Árboles de decisión (CART)

Matías Carrasco

12 de octubre de 2023

Índice

1. Introducción

Los árboles de decisión son modelos basados en reglas. La razón es que las reglas utilizadas para definir el modelo pueden organizarse en una estructura de grafo conocida como árbol binario. El árbol de decisión divide el espacio de atributos en múltiples regiones disjuntas y en cada región se utiliza un valor constante para la predicción $\widehat{y}$.

Consideremos nuevamente el ejemplo del rendimiento de los cultivos de papas:

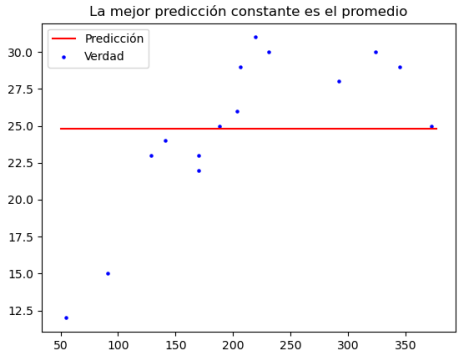
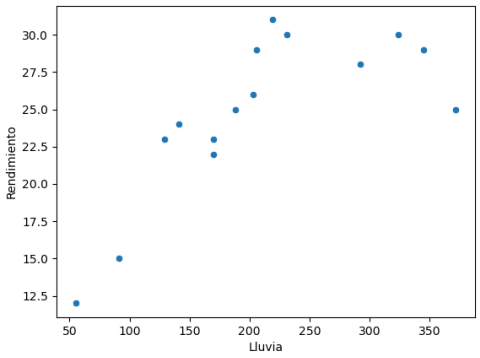


Figura 1: La mejor predicción constante es el promedio

x = Lluvia (mm)	y = Rendimiento (ton/ha)
206	29
188	25
219	31
372	25
345	29
231	30
203	26
170	23
55	12
91	15
292	28
141	24
129	23
170	22
324	30



Si tuviéramos que predecir y con una constante c y la utilizamos como función de pérdida la MSE

$$\text{MSE}(c) = \frac{1}{N}\sum_{i=1}^{N}(c - y_i)^2$$

es sencillo ver (derivar e igualar a cero) que

$$\text{promedio} = \bar{y} = \arg\min_{c\in\mathbb{R}} \text{MSE}(c).$$

Es decir, la mejor predicción constante es el promedio, ver Fig. 1.

Consideremos ahora funciones $t : \mathcal{X} \to \mathcal{Y}$ de la forma

$$t(x) = \begin{cases} c_1 & \text{si } x < x^* \\ c_2 & \text{si } x \geq x^* \end{cases}$$

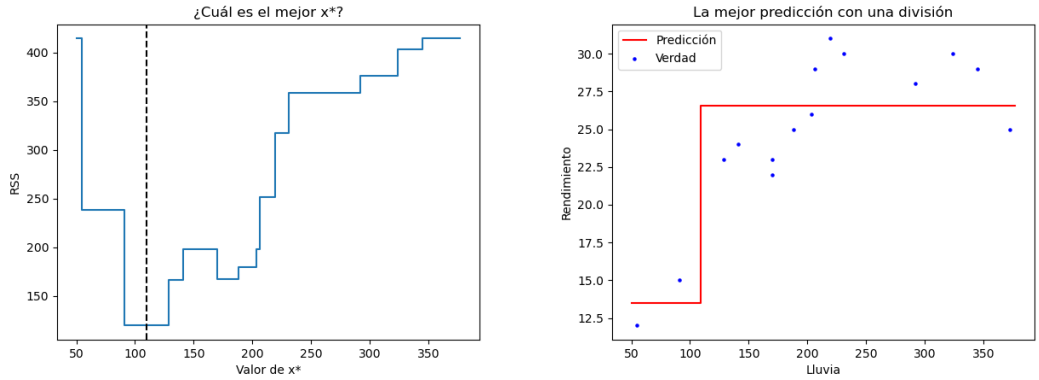


Figura 2: Dividiendo el dataset con una pregunta.

Queremos (c_1, c_2, x^*) que minimizan la MSE, lo que equivale a minimizar

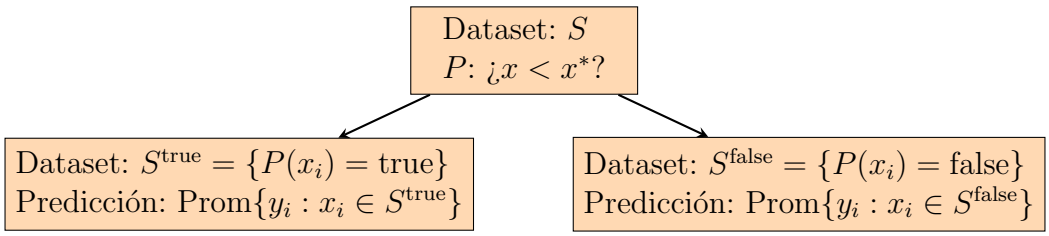
$$\text{MSE}(t) = \frac{1}{N}\left[\sum_{x_i<x^*}(y_i - c_1)^2 + \sum_{x_i\geq x^*}(y_i - c_2)^2\right]$$

Fijado x^*, la MSE se minimiza para

$$\begin{cases} c_1 = \bar{y}_1 = \text{promedio}\{y_i : x_i < x^*\} \\ c_2 = \bar{y}_2 = \text{promedio}\{y_i : x_i \geq x^*\} \end{cases}$$

¿Cuál es el mejor x^*? Ver Fig. 2.

Dividir el dataset según $x < x^*$ o no puede pensarse como dividirlo de acuerdo a la respuesta a una pregunta:



Una pregunta P: ¿$x < x^*$? divide al dataset S en dos datasets

$$S^{\text{true}} = \{(x_i, y_i) \in S : P(x_i) = \text{true}\} \quad S^{\text{false}} = \{(x_i, y_i) \in S : P(x_i) = \text{false}\}$$

Observar que la MSE de la predicción promedio $\bar{y}$ en un dataset S es igual a la varianza

$$\text{MSE}(\bar{y}, S) = \frac{1}{|S|}\sum_{S}(y - \bar{y})^2 = \text{Var}(y; S)$$

La mejor pregunta P maximiza la **reducción de varianza**:

$$\mathrm{Var}(y; S) - \left(\frac{|S^{\text{true}}|}{|S|} \mathrm{Var}(y; S^{\text{true}}) + \frac{|S^{\text{false}}|}{|S|} \mathrm{Var}(y; S^{\text{false}}) \right)$$

En efecto, la MSE de t se puede descomponer como

$$\begin{aligned}
\mathrm{MSE}(t) &= \frac{1}{N} \sum_{x_i < x^*} (y_i - c_1)^2 + \frac{1}{N} \sum_{x_i \geq x^*} (y_i - c_2)^2 \\
&= \frac{|S^{\text{true}}|}{N} \left(\frac{1}{|S^{\text{true}}|} \sum_{x_i < x^*} (y_i - c_1)^2 \right) + \frac{|S^{\text{false}}|}{N} \left(\frac{1}{|S^{\text{false}}|} \sum_{x_i \geq x^*} (y_i - c_2)^2 \right) \\
&= \frac{|S^{\text{true}}|}{|S|} \mathrm{Var}(y; S^{\text{true}}) + \frac{|S^{\text{false}}|}{|S|} \mathrm{Var}(y; S^{\text{false}})
\end{aligned}$$

Como S está fijo, minimizar la MSE de t equivale a maximizar la reducción de varianza.

Esta idea de ir haciendo preguntas, cuyas respuestas pueden ser true o false, se puede repetir de forma iterativa, lo que resulta en un árbol binario. Esto se llama **Recursive Binary Splitting**, ver Fig. 3.

2. Aprendiendo un árbol de regresión

2.1. Definición general de árbol de decisión

La predicción $\widehat{y}$ de un árbol de regresión es una función constante por tramos del input x. Es decir

$$\widehat{y} = t(\boldsymbol{x}) = \sum_{\ell=1}^{L} \widehat{y}_\ell \mathbb{1}_{\{\boldsymbol{x} \in R_\ell\}},$$

donde L es el número total de regiones (nodos hoja) en el árbol, R_ℓ es la región ℓ-ésima, y $\widehat{y}_\ell$ es la predicción constante para la región ℓ-ésima. Ver Fig. 4.

Es decir, el árbol está constituido de

- Nodos: que se identifican con preguntas.
- Arcos: que se identifican con las respuestas a las preguntas (true/false).
- Hojas: que se identifican con las regiones que particionan el espacio.

Tanto los nodos como las hojas tienen asociados datasets, subconjuntos de S, que corresponden a los datos de entrenamiento que caen en dicho nodo al recorrer el

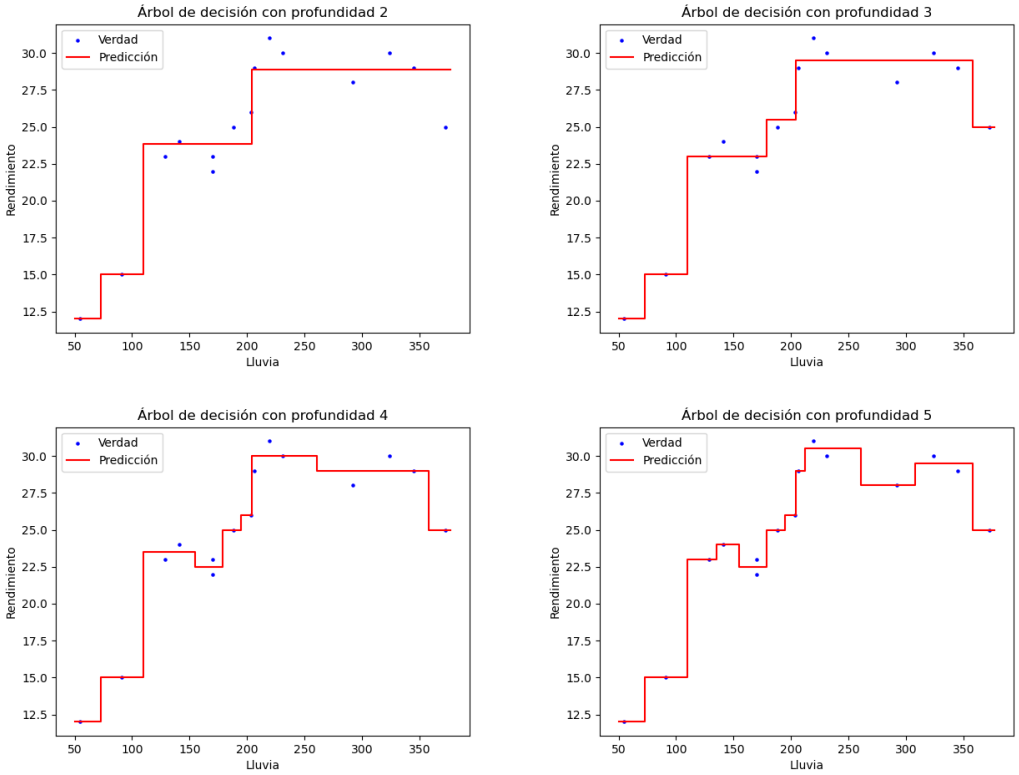


Figura 3: Recursive Binary Splitting

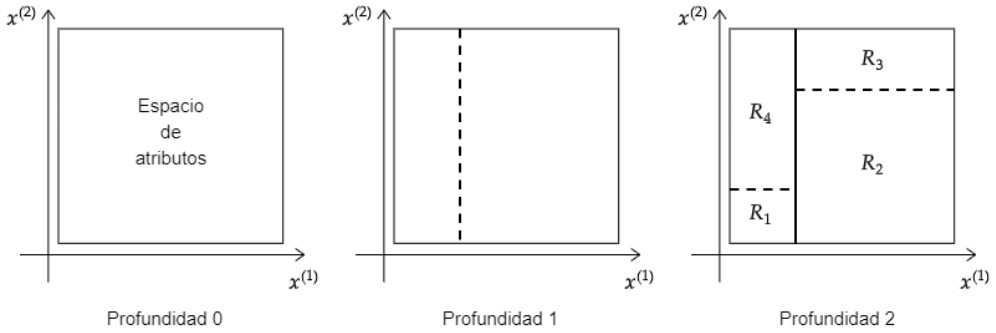


Figura 4: Partición en regiones asociada a un árbol de decisión.

árbol respondiendo a las preguntas. Además en las hojas es en donde hacemos las predicciones.

Aprender el árbol a partir de los datos corresponde a encontrar valores adecuados para los parámetros que definen la función $t(\boldsymbol{x})$, es decir, las regiones R_ℓ y las predicciones constantes $\widehat{y}_\ell, \ell = 1, \ldots, L$, así como el tamaño total del árbol L. Al igual que hicimos en la introducción, si fijamos la partición $\{R_\ell\}_{\ell=1}^{L}$, entonces las constantes $\{\widehat{y}_\ell\}_{\ell=1}^{L}$ simplemente son el promedio de los puntos de datos de entrenamiento que caen en cada región:

$$\widehat{y}_\ell = \text{Promedio}\,\{y_i : \boldsymbol{x}_i \in R_\ell\}$$

Encontrar el árbol (una colección de reglas de división) que particione óptimamente el espacio de atributos para ajustar los datos de entrenamiento lo mejor posible resulta ser computacionalmente inviable. El problema es que hay una explosión combinatoria en el número de formas en que podemos particionar el espacio de entrada. Buscar entre todos los árboles binarios posibles no es práctico.

2.2. Recursive Binary Splitting

Es por eso que usamos Recursive Binary Splitting para aprender el árbol. La palabra recursiva significa que determinaremos las reglas de división una tras otra, comenzando con la primera división en la raíz y luego construyendo el árbol de arriba hacia abajo. El algoritmo es **greedy**, en el sentido de que el árbol se construye una división a la vez, sin tener en mente el árbol completo. Es decir, al determinar la regla de división en el nodo raíz, el objetivo es obtener un modelo que explique los datos de entrenamiento lo mejor posible después de una única división, sin tener en cuenta que se pueden agregar divisiones adicionales antes de llegar al modelo final. Cuando hemos decidido la primera división del espacio de atributos (correspondiente al nodo raíz del árbol), esta división se mantiene fija y continuamos de manera similar para los dos espacios resultantes (correspondientes a las dos ramas del árbol), etc.

2.3. Criterios de parada

¿Cuándo se termina? Los criterios de parada más comunes son (i.e. las condiciones para no dividir un nodo del árbol):

- Ya se consideraron todos los pares (j, s) (i.e., no hay más preguntas)
- No hay preguntas que reduzcan la varianza.
- Se alcanza una profundidad máxima para el árbol.

Algorithm 1 Recursive Binary Splitting: aprender un árbol en regresión

Input: Datos de entrenamiento S
Output: Árbol con regiones $R_1, \ldots, R_L$ y predicciones $\widehat{y}_1, \ldots, \widehat{y}_L$
Sea R todo el espacio de atributos.
Calcular las regiones $R_1, \ldots, R_L = \textsc{Split}(R, S)$
Calcular las predicciones $\widehat{y}_\ell = \text{Promedio}\{y_i : \boldsymbol{x}_i \in R_\ell\}$ para $\ell = 1, \ldots, L$

function $\textsc{Split}(R, S)$
    if criterio de parada se cumple **then**
        return R
    else
        Examinar pregunta $x^{(j)} < s$ para todo par (j, s), $j = 1, \ldots, D$.
        Elegir el par (j, s) que maximiza la reducción de varianza.
        Dividir la región R en R^{true} y R^{false} con el par (j, s).
        Dividir los datos S en S^{true} y S^{false} en consecuencia.
        return $\textsc{Split}(R^{\text{true}}, T^{\text{true}})$, $\textsc{Split}(R^{\text{false}}, T^{\text{false}})$
    end if
end function

Algorithm 2 Predecir a partir de un árbol de decisión

Input: Árbol con regiones $R_1, \ldots, R_L$, dato de prueba $\boldsymbol{x}$
Output: predicción $\widehat{y}$
Encontrar la región R_ℓ a la que pertenece $\boldsymbol{x}$.
return la predicción $\widehat{y} = \widehat{y}_\ell$.

- Se alcanza un mínimo de observaciones en el nodo.
- La división del nodo alcanza un mínimo de observaciones en las hojas.

La profundidad y los mínimos de observaciones (en nodos y hojas) son los **hiperparámetros** del algoritmo.

3. Aprendiendo un árbol de clasificación

El árbol de decisión para clasificación es completamente análogo al de regresión, excepto que no usamos la MSE (o la reducción en varianza) para elegir la mejor pregunta. En su lugar, se utilizan comúnmente uno de dos criterios específicos para medir el grado de variabilidad de una variable categórica: **impureza de Gini** o **entropía**.

Supongamos que el problema de clasificación en cuestión consiste en predecir una clase en $\mathcal{Y} = \{1, \dots, m\}$. Ambos criterios producen una medida cuantitativa de la variabilidad de la distribución de etiquetas en un dataset S:

$$p_c = \frac{\#\{i : y_i = c\}}{|S|}$$

Lo que nos interesa es disponer de una cantidad numérica $H(y; S)$ que mida la heterogeneidad del target y en el dataset S. Sea j un atributo, s un valor de j, y P la pregunta $x_j < s$ (en el caso categórico $x_j = s$), denotemos

$$S^r = \{s \in S \mid P(s) = r\}$$

para r = true/false. La heterogeneidad esperada después de dividir S usando P es:

$$H_P(y; S) = \sum_r \text{Prob}(P = r \mid S) \cdot H(y; S^r) = \sum_r \frac{|S^r|}{|S|} \cdot H(y; S^r)$$

Una vez definida esta cantidad, el mecanismo de elección de la mejor pregunta es completamente análogo: se elige P correspondiente al par (j, s) que maximiza la **reducción de heterogeneidad**:

$$P^* = \arg\max_P \Big\{ \underbrace{H(y; S) - H_P(y; S)}_{\text{reducción de heterogeneidad}} \Big\} = \arg\min_P H_P(y; S)$$

3.1. La impureza de Gini

La impureza de Gini evalúa la variabilidad en una distribución categórica, basándose en cuán difícil es predecir su valor a partir de una realización aleatoria de esta. La premisa es que este método acierta en la predicción de la clase cuando la distribución está altamente concentrada en una categoría. Sin embargo, su precisión disminuye considerablemente cuando la distribución tiende a ser uniforme.

Supongamos que y toma los valores $\{1,\ldots,\mathrm{m}\}$ con probabilidad p_i. Consideremos la siguiente forma de predecir y:

- Muestreamos un solo valor de y.
- Utilizamos dicho valor para predecir y.

La idea es que cuanto más heterogénea es la distribución de y mayor será el error de predicción.

¿Cuál es el error esperado?

- Si predecimos $\widehat{y} = i$ constante, la probabilidad de error es $1 - p_i$.
- El error esperado es

$$G(y) = \sum_{i=1}^{m} p_i(1 - p_i) = 1 - \sum_{i=1}^{m} p_i^2$$

Notar que $G(y)$ es máximo cuando $p_i = 1/m$ para todo $i = 1, \ldots, m$.

La **impureza de Gini** de y en un dataset S es por definición

$$G(y; S) = 1 - \sum_{c=1}^{m} \left(\frac{\#\{i : y_i = c\}}{|S|} \right)^2$$

3.2. El criterio de entropía

Otra forma de medir la variabilidad de una distribución categórica es basarse en conceptos de teoría de la información. Supongamos nuevamente que y toma los valores $\{1,\ldots,\mathrm{m}\}$ con probabilidad p_i.

La información asociada al evento $\{y = i\}$ se mide por $-\log_2 p_i$ (es una historia larga). La idea intuitiva de la fórmula es:

- la información es grande para eventos poco probables, y pequeña para eventos muy probables.

- la información es aditiva para eventos independientes:

$$-\log_2(pq) = -\log_2 p - \log_2 q.$$

¿Cuál es la información esperada?

$$H(y) = -\sum_{i=1}^{m} p_i \log_2 p_i$$

La información esperada $H(y)$ se llama **entropía** de y. También es máxima cuando $p_i = 1/m$ para todo $i = 1, \ldots, m$.

La entropía de y en un dataset S es por definición

$$H(y; S) = -\sum_{c=1}^{m} \frac{\#\{i : y_i = c\}}{|S|} \log_2 \left(\frac{\#\{i : y_i = c\}}{|S|} \right)$$