

Hoy

Árboles de decisión (CART: Classification and regression trees)

Árboles de clasificación

El algoritmo ID3

Expresividad

Árboles de regresión



juego de las 20 preguntas

20q.net

Vas a decidir si entras a un restaurante

Input :

Output : Sí o No



juego de las 20 preguntas

20q.net

Vas a decidir si entras a un restaurante

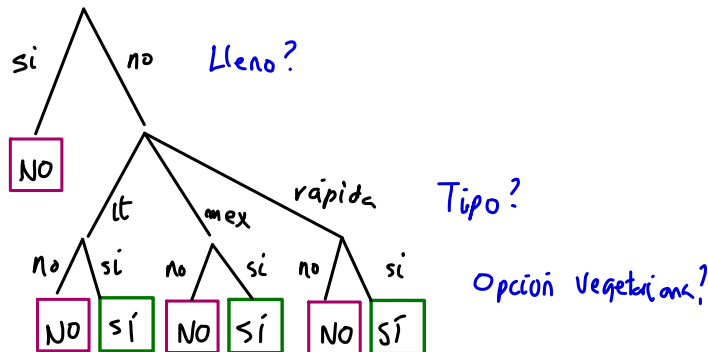
Input :

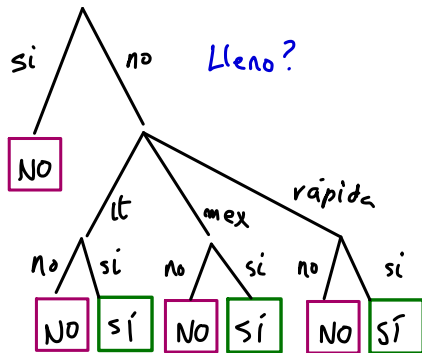
Output : Sí o No

Lleno?

Tipo?

Vegetariano?





Tipo?

Opción Vegetación?

Lleno	Tipo	Veg	Entro
T	it	F	NO
T	max	T	NO
F	max	F	NO
F	rápida	T	SÍ

Juego de las 20 preguntas: ¿Por cuál pregunta empieza?

No es igual de eficiente empezar por cualquier pregunta

Aprendizaje de árboles de decisión → Variable/característica

① Elige Mejor atributo (mejor respuesta)

② "Pregunta"

③ Sigue la rama de la respuesta


④ Ir al paso 1 hasta encontrar respuesta

→ ojo hay diferentes formas de terminar el proceso

¿Cómo elegir el mejor atributo?

Information Gain

$$\text{Gain}(S, A) = \text{Ent}(S) - \sum_{v \in \text{values}(A)} \frac{|S_v|}{|S|} \text{Ent}(S_v)$$

Atributo

$$\text{Ent}(S) = \sum_{i=1}^c p_i \log_2 p_i$$

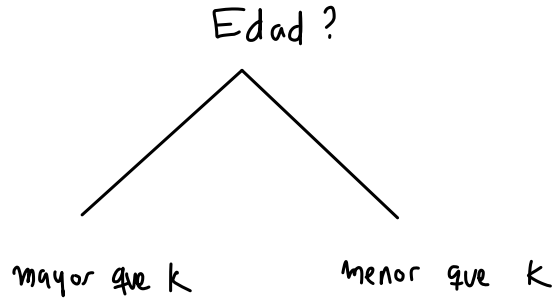

En el taller ahondaremos

Una vez tiene el atributo decidido

¿Cómo el árbol "pregunta"?

El atributo realmente tendrá valores numéricos y cada rama corresponderá a intervalos numéricos.

Ejemplo :



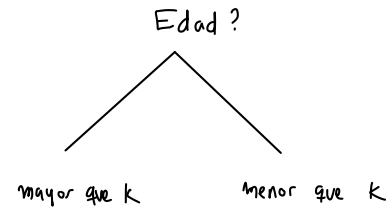
¿Ok... pero quién es k ?



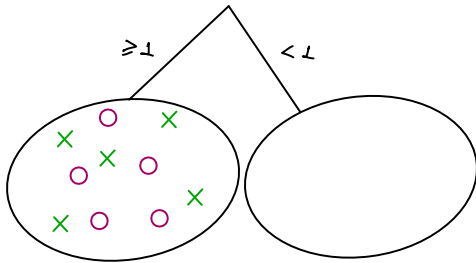
Lo que hace k es partir el conjunto de registros en 2

Puede haber particiones más convenientes que otras.

Supongamos que nuestra variable objetivo es binaria SÍ ○ NO ×

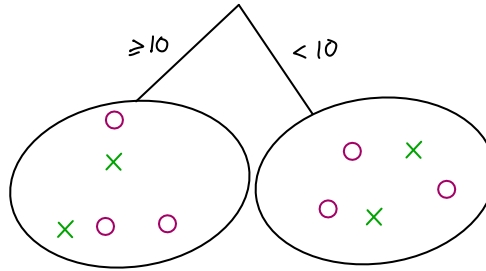


$K=1$

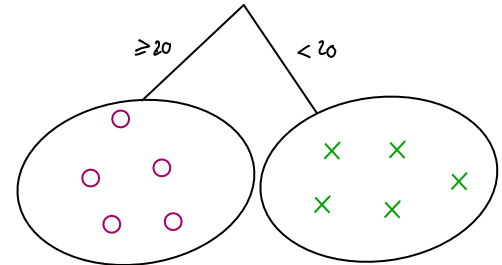


Mayor Entropía

$K=1$



$K=20$



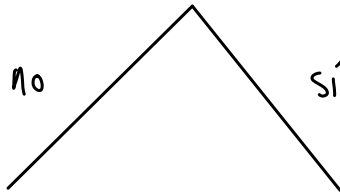
Menor entropía

Para decidir k (cómo partir) se utiliza una medida de "impureza"

Esta medida puede ser Entropy, Gini, entre otras

Esse valor k se convierte en la "pregunta" del árbol.

¿Es mayor que k ?



El algoritmo

ID3 (Iterative Dichotomy 3)

- ▶ Elegir el mejor atributo A
- ▶ Asignar al atributo A un nodo
- ▶ Para cada valor de A
 Crear un nodo descendente
- ▶ Partir los datos de acuerdo a los valores
- ▶ Recursivamente hacer nuevos árboles para cada nodo descendente
- ▶ Si todos los elementos de un subconjunto tienen la misma clasificación, asignar esa clasificación a la hoja
- ▶ Si un subconjunto es vacío, asignar la clasificación más popular

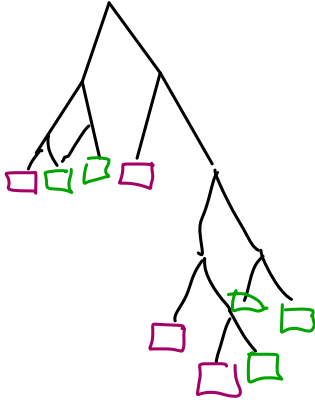
¿Cuándo para el proceso?

Todo se clasificó
correctamente
(todos los nodos
finales son hojas
con etiquetas)

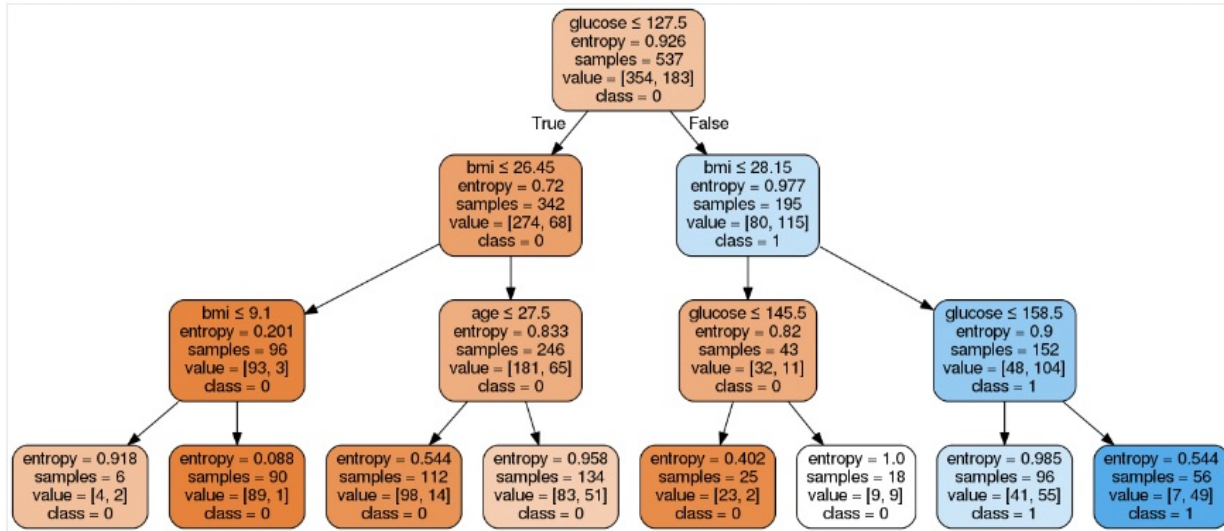
No hay más
atributos

Se llega a un
límite preestablecido
para el modelo

Ej. Número máximo de hojas
Profundidad máxima



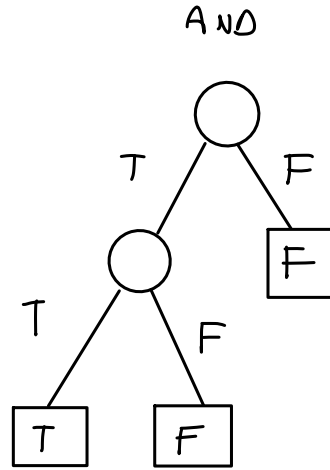
Así luce el árbol con SKLearn



Expresividad de los árboles de decisión

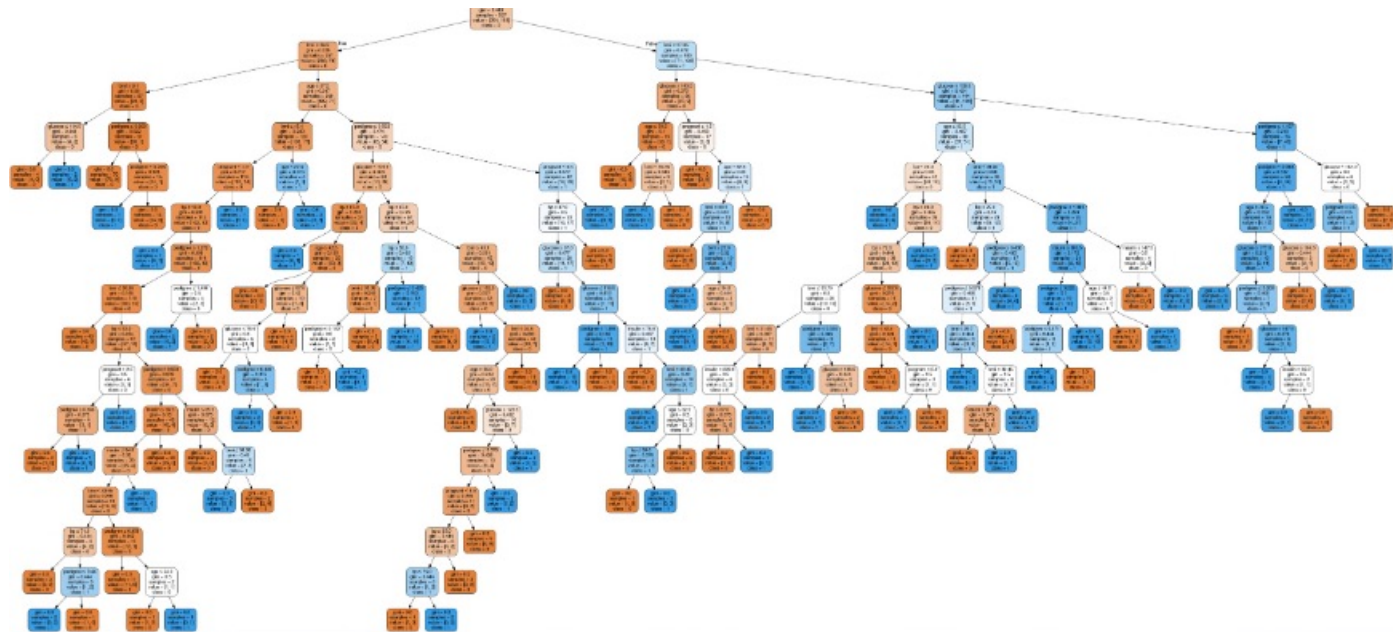
Toda función en lógica proposicional se puede expresar como un árbol de decisión

AND	F	T
F	F	F
T	F	T

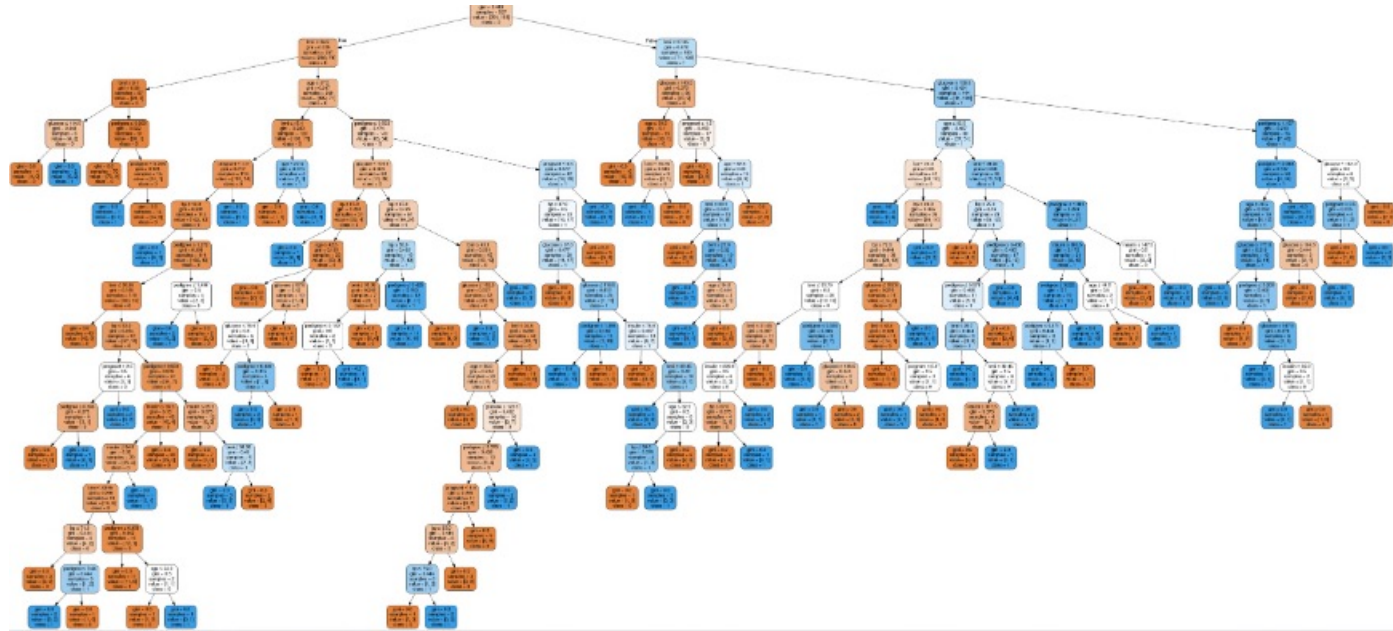


XOR

Un árbol muy grande



Un árbol muy grande



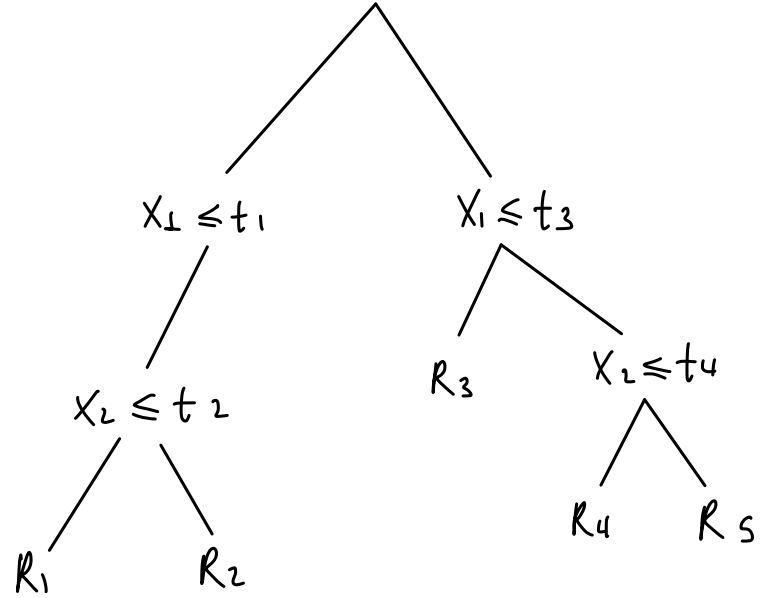
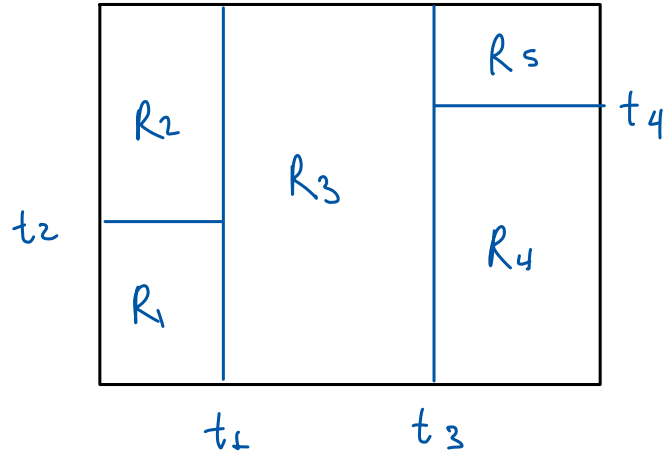
Puede estar sobre ajustandose a los datos

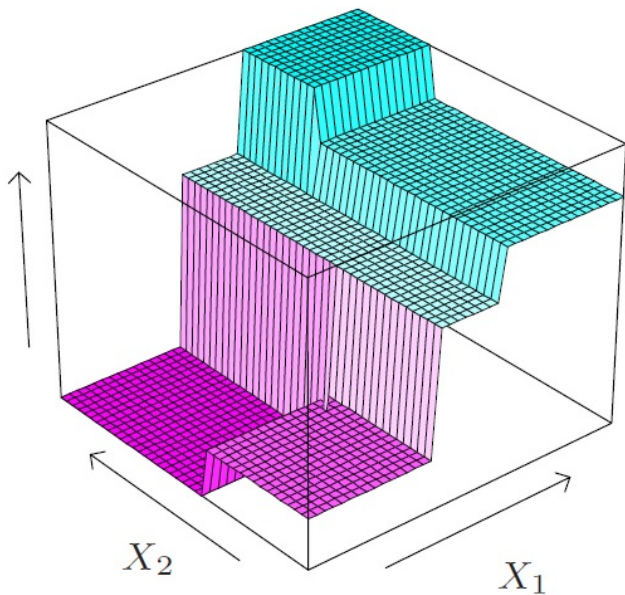
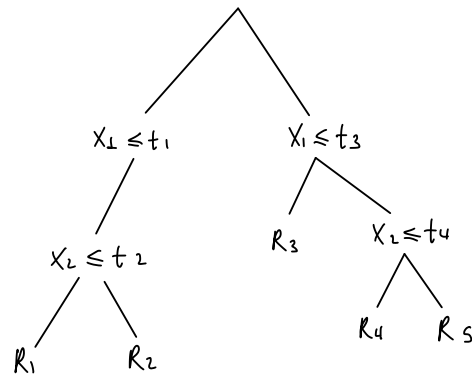
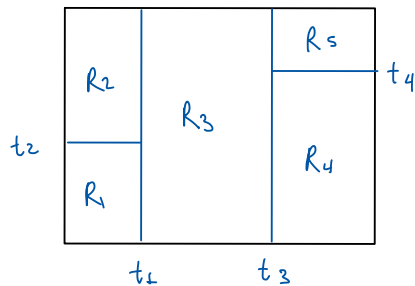
Por eso suele ser conveniente que el árbol no sea tan grande.

Esto se hace de 2 maneras

1. Limitando la construcción del árbol mediante parámetros del algoritmo como (# max de hojas, profundidad máxima etc.)
2. Podando el árbol (Ahondaremos en el taller)

Árboles de Regresión



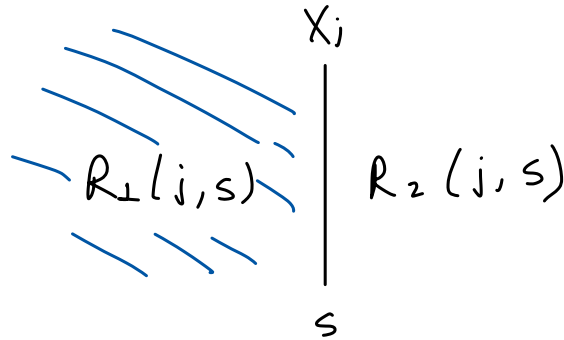


La altura en cada región R_m
se asigna como :

$$f(x) = \sum_{m=1}^n c_m I(x \in R_m)$$

\searrow promedio $(y_i | x_i \in R_m)$

¿Cómo encontrar la mejor partición?



$$\min_{j, s} \left[\min_{c_1} \sum_{x_i \in R_1(j, s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j, s)} (y_i - c_2)^2 \right]$$