Árboles para clasificación y regresión
Aprendizaje Estadístico, 2024/2025

UNIVERSIDAD DE
MURCIA

Semana del 7 de Noviembre, 2024

Aprendizaje Estadístico

Máster en Tecnología de Análisis de Datos Masivos - Big Data.

Juan A. Botía (juanbot@um.es)

Métodos basados en árboles

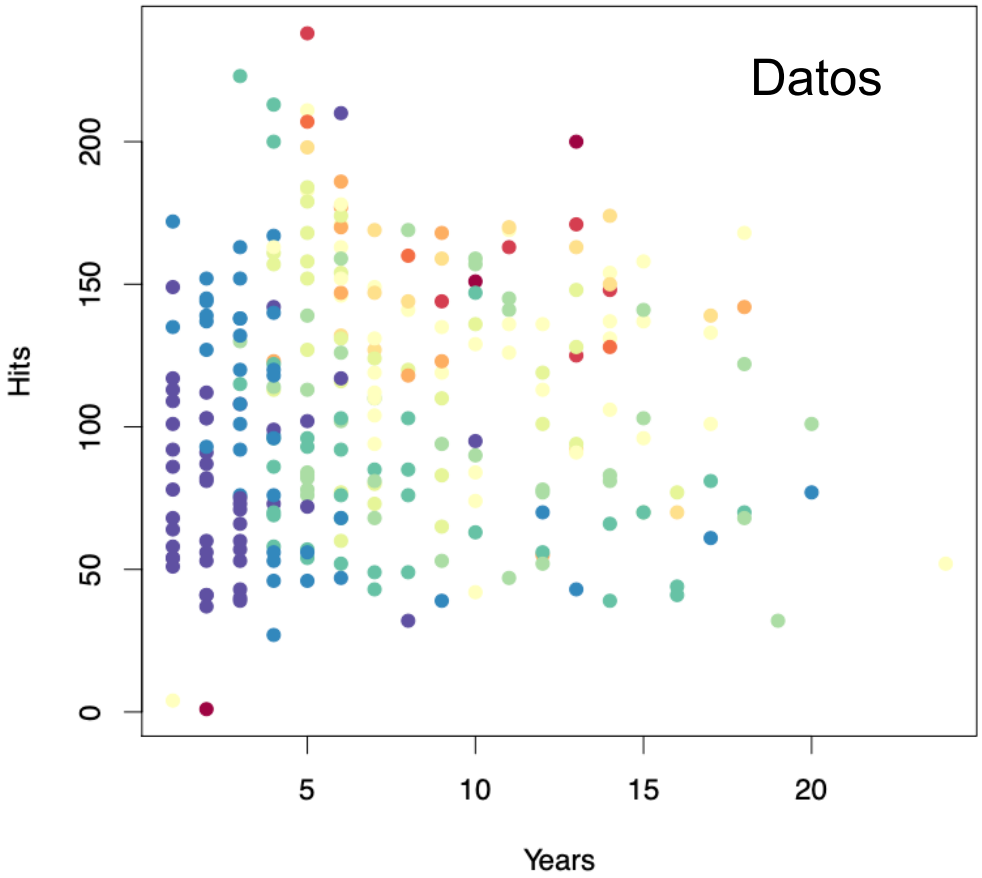
- Aquí describimos métodos basados en árboles para regresión y clasificación.
- Estos implican estratificar o segmentar el espacio del predictor en varias regiones simples.
- Dado que el conjunto de reglas de división que se utiliza para segmentar el espacio del predictor puede resumirse en un árbol, estos enfoques se conocen como métodos de árbol de decisión.
- Los árboles de decisión pueden aplicarse tanto a problemas de regresión como de clasificación.

Ventajas y desventajas

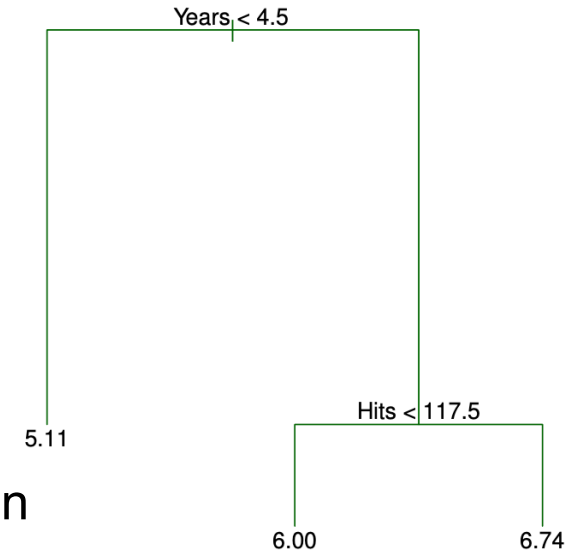
- Los métodos basados en árboles son sencillos y útiles para la interpretación.
- Generalmente no son tan competitivos como los mejores enfoques de aprendizaje supervisado en términos de precisión de predicción.
- Por lo tanto, también discutimos el "bagging", "random forests" y "boosting" (impulso).
  - Estos métodos desarrollan múltiples árboles que luego se combinan para ofrecer una única predicción de consenso.
- La combinación de muchos árboles puede resultar en mejoras significativas en la precisión de predicción, a cambio de perder algo de interpretabilidad.

Baseball salary data

- El salario está codificado por color de bajo (azul, verde) a alto (amarillo, rojo).



Estructura de árbol



Intrepretación

Years < 4.5

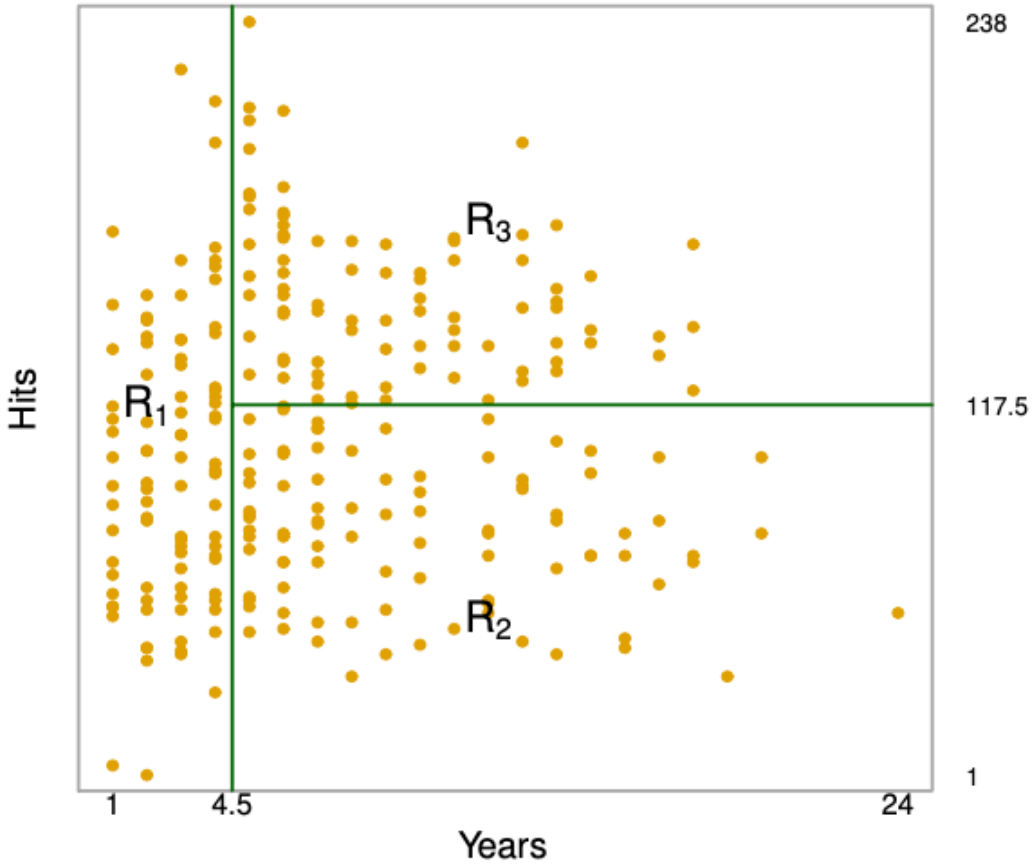
- (Yes) 5.11
- (No) Hits < 117.5
  - (Yes) 6.00
  - (No) 6.74

Detalles sobre la figura anterior

- Se centra en los datos de *Hitters*,
  - Construye un árbol de regresión para predecir el logaritmo del salario de un jugador de béisbol, basado en el número de años que ha jugado en las ligas mayores y el número de hits que hizo el año anterior.
- En un nodo interno dado, la etiqueta (de la forma $X_j < t_k$) indica que la rama izquierda corresponde a esa división, mientras que la rama derecha corresponde a $Xj \geq t_k$. Por ejemplo, la división en la parte superior del árbol da como resultado dos ramas grandes. La rama izquierda corresponde a Años < 4.5, y la derecha a Años ≥ 4.5.
- El árbol tiene dos nodos internos y tres nodos terminales o "hojas". El número en cada hoja es el promedio de la respuesta para las observaciones que caen en esa región.

Resultados

- En general, el árbol estratifica o segmenta a los jugadores en tres regiones del espacio predictor: R1 = {X | Años < 4.5}, R2 = {X | Años ≥ 4.5, Hits < 117.5}, y R3 = {X | Años ≥ 4.5, Hits ≥ 117.5}.



Un poco de terminología

- En línea con la analogía del árbol, las regiones R1, R2 y R3 se conocen como nodos terminales.
- Los árboles de decisión se suelen dibujar al revés, con las hojas en la parte inferior del árbol.
- Los puntos a lo largo del árbol donde se divide el espacio predictor se denominan nodos internos.
- En el árbol de *Hitters*, los dos nodos internos están indicados por los textos Años < 4.5 y Hits < 117.5.

Interpretando resultados

- Los años de experiencia es el factor más importante para determinar el salario, y los jugadores con menos experiencia ganan salarios más bajos que los jugadores más experimentados.
- Dado que un jugador es menos experimentado, el número de Hits que hizo el año anterior parece tener poco impacto en su salario.
- Sin embargo, entre los jugadores que han estado en las ligas mayores durante cinco o más años, el número de Hits hechos el año anterior afecta el salario, y aquellos que hicieron más Hits tienden a tener salarios más altos.
- Claramente, es una simplificación, pero comparado con un modelo de regresión, es fácil de visualizar, interpretar y explicar.

Detalles del proceso de construcción del árbol

- Dividimos el espacio del predictor, es decir, el conjunto de valores posibles para X_1, X_2, ..., X_p, en J regiones distintas y no superpuestas, R_1, R_2, ..., R_J.
- Para cada observación que cae en la región R_j, hacemos la misma predicción, que es simplemente el promedio de los valores de respuesta para las observaciones de entrenamiento en R_j.
- En teoría, las regiones podrían tener cualquier forma. Sin embargo, optamos por dividir el espacio predictor en rectángulos de alta dimensión, o cajas, para facilitar la interpretación del modelo predictivo resultante.
- El objetivo es encontrar cajas R_1, ..., R_I que minimicen el RSS, dado por:

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

donde $\hat{y}_{R_j}$ es la respuesta media para las observaciones de entrenamiento dentro de la j-ésima caja.

Más detalles

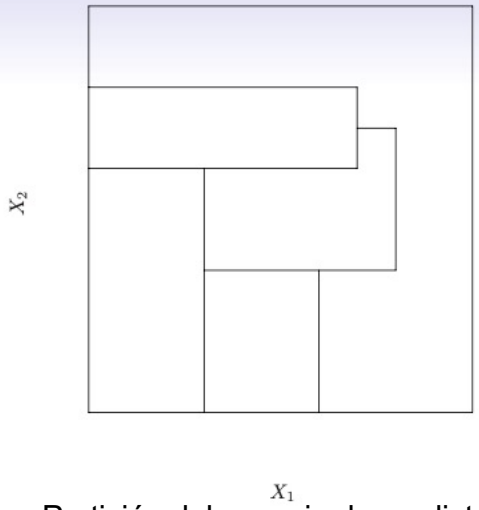
- El problema de considerar todas las posibles particiones del espacio de características es NP-completo
  - Necesitamos una estrategia to—down y voraz, denominada separación binaria recursiva (recursive binary splitting)
- Es top-down porque empezamos en la raíz, con particiones sucesivas del espacio de predictores (cada separación se indica mediante dos nuevas ramas en el arbol)
- Es voraz porque siempre escogemos el atributo y split más prometedor cada vez en lugar de mirar más adelante y ser paciente para tomar esa decisión

Más detalles

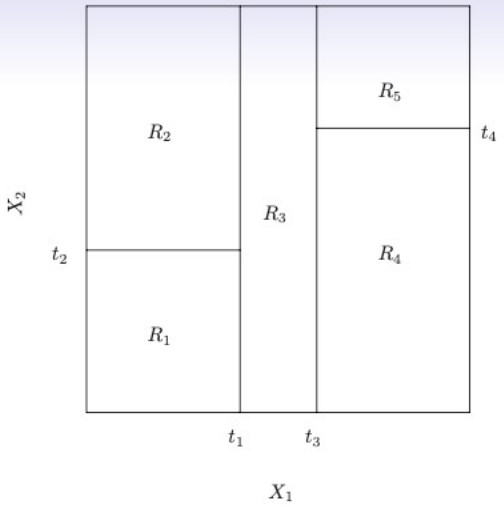
- Primero, seleccionamos el predictor X_j y el punto de corte s de tal manera que dividir el espacio del predictor en las regiones $\{X| X_j < s\}$ y $\{X| X_j \geq s\}$ conduzca a la mayor reducción posible en el RSS.
- Luego, repetimos el proceso, buscando el mejor predictor y el mejor punto de corte para dividir los datos nuevamente y minimizar el RSS dentro de cada una de las regiones resultantes.
- Sin embargo, esta vez, en lugar de dividir todo el espacio del predictor, dividimos una de las dos regiones previamente identificadas. Ahora tenemos tres regiones.
- Nuevamente, buscamos dividir una de estas tres regiones aún más, para minimizar el RSS. El proceso continúa hasta que se alcanza un criterio de parada; por ejemplo, podemos continuar hasta que ninguna región contenga más de cinco observaciones.

¿Y cómo hace inferencia?

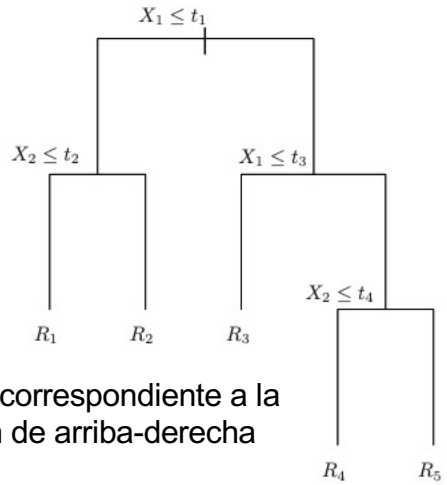
- Predecimos la respuesta para una observación de prueba dada usando el promedio de las observaciones de entrenamiento en la región a la que pertenece esa observación de prueba.



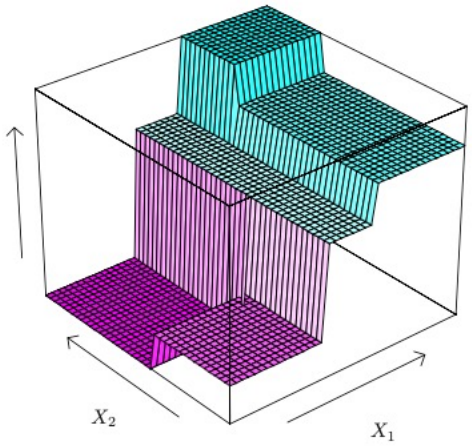
Partición del espacio de predictores imposible de construir mediante separación binaria recursiva



Partición del espacio de predictores del árbol de abajo



El árbol correspondiente a la partición de arriba-derecha



La superficie de decision del árbol

Poda de un árbol

- El proceso descrito anteriormente puede producir buenas predicciones en el conjunto de entrenamiento, pero es probable que se sobreajuste a los datos, lo que lleva a un rendimiento deficiente en el conjunto de prueba.
- Un árbol más pequeño con menos divisiones (es decir, menos regiones R1, ..., RJ) podría conducir a una menor varianza y una mejor interpretación, a cambio de un poco de sesgo.
- Una estrategia mejor es construir un árbol grande T0 y luego podarlo para obtener un subárbol.
- Esta estrategia podría funcionar mejor en árboles pequeños pero es cortoplacista ya que un split aparentemente malo podría resultar bueno a la larga teniendo en cuenta los splits subsiguientes (más reducción en RSS posterior)

Más sobre poda

- Una estrategia mejor es construir un árbol muy grande T_0 y luego podarlo para obtener un subárbol.
- La **poda por complejidad de costos** — también conocida como **poda por el eslabón más débil** — se utiliza para hacer esto.
- Consideramos una secuencia de árboles indexados por un parámetro de ajuste no negativo α. Para cada valor de α, corresponde un subárbol $T \subseteq T_0$ tal que

$$\sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$

es lo más pequeño posible. Aquí, $|T|$ indica el número de nodos terminales del árbol T, R_m es el rectángulo (es decir, el subconjunto del espacio predictor) correspondiente al m-ésimo nodo terminal, y $\hat{y}_{R_m}$ es el promedio de las observaciones de entrenamiento en R_m.

¿Cuál es el mejor árbol?

- El parámetro de ajuste α controla una compensación entre la complejidad del subárbol y su ajuste a los datos de entrenamiento.
- Seleccionamos un valor óptimo α^* usando validación cruzada.
- Con ese α^* y el dataset total obtenemos un árbol final

Resumen del algoritmo total

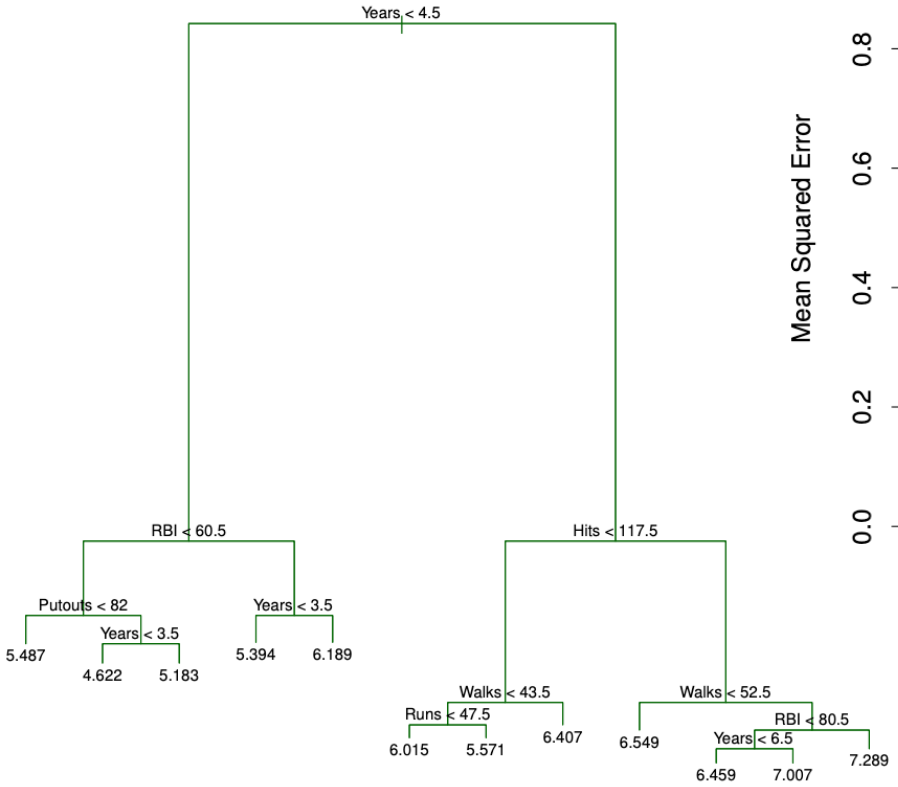
Resumen: algoritmo de árboles

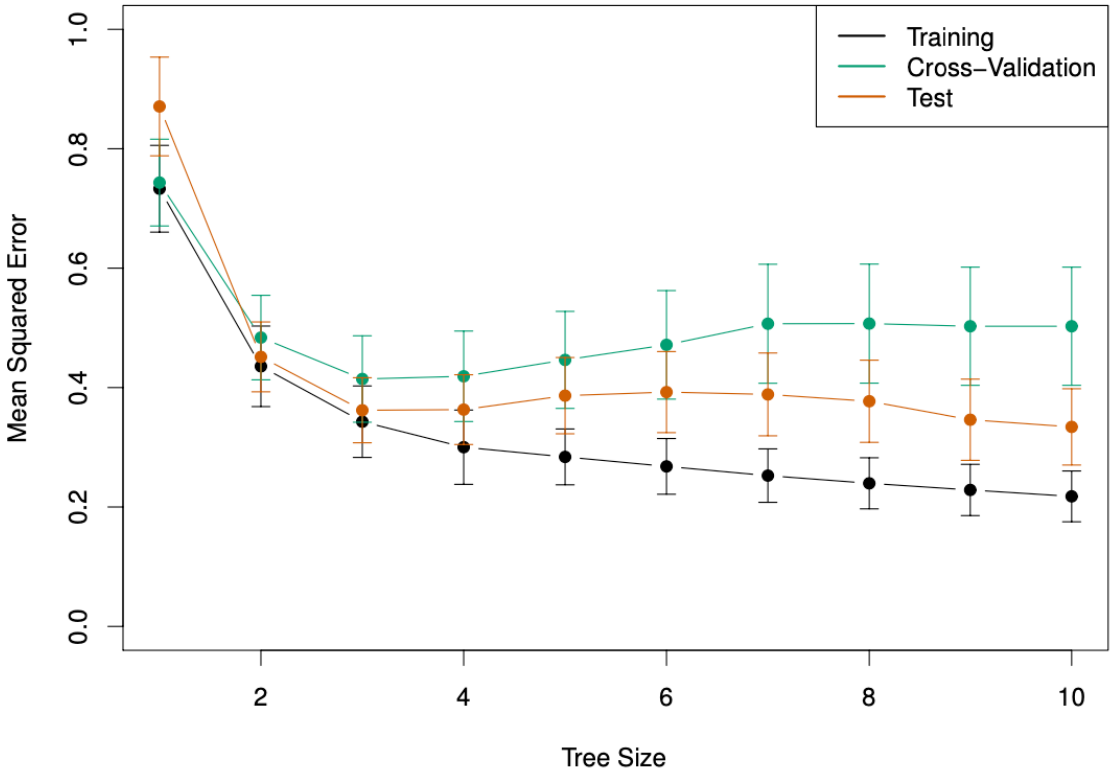
1. Utiliza la división binaria recursiva para desarrollar un árbol grande en los datos de entrenamiento, deteniéndose solo cuando cada nodo terminal tenga menos de un número mínimo de observaciones.
2. Aplica la poda por complejidad de costos al árbol grande para obtener una secuencia de los mejores subárboles, en función de α.
3. Utiliza validación cruzada de K-pliegues para elegir α. Para cada $k = 1, \ldots, K$:
   - 3.1 Repite los pasos 1 y 2 en la fracción $\frac{K-1}{K}$ de los datos de entrenamiento, excluyendo el k-ésimo pliegue.
   - 3.2 Evalúa el error cuadrático medio de predicción en los datos del k-ésimo pliegue excluido, en función de α.
   - Promedia los resultados y elige α para minimizar el error promedio.
4. Devuelve el subárbol del paso 2 que corresponde al valor de α elegido.

Continuamos con el ejemplo del béisbol

- Primero, dividimos aleatoriamente el conjunto de datos a la mitad, obteniendo 132 observaciones en el conjunto de entrenamiento y 131 observaciones en el conjunto de prueba.
- Luego, construimos un árbol de regresión grande en los datos de entrenamiento y variamos α para crear subárboles con diferentes números de nodos terminales.
- Finalmente, realizamos una validación cruzada de seis pliegues para estimar el MSE validado de los árboles en función de α.

Resultados


Years < 4.5
RBI < 60.5
Hits < 117.5
Putouts < 82
Years < 3.5
Years < 3.5
Walks < 43.5
Walks < 52.5
Runs < 47.5
RBI < 80.5
Years < 6.5
5.487
4.622
5.183
5.394
6.189
6.015
5.571
6.407
6.549
6.459
7.007
7.289


Training
Cross–Validation
Test
Mean Squared Error
Tree Size
0.0
0.2
0.4
0.6
0.8
1.0
2
4
6
8
10

Árboles de clasificación

- Muy similar a un árbol de regresión, excepto que se usa para predecir una respuesta cualitativa en lugar de una cuantitativa.
- Para un árbol de clasificación, predecimos que cada observación pertenece a la clase más común de las observaciones de entrenamiento en la región a la que pertenece.
- Al igual que en la configuración de regresión, usamos la división binaria recursiva para desarrollar un árbol de clasificación.
- En el contexto de clasificación, RSS no se puede usar como criterio para realizar las divisiones binarias.
- Una alternativa natural a RSS es la tasa de error de clasificación, que es simplemente la fracción de las observaciones de entrenamiento en esa región que no pertenecen a la clase más común:

$$E = 1 - \max_k(p_{mk})$$

- donde p_{mk} representa la proporción de observaciones de entrenamiento en la región mth que pertenecen a la clase kth.
- Sin embargo, el error de clasificación no es suficientemente sensible para el crecimiento del árbol, y en la práctica se prefieren otras dos medidas.

Índice Gini y entropía

- El índice de Gini se define como:

$$G = \sum_{k=1}^{K} p_{mk}(1 - p_{mk})$$

una medida de la variación total entre las K clases. El índice de Gini toma un valor pequeño si todos los p_{mk} están cerca de cero o uno.

- Por esta razón, el índice de Gini se conoce como una medida de la pureza del nodo: un valor pequeño indica que un nodo contiene predominantemente observaciones de una sola clase.
- Una alternativa al índice de Gini es la entropía cruzada, dada por:

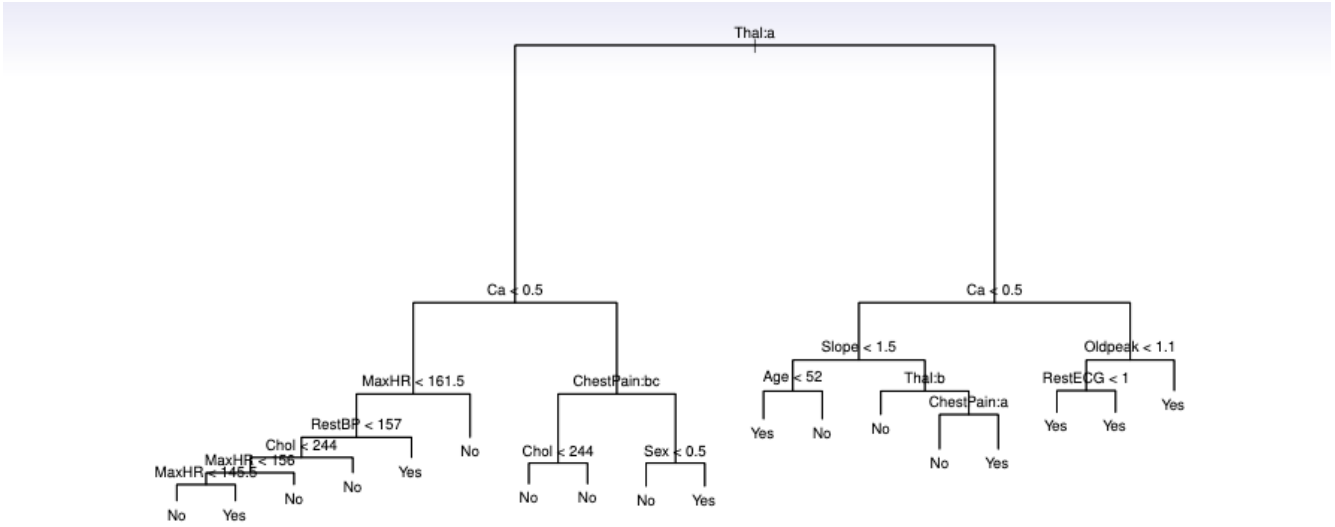
$$D = -\sum_{k=1}^{K} p_{mk} \log(p_{mk})$$

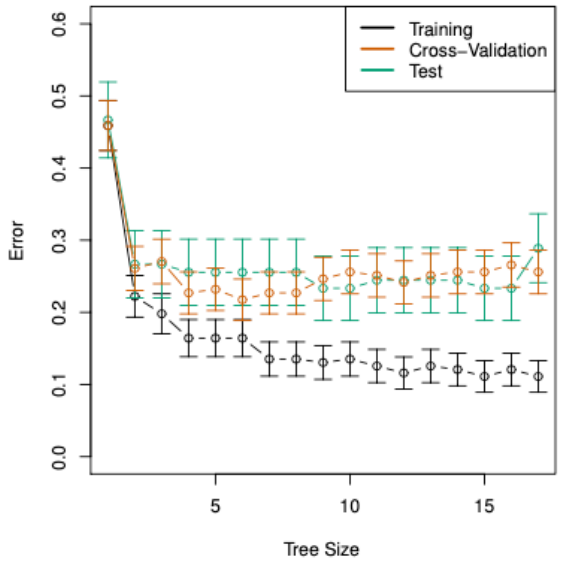
Resulta que el índice de Gini y la entropía cruzada son muy similares numéricamente.

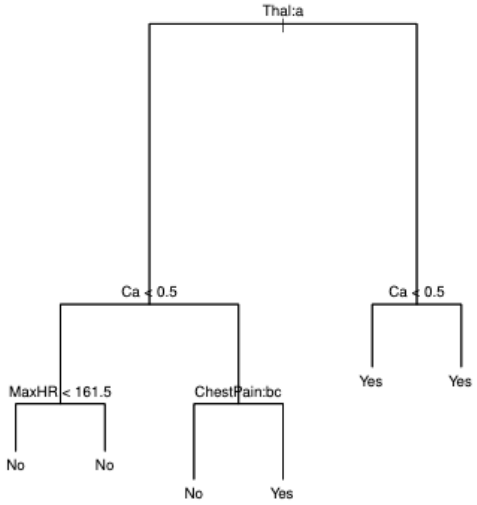
Nuevo ejemplo: heart dataset

- Estos datos contienen un resultado binario HD para 303 pacientes que se presentaron con dolor en el pecho.
- Un valor de resultado de ”Yes" indica la presencia de enfermedad cardíaca basada en una prueba angiográfica, mientras que "No" significa que no hay enfermedad cardíaca.
- Hay 13 predictores, incluyendo Age, Sex, Chol (una medida de colesterol) y otras mediciones de función cardíaca y pulmonar.
- La validación cruzada produce un árbol con seis nodos terminales.

Resultados


Thal:a
Ca < 0.5
Ca < 0.5
MaxHR < 161.5
ChestPain:bc
Slope < 1.5
Oldpeak < 1.1
Age < 52
Thal:b
RestECG < 1
Yes
RestBP < 157
Chol < 244
Chol < 244
Sex < 0.5
ChestPain:a
MaxHR < 156
MaxHR < 145.5
No
No
No
Yes
No
No
No
Yes
Yes
No
No
Yes
Yes
Yes
No
Yes
No
Yes
No
Yes


Training
Cross–Validation
Test
Error
0.0
0.1
0.2
0.3
0.4
0.5
0.6
5
10
15
Tree Size

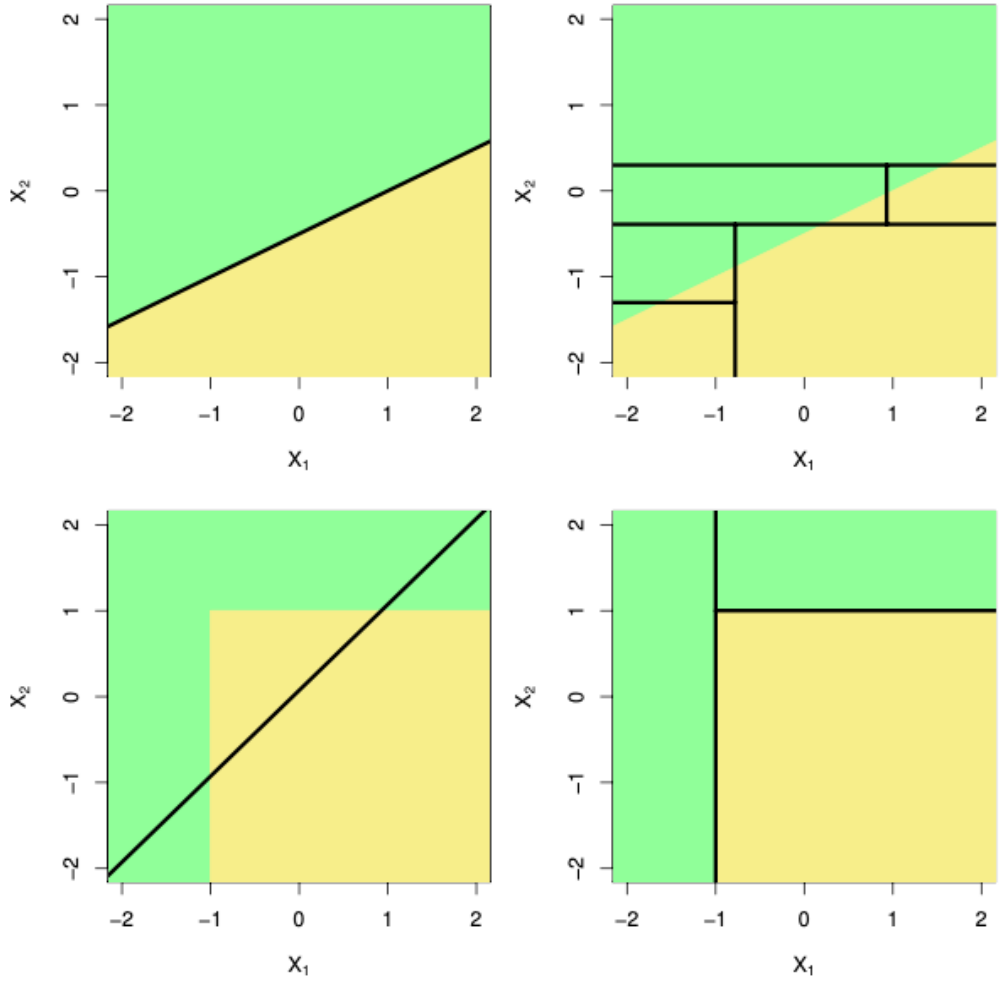

Thal:a
Ca < 0.5
Ca < 0.5
MaxHR < 161.5
ChestPain:bc
Yes
Yes
No
No
No
Yes

Sesgo de representación

- Árboles versus modelos lineales

Comparación gráfica entre modelos lineales y modelos basados en árboles, mostrando los límites de decisión lineales y no lineales en distintos contextos

- Left: linear
- Right: tree
- Up: Linear
- Down: tree



Ventajas y desventajas

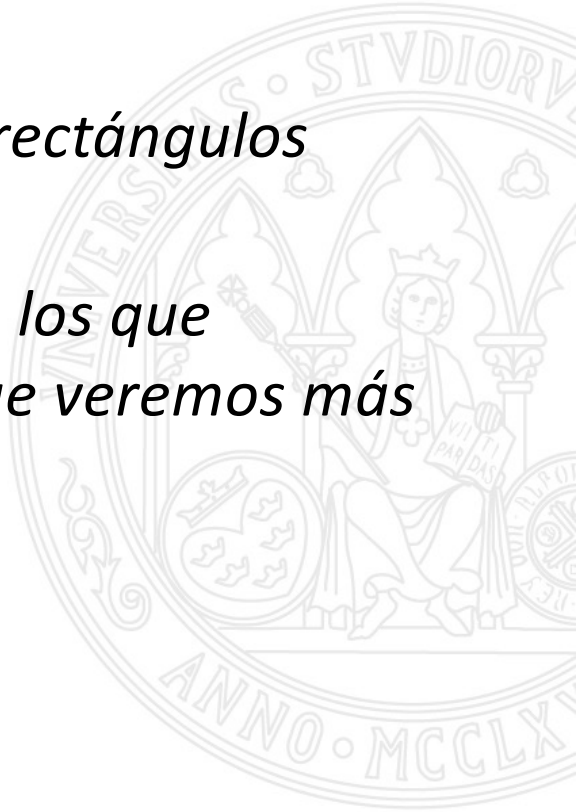
- **Ventajas**:Los árboles, como modelo, son muy fáciles de explicar/transmitir. Más que la regresión lineal.
- Quizás los árboles de decisión reflejan más de cerca la toma de decisiones humanas que otros enfoques de regresión y clasificación.
- Los árboles se pueden mostrar gráficamente y son fácilmente interpretables incluso por alguien sin experiencia (especialmente si son pequeños).
- Los árboles pueden manejar fácilmente predictores cualitativos sin la necesidad de crear variables ficticias.
- **Desventajas**: no tienen el mismo nivel de precisión predictiva que algunos de los otros enfoques de regresión y clasificación vistos en la asignatura!!!

Al agregar muchos árboles de decisión, se puede mejorar sustancialmente el rendimiento predictivo de los árboles. Introducimos estos conceptos a continuación.

Conclusiones

UNIVERSIDAD DE MURCIA

- *Fáciles de interpreter, fáciles de construir*
- *Suelen tener una capacidad predictiva baja (alto sesgo, baja varianza)*
- *Las fronteras de decision están formadas por rectángulos adyacentes*
- *Podemos usarlos como modelos básicos sobre los que construir ensamblajes (boosting y bagging, que veremos más adelante)*



Referencias

UNIVERSIDAD DE MURCIA

- G. James, D. Witten, T. Hastie, y R. Tibshirani, An Introduction to Statistical Learning with Applications in R. Springer, 2013.
  - Chapter 8, Sec. 8.1.
- T. Hastie, R. Tibshirani, y J. Friedman, The elements of statistical learning. Springer, 2009.
  - Chapter 9, Sec. 9.2.