepalwidth', 'petallength', 'petalwidth', 'class']
#dataset = read_csv(url, names=names)
dataset = read_csv(url)

# mostramos la "forma", debería haber 150 entradas con 5 atributos cada una
print(dataset.shape)
#> (150 , 5)

# mostramos las 3 primeras entradas para echar un vistazo
print(dataset.head(3))
#>   sepallength  sepalwidth  petallength  petalwidth        class
#>0          5.1         3.5          1.4         0.2  Iris-setosa
#>1          4.9         3.0          1.4         0.2  Iris-setosa
#>2          4.7         3.2          1.3         0.2  Iris-setosa
```

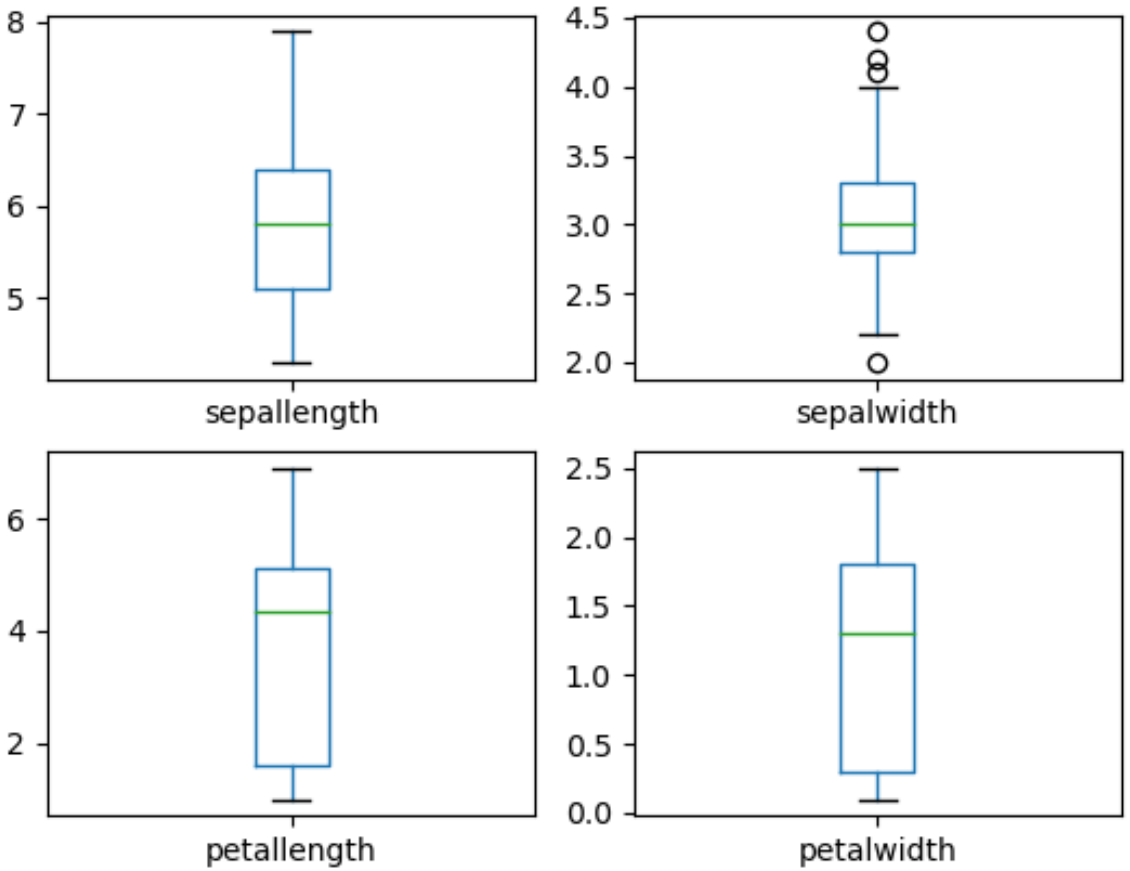
Ejemplos con Python: explorando los datos

```
# mostramos un resumen estadístico de los datos
print(dataset.describe())
#>       sepallength  sepalwidth  petallength  petalwidth
#>count   150.000000  150.000000   150.000000  150.000000
#>mean      5.843333    3.054000     3.758667    1.198667
#>std       0.828066    0.433594     1.764420    0.763161
#>min       4.300000    2.000000     1.000000    0.100000
#>25%       5.100000    2.800000     1.600000    0.300000
#>50%       5.800000    3.000000     4.350000    1.300000
#>75%       6.400000    3.300000     5.100000    1.800000
#>max       7.900000    4.400000     6.900000    2.500000

# distribución por clases
print(dataset.groupby('class').size())
#>Iris-setosa        50
#>Iris-versicolor    50
#>Iris-virginica     50
#>dtype: int64
```

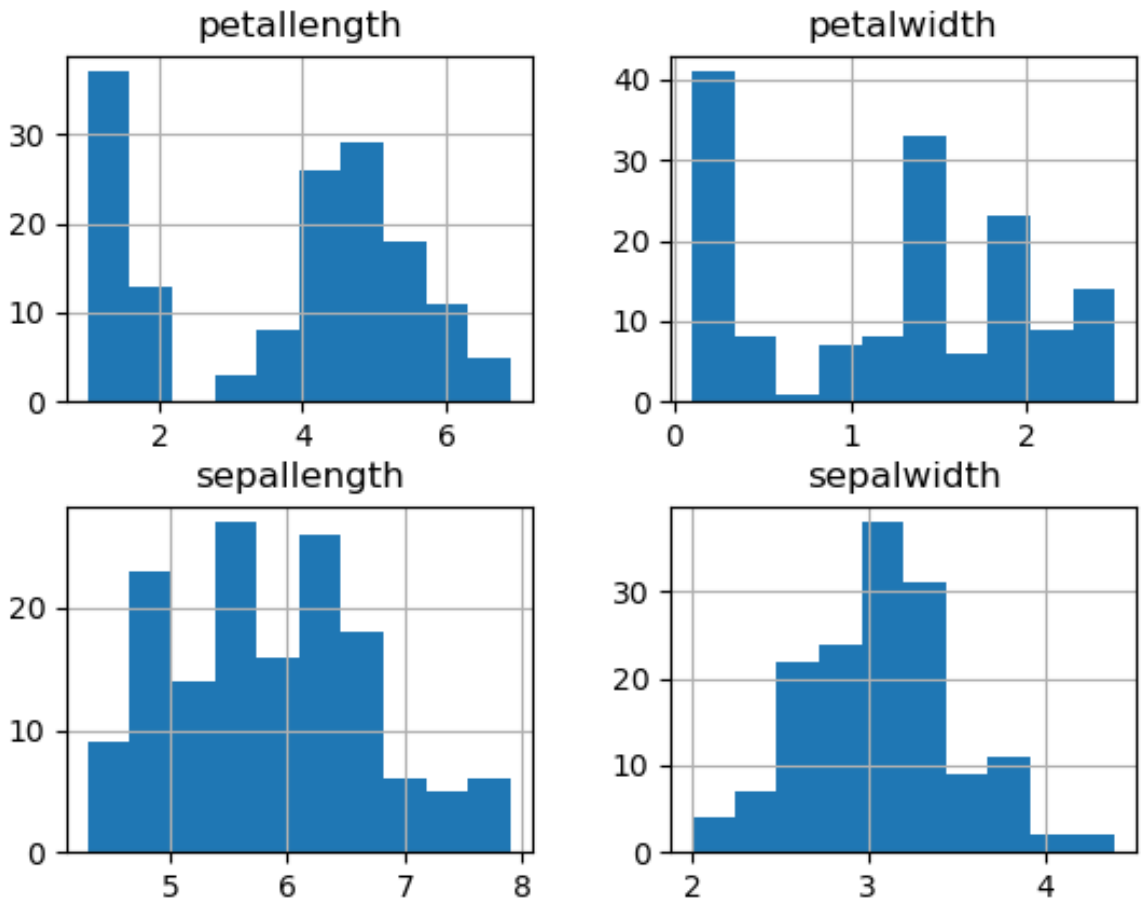
Ejemplos con Python: explorando los datos

```
# Load libraries
from pandas import read_csv
from pandas.plotting import scatter_matrix
from matplotlib import pyplot
url = "https://www.openml.org/data/get_csv/61/dataset_61_iris.arff"
# La siguiente línea no es necesaria usando esta fuente, pues ya incluye la
# cabecera
#names = ['sepallength', 'sepalwidth', 'petallength', 'petalwidth', 'class']
#dataset = read_csv(url, names=names)
dataset = read_csv(url)
# gráficos univariable:
 # diagramas de caja (box and whisker)
dataset.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=Fal
se)
pyplot.show()
 # histogramas
dataset.hist()
pyplot.show() # gráficos multivariable
 # matriz de dispersión

scatter_matrix(dataset)
pyplot.show()
```

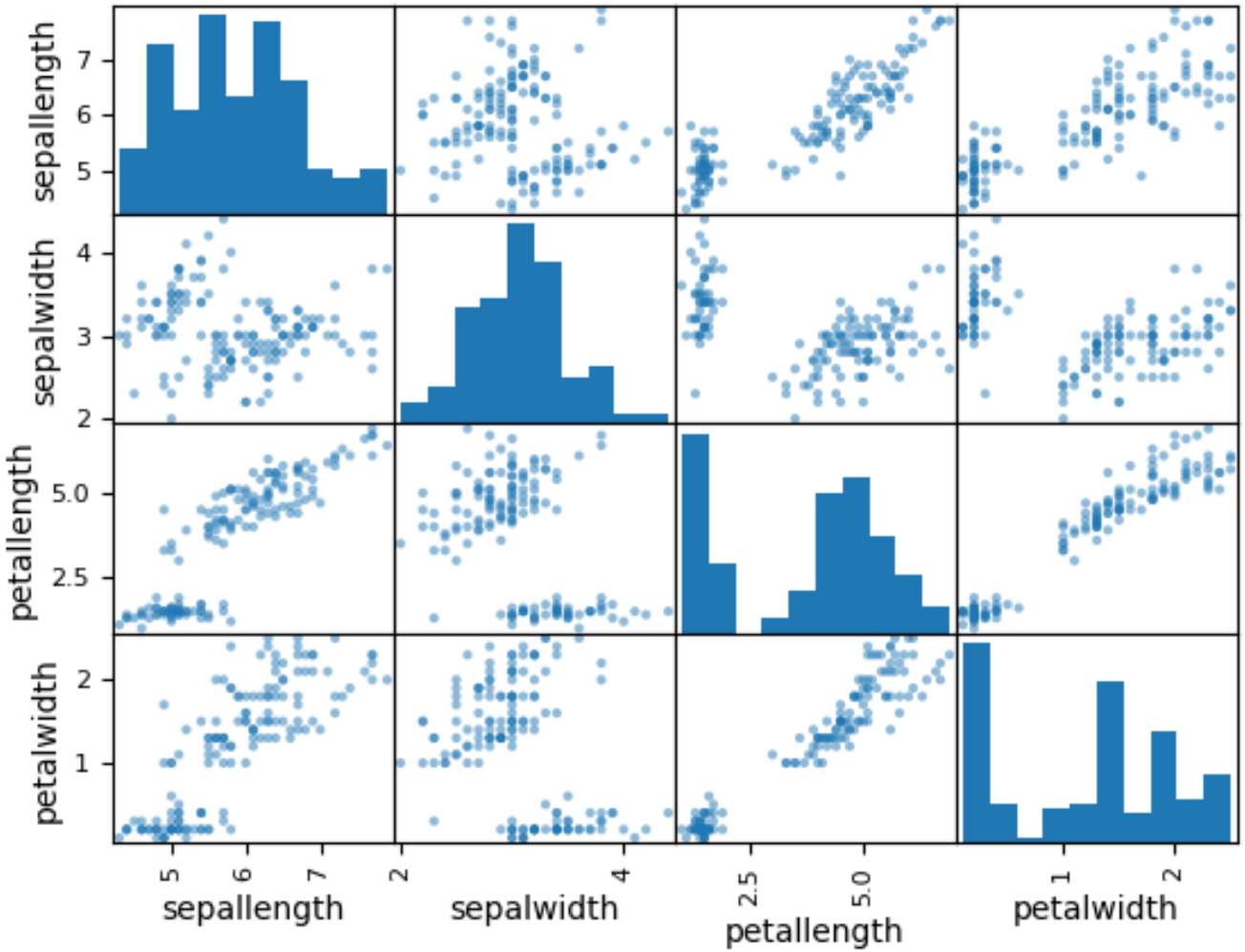
Ejemplos con Python: explorando los datos



Ejemplos con Python: explorando los datos



Ejemplos con Python: explorando los datos



Ejemplos con Python: comparativa de algoritmos

- **Para ello, separaremos nuestro dataset: usaremos un 80% de los datos para entrenar los algoritmos y un 20% de los datos para hacer los tests de predicción.**
  - **Ésta suele ser una proporción habitual.**

- Además, utilizaremos una **validación cruzada estratificada de 10 veces (*k-fold*)** para estimar la precisión del modelo.

Ejemplos con Python: comparativa de algoritmos

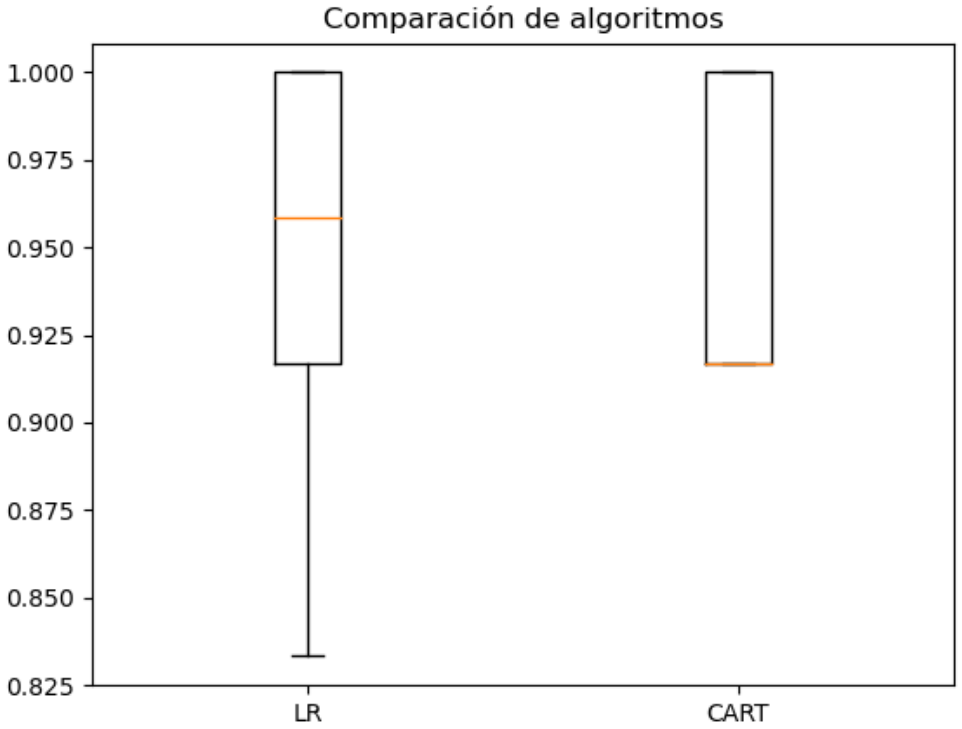
```
# Load libraries
from pandas import read_csv
from matplotlib import pyplot
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import StratifiedKFold
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier

# Cargamos el dataset
url = "https://www.openml.org/data/get_csv/61/dataset_61_iris.arff"
dataset = read_csv(url)

# Dividimos el dataset en 80% de datos para entrenar y 20% para test
array = dataset.values
X = array[:,0:4]
y = array[:,4]
X_train, X_validation, Y_train, Y_validation = train_test_split(X, y
, test_size=0.20, random_state=1, shuffle=True)
```

Ejemplos con Python: comparativa de algoritmos

```
# Cargamos los algoritmos
models = []
models.append(('LR', LogisticRegression(solver='liblinear', multi_cl
ass='ovr')))
models.append(('CART', DecisionTreeClassifier()))

# evaluamos cada modelo por turnos
results = []
names = []
for name, model in models:
    kfold = StratifiedKFold(n_splits=10, random_state=1)
    cv_results = cross_val_score(model, X_train, Y_train, cv=kfold,
scoring='accuracy')
    results.append(cv_results)
    names.append(name)
    print('%s: %f (%f)' % (name, cv_results.mean(), cv_results.std()
))
```

Ejemplos con Python: comparativa de algoritmos

```
# Comparación de algoritmos
pyplot.boxplot(results, labels=names)
pyplot.title('Comparación de algoritmos')
pyplot.show()

#>LR: 0.941667 (0.065085)
#>CART: 0.950000 (0.040825)
```



Ejemplos con Python: test de árboles de decisión

```
# Load libraries
from pandas import read_csv
from matplotlib import pyplot
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import StratifiedKFold
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
# Cargamos el dataset
url = "https://www.openml.org/data/get_csv/61/dataset_61_iris.arff"
dataset = read_csv(url)
# Dividimos el dataset en 80% de datos para entrenar y 20% para test
array = dataset.values
X = array[:,0:4]
y = array[:,4]
X_train, X_validation, Y_train, Y_validation = train_test_split(X, y
, test_size=0.20, random_state=1, shuffle=True)
```

Ejemplos con Python: test de árboles de decisión

```
# Dividimos el dataset en 80% de datos para entrenar y 20% para test
array = dataset.values
X = array[:,0:4]
y = array[:,4]
X_train, X_validation, Y_train, Y_validation = train_test_split(X, y
, test_size=0.20, random_state=1, shuffle=True)

# Realizamos predicciones con el dataset de validación
model = DecisionTreeClassifier()
model.fit(X_train, Y_train)
predictions = model.predict(X_validation)

# Evaluamos las predicciones, en primer lugar la precisión obtenida
print(accuracy_score(Y_validation, predictions))
#> 0.9666666666666667
```

Ejemplos con Python: test de árboles de decisión

```
# ahora la matriz de confusión (vemos en este ejemplo que sólo hemos
 cometido un fallo)
print(confusion_matrix(Y_validation, predictions))
#>[[11  0  0]
#> [ 0 12  1]
#> [ 0  0  6]]

# y finalmente un informe de clasificación
print(classification_report(Y_validation, predictions))
#>                 precision    recall  f1-score   support
#>
#>    Iris-setosa       1.00      1.00      1.00        11
#>Iris-versicolor       1.00      0.92      0.96        13
#> Iris-virginica       0.86      1.00      0.92         6
#>
#>       accuracy                           0.97        30
#>      macro avg       0.95      0.97      0.96        30
#>   weighted avg       0.97      0.97      0.97        30
```

Ejemplos con Python: test de árboles de decisión

```
# realizamos una predicción de ejemplo:
print(model.predict([[6.0, 3.0, 5.0, 2.0]]))
#>['Iris-virginica']
```

Ejemplos con Python: visualizando el árbol

```
# Load libraries
from pandas import read_csv
import matplotlib.pyplot as plt
import matplotlib.image as pltimg
import pydotplus
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import StratifiedKFold
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree

# Cargamos el dataset
url = "https://www.openml.org/data/get_csv/61/dataset_61_iris.arff"
dataset = read_csv(url)
```

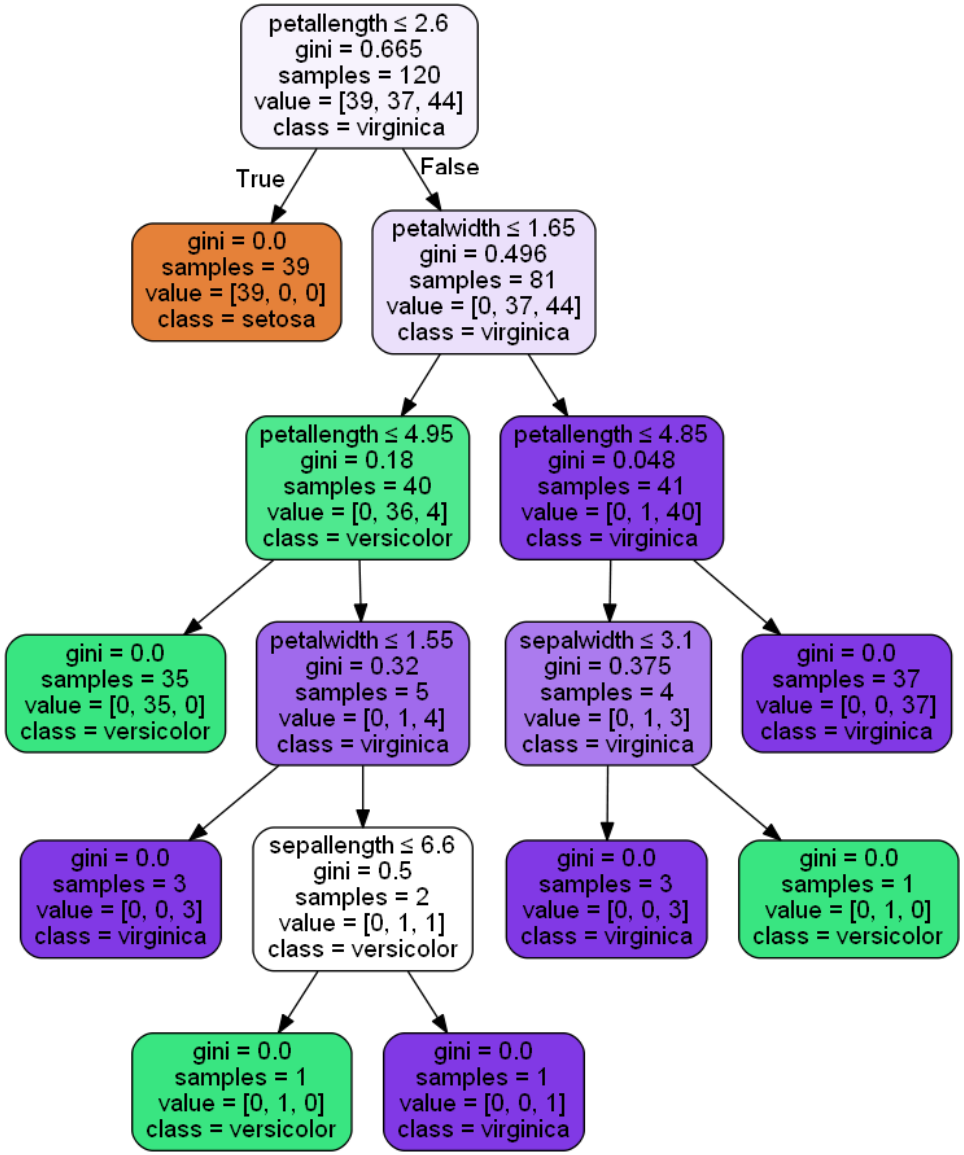
Ejemplos con Python: visualizando el árbol

```
# Dividimos el dataset en 80% de datos para entrenar y 20% para test
array = dataset.values
X = array[:,0:4]
y = array[:,4]
X_train, X_validation, Y_train, Y_validation = train_test_split(X, y
, test_size=0.20, random_state=1, shuffle=True)

# Realizamos predicciones con el dataset de validación
model = DecisionTreeClassifier()
model.fit(X_train, Y_train)
predictions = model.predict(X_validation)

# damos detalles sobre el modelo
print(model)
```

Ejemplos con Python: visualizando el árbol

```
# mostramos el árbol gráifcamente
data = tree.export_graphviz(model, out_file=None, feature_names=data
set.columns.values[0:4], class_names=["setosa", "versicolor", "virgi
nica"], filled=True, rounded=True, special_characters=True)

graph = pydotplus.graph_from_dot_data(data)
graph.write_png('mydecisiontree.png')

img = pltimg.imread('mydecisiontree.png')
imgplot = plt.imshow(img)
plt.show()
```

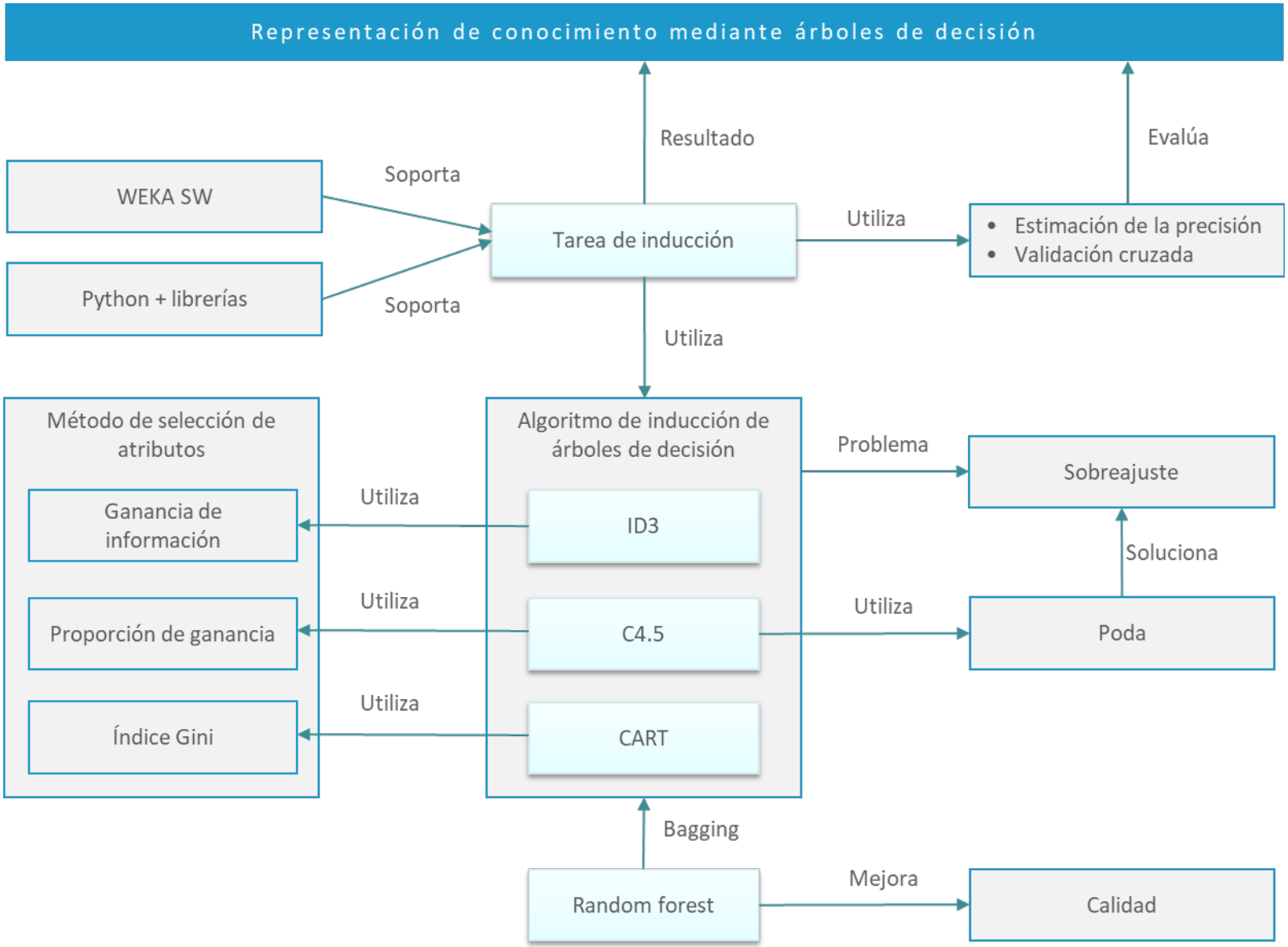
Ejemplos con Python: visualizando el árbol



Ejemplos con Python: visualizando el árbol

```
DecisionTreeClassifier(ccp_alpha=0.0,
class_weight=None,
criterion='gini',
max_depth=None,
max_features=None,
max_leaf_nodes=None,
min_impurity_decrease=0.0,
min_impurity_split=None,
min_samples_leaf=1,
min_samples_split=2,
min_weight_fraction_leaf=0.0,
presort='deprecated',
random_state=None,
splitter='best')
```

Resumen


Representación de conocimiento mediante árboles de decisión
Resultado
Evalúa
WEKA SW
Soporta
Tarea de inducción
Utiliza
Estimación de la precisión
Validación cruzada
Python + librerías
Soporta
Utiliza
Método de selección de atributos
Algoritmo de inducción de árboles de decisión
Problema
Sobreajuste
Ganancia de información
Utiliza
ID3
Soluciona
Proporción de ganancia
Utiliza
C4.5
Utiliza
Poda
Índice Gini
Utiliza
CART
Bagging
Random forest
Mejora
Calidad

Gracias por vuestra atención
¿Dudas?

Imagen por Peggy und Marco Lachmann-Anke
Licencia: Creative Commons Zero

www.unir.net

Técnicas de Inteligencia Artificial

Ricardo S. Alonso Rincón

Anexo. Métricas para medir la precisión de un modelo clasificador

UNIVERSIDAD INTERNACIONAL DE LA RIOJA

La Universidad en Internet

Medidas de la precisión de la clasificación

Matriz de confusión

Dadas n clases, la **matriz de confusión** tiene un tamaño $n \times n$ y sus elementos p_{ij} indican el número de instancias de la clase *i* que han sido clasificadas por el modelo como de la clase *j*.

- Filas → clase real de la instancia
- Columnas → clase estimada por el clasificador.

```
=== Confusion Matrix ===

 a b   <-- classified as
 8 1 | a = yes
 1 4 | b = no
```

Fuente: Elena Verdú, UNIR

Medidas de la precisión de la clasificación

Verdaderos y Falsos Positivos y Negativos

		Clase Predicha	
		Sí	No
Clase Real	Sí	Verdadero Positivo (TP)	Falso Negativo (FN)
	No	Falso Positivo (FP)	Verdadero Negativo (TN)

¡El coste de un falso positivo podría ser distinto al de un falso negativo!

Fuente: Elena Verdú, UNIR

Medidas de la precisión de la clasificación

Tasa de verdaderos positivos *(TP Rate)* → proporción de instancias positivas correctamente clasificadas. También llamado Recall, alcance y sensibilidad.

$$TP\ rate = \frac{TP}{P}$$

TP: número de instancias positivas correctamente clasificadas
P: total de instancias positivas

=== Confusion Matrix ===

```
 a b   <-- classified as
 8 1 | a = yes
 1 4 | b = no
```

P (positivos): instancias de la clase yes

$$TP\ rate = \frac{TP}{P} = \frac{8}{8+1} = 0{,}889$$

TP (Verdaderos positivos): Instancias positivas (de la clase yes) clasificadas correctamente como positivas (clasificadas como yes)

Fuente: Elena Verdú, UNIR

Medidas de la precisión de la clasificación

Tasa de verdaderos negativos *(TN Rate)* → proporción de instancias negativas correctamente clasificadas. También llamado Especificidad (specifity).

$$TN\ rate = \frac{TN}{N}$$

TN: número de instancias negativas correctamente clasificadas
N: total de instancias negativas

=== Confusion Matrix ===

a	b	<-- classified as
8	1	a = yes
1	**4**	b = no

N (negativos): instancias de la clase no

$$TN\ rate = \frac{TN}{N} = \frac{4}{4+1}$$
$$= 0{,}8$$

TN (Verdaderos negativos): instancias negativas (de la clase no) clasificadas correctamente como negativas (clasificadas como no)

Fuente: Elena Verdú, UNIR

Medidas de la precisión de la clasificación

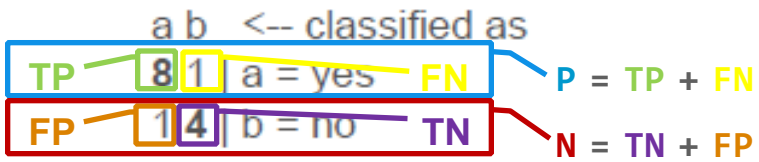
La *exactitud (accuracy)* es porcentaje de instancias del conjunto de datos de prueba que son clasificadas correctamente

La *tasa de error (error rate)* es porcentaje de instancias del conjunto de datos de prueba que son clasificadas incorrectamente

$$accuracy = \frac{TP + TN}{P + N}$$

$$Tasa\ de\ error = 1 - accuracy = \frac{FP + FN}{P + N}$$

=== Confusion Matrix ===



TP: Verdadero positivo → clasificado de forma verdadera (correcta) como positivo, por tanto era un positivo y se clasifica como positivo.
FN: Falso negativo → clasificado de forma falsa (incorrecta) como negativo, por tanto era un positivo y se clasifica como negativo.
TN: Verdadero negativo → clasificado de forma verdadera (correcta) como negativo, por tanto era un negativo y se clasifica como negativo.
FP: Falso positivo → clasificado de forma falsa (incorrecta) como positivo, por tanto era un negativo y se clasifica como positivo.

$$accuracy = \frac{TP + TN}{P + N} = \frac{8 + 4}{9 + 5} = 0{,}857$$

$$tasa\ de\ error = \frac{FP + FN}{P + N} = \frac{1 + 1}{9 + 5} = 0{,}142$$

Fuente: Elena Verdú, UNIR

Medidas de la precisión de la clasificación

La *precisión* es el porcentaje de instancias positivas respecto al total predichas como positivas

$$precisión = \frac{TP}{TP + FP}$$

```
=== Confusion Matrix ===

   a b   <-- classified as
 8 1 | a = yes
 1 4 | b = no
```

TP

FP

$$precisión = \frac{8}{8 + 1} = 0{,}89$$

TP

TP + FP

TP (Verdaderos positivos):
Instancias positivas (de la clase yes) clasificadas correctamente (clasificadas como yes)

FP (Falsos positivos):
Instancias negativas (de la clase no) clasificadas incorrectamente (clasificadas como yes)

Fuente: Elena Verdú, UNIR

Medidas de la precisión de la clasificación

La *precisión* es el porcentaje de instancias positivas respecto al total predichas como positivas

$$precisión = \frac{TP}{TP + FP}$$

TP: número de instancias positivas correctamente clasificadas
FP: número de instancias incorrectamente clasificadas como positivas

Tasa de verdaderos positivos *(TP Rate), o* Recall, alcance,sensibilidad.
→ proporción de instancias positivas correctamente clasificadas.

$$TP\ rate = \frac{TP}{P}$$

TP: número de instancias positivas correctamente clasificadas
P: total de instancias positivas

$$F - measure = \frac{2 \times precision \times recall}{precision + recall}$$

Fuente: Elena Verdú, UNIR

Weka: TP Rate

(también llamado recall, alcance o sensibilidad)

```
=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0,778     0,600     0,700       0,778   0,737      0,189    0,789     0,847     yes
                 0,400     0,222     0,500       0,400   0,444      0,189    0,789     0,738     no
Weighted Avg.    0,643     0,465     0,629       0,643   0,632      0,189    0,789     0,808
```

- TP Rate para la clase yes

```
=== Confusion Matrix ===

 a b   <-- classified as
 7 2 | a = yes
 3 2 | b = no
```

P

TP

$$\text{TP rate} = \frac{TP}{P} = \frac{7}{7+2} = 0{,}778$$

Weka: TP Rate

(también llamado recall, alcance o sensibilidad)

```
=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 0,778     0,600     0,700       0,778   0,737       0,189    0,789     0,847     yes
                 0,400     0,222     0,500       0,400   0,444       0,189    0,789     0,738     no
Weighted Avg.    0,643     0,465     0,629       0,643   0,632       0,189    0,789     0,808
```

- TP Rate para la clase no

```
=== Confusion Matrix ===

 a b   <-- classified as
 7 2 | a = yes
 3 2 | b = no
```

P

TP

$$\text{TP rate} = \frac{TP}{P} = \frac{2}{2+3} = 0{,}400$$

Weka: TP Rate

(también llamado recall, alcance o sensibilidad)

```
=== Detailed Accuracy By Class ===
```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,778	0,600	0,700	0,778	0,737	0,189	0,789	0,847	yes
	0,400	0,222	0,500	0,400	0,444	0,189	0,789	0,738	no
Weighted Avg.	0,643	0,465	0,629	0,643	0,632	0,189	0,789	0,808	

- Media ponderada del TP Rate

$$\text{TP rate} = \frac{P}{P+N} \cdot \text{TP rate } clase\ P + \frac{N}{P+N} \cdot \text{TP rate } clase\ N$$

$$\text{TP rate } 'yes' = 0{,}778$$

$$\text{TP rate } 'no' = 0{,}400$$

$$\text{TP rate} = \frac{P}{P+N} \cdot \text{TP rate}'yes' + \frac{N}{P+N} \cdot \text{TP rate}'no' =$$

$$\frac{9}{9+5} \cdot 0{,}778 + \frac{5}{9+5} \cdot 0{,}400 = 0{,}643$$

Weka: TN Rate (también llamado especificidad)

```
=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 0,778     0,600     0,700       0,778   0,737       0,189    0,789     0,847     yes
                 0,400     0,222     0,500       0,400   0,444       0,189    0,789     0,738     no
Weighted Avg.    0,643     0,465     0,629       0,643   0,632       0,189    0,789     0,808
```

- TN Rate para la clase yes

```
=== Confusion Matrix ===

 a b   <-- classified as
 7 2 | a = yes
 3 2 | b = no
```

TN

N

$$\text{TN rate} = \frac{TN}{N} = \frac{2}{2+3} = 0{,}400$$

TN Rate para la clase yes = TP Rate para la clase no

Weka: TN Rate (también llamado especificidad)

```
=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0,778     0,600     0,700       0,778   0,737      0,189    0,789     0,847     yes
                 0,400     0,222     0,500       0,400   0,444      0,189    0,789     0,738     no
Weighted Avg.    0,643     0,465     0,629       0,643   0,632      0,189    0,789     0,808
```

- TN Rate para la clase no

```
=== Confusion Matrix ===

 a b   <-- classified as
 7 2 | a = yes
 3 2 | b = no
```

TN

N

$$\text{TN rate} = \frac{TN}{N} = \frac{7}{7+2} = 0{,}778$$

TN Rate para la clase no = TP Rate para la clase yes

Weka: FP Rate

```
=== Detailed Accuracy By Class ===
```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,778	0,600	0,700	0,778	0,737	0,189	0,789	0,847	yes
	0,400	0,222	0,500	0,400	0,444	0,189	0,789	0,738	no
Weighted Avg.	0,643	0,465	0,629	0,643	0,632	0,189	0,789	0,808	

- FP Rate para la clase yes

```
=== Confusion Matrix ===

 a b   <-- classified as
 7 2 | a = yes
 3 2 | b = no
```

FP

N

$$\text{FP rate} = \frac{FP}{N} = \frac{3}{3+2} = 0{,}600$$

Weka: FP Rate

```
=== Detailed Accuracy By Class ===
```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,778	0,600	0,700	0,778	0,737	0,189	0,789	0,847	yes
	0,400	0,222	0,500	0,400	0,444	0,189	0,789	0,738	no
Weighted Avg.	0,643	0,465	0,629	0,643	0,632	0,189	0,789	0,808	

- FP Rate para la clase no

```
=== Confusion Matrix ===

 a b   <-- classified as
 7 2 | a = yes
 3 2 | b = no
```

N

FP

$$\text{FP rate} = \frac{FP}{N} = \frac{2}{2+7} = 0{,}222$$

Weka: Precisión

```
=== Detailed Accuracy By Class ===
```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,778	0,600	0,700	0,778	0,737	0,189	0,789	0,847	yes
	0,400	0,222	0,500	0,400	0,444	0,189	0,789	0,738	no
Weighted Avg.	0,643	0,465	0,629	0,643	0,632	0,189	0,789	0,808	

- Precisión para la clase yes

```
=== Confusion Matrix ===

   a b   <-- classified as
   7 2 | a = yes
   3 2 | b = no
```

TP: 7

FP: 3

$$\text{Precision} = \frac{TP}{TP+FP} = \frac{7}{7+3} = 0{,}700$$

Weka: Precisión

```
=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0,778     0,600     0,700       0,778   0,737      0,189    0,789     0,847     yes
                 0,400     0,222     0,500       0,400   0,444      0,189    0,789     0,738     no
Weighted Avg.    0,643     0,465     0,629       0,643   0,632      0,189    0,789     0,808
```

- Precisión para la clase no

```
=== Confusion Matrix ===

   a b   <-- classified as
   7 2 | a = yes
   3 2 | b = no
```

FP

TP

$$\text{Precision} = \frac{TP}{TP+FP} = \frac{2}{2+2} = 0{,}500$$

Weka: F-Measure

```
=== Detailed Accuracy By Class ===
```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,778	0,600	0,700	0,778	0,737	0,189	0,789	0,847	yes
	0,400	0,222	0,500	0,400	0,444	0,189	0,789	0,738	no
Weighted Avg.	0,643	0,465	0,629	0,643	0,632	0,189	0,789	0,808	

- F-measure para la clase yes

$$F\ measure = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

$$F\ measure = \frac{2 \cdot 0{,}700 \cdot 0{,}778}{0{,}700 + 0{,}778} = 0{,}737$$

Weka: F-Measure

```
=== Detailed Accuracy By Class ===
```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,778	0,600	0,700	0,778	0,737	0,189	0,789	0,847	yes
	0,400	0,222	0,500	0,400	0,444	0,189	0,789	0,738	no
Weighted Avg.	0,643	0,465	0,629	0,643	0,632	0,189	0,789	0,808	

- F-measure para la clase no

$$F\ measure = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

$$F\ measure = \frac{2 \cdot 0{,}500 \cdot 0{,}400}{0{,}500 + 0{,}400} = 0{,}444$$

Weka: Exactitud (accuracy)

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances           9               64.2857 %
Incorrectly Classified Instances         5               35.7143 %
Kappa statistic                          0.186
Mean absolute error                      0.2857
Root mean squared error                  0.4818
Relative absolute error                 60      %
Root relative squared error             97.6586 %
Total Number of Instances               14
```

- Exactitud

```
=== Confusion Matrix ===

   a b   <-- classified as
   7 2 | a = yes
   3 2 | b = no
```

TP = 7, FN = 2, FP = 3, TN = 2

P = TP + FN

N = TN + FP

$$\text{accuracy} = \frac{TP+TN}{P+N} = \frac{7+2}{(7+2)+(2+3)} = 0{,}643$$

Weka: Tasa de error

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances           9               64.2857 %
Incorrectly Classified Instances         5               35.7143 %
Kappa statistic                          0.186
Mean absolute error                      0.2857
Root mean squared error                  0.4818
Relative absolute error                 60      %
Root relative squared error             97.6586 %
Total Number of Instances               14
```

- Tasa de error

```
=== Confusion Matrix ===

   a b   <-- classified as
   7 2 | a = yes
   3 2 | b = no
```

TP = 7, FN = 2, FP = 3, TN = 2

P = TP + FN

N = TN + FP

$$\text{Tasa error} = \frac{FP+FN}{P+N} = \frac{3+2}{(7+2)+(2+3)} = 0{,}357$$

$$\text{Tasa error} = 1 - \text{accuracy}$$