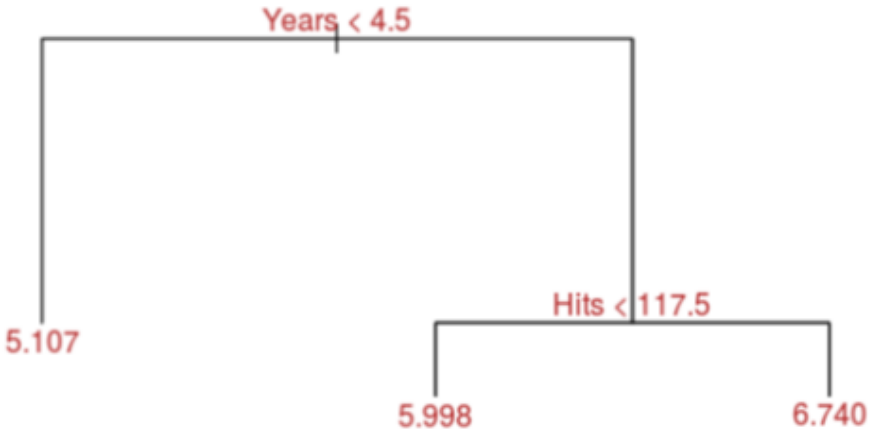
Introducción.

1. Los métodos predictivos como la regresión lineal o polinómica generan modelos globales en los que una única ecuación se aplica a todo el espacio muestral. Cuando el estudio implica múltiples predictores, que interaccionan entre ellos de forma compleja y no lineal, es muy difícil encontrar un modelo global que sea capaz de reflejar la relación entre las variables. Existen métodos de ajuste no lineal (step functions, splines,...) que combinan múltiples funciones y que realizan ajustes locales, sin embargo, suelen ser difíciles de interpretar. Los métodos estadísticos basados en árboles engloban a un conjunto de técnicas supervisadas no paramétricas que consiguen segmentar el espacio de los predictores en regiones simples, dentro de las cuales es más sencillo manejar las interacciones. Los métodos basados en árboles se han convertido en uno de los referentes dentro del ámbito predictivo debido a los buenos resultados que generan en ámbitos muy diversos.

2. Árboles de regresión

Los árboles de regresión son el subtipo de árboles de predicción que se aplica cuando la variable respuesta es continua. En términos generales, en el entrenamiento de un árbol de regresión, las observaciones se van distribuyendo por bifurcaciones (nodos) generando la estructura del árbol hasta alcanzar un nodo terminal. Cuando se quiere predecir una nueva observación, se recorre el árbol acorde al valor de sus predictores hasta alcanzar uno de los nodos terminales. La predicción del árbol es la media de la variable respuesta de las observaciones de entrenamiento que están en ese mismo nodo terminal.

La forma más sencilla de entender la idea detrás de los árboles de regresión es mediante de un ejemplo simplificado. El set de datos Hitter contiene información sobre 322 jugadores de béisbol de la liga profesional. Entre las variables registradas para cada jugador se encuentran: el salario (*Salary*), años de experiencia (*Years*) y el número de bateos durante los últimos años (*Hits*). Utilizando estos datos, se quiere predecir el salario (en unidades logarítmicas) de un jugador en base a su experiencia y número de bateos. El árbol resultante se muestra en la siguiente imagen:



La interpretación del árbol se hace en sentido descendente, la primera división es la que separa a los jugadores en función de si superan o no los 4.5 años de experiencia, la segunda división está en función de si superan o no los 117.5 bateos.

- $ 5.107. Es el salario promedio de todos los jugadores que no superan los 4.5 años de experiencia.
- $ 5.998. Es el salario promedio de todos los jugadores que tienen o superan los 4.5 años de experiencia y han conseguido menos de 117.5 bateos.
- $ 6.740. Es el salario promedio de todos los jugadores que tienen o superan los 4.5 años de experiencia y han conseguido igual o más de 117.5 bateos.

Los resultados de las estratificaciones han generado 3 regiones que pueden identificarse con la siguiente nomenclatura:

- $R1=\{X|Year<4.5\}$: jugadores que han jugado menos de 4.5 años.
- $R2=\{X|Year\geq 4.5,\ Hits<117.5\}$: jugadores que han jugado 4.5 años o más y que han conseguido menos de 117.5 bateos.
- $R3=\{X|Year\geq 4.5,\ Hits\geq 117.5\}$: jugadores que han jugado 4.5 años o más y que han conseguido 117.5 o más bateos.

A las regiones $R1$, $R2$ y $R3$ se les conoce como nodos terminales o leaves del árbol, a los puntos en los que el espacio de los predictores sufre una división como internal nodes o splits y a los segmentos que conectan dos nodos como branches o ramas.

Descripción de los resultados: Viendo el árbol anterior, la interpretación del modelo es, la variable más importante a la hora de determinar el salario de un jugador es el número de años de experiencia, los jugadores con más experiencia ganan más. Entre los jugadores con menos años de experiencia, el número de bateos logrados en los años previos no tiene mucho impacto en el salario, sin embargo, sí lo tiene para jugadores con cuatro años y medio o más de experiencia. Para estos últimos, a mayor número de bateos logrados mayor salario. Con este ejemplo, queda patente que la principal ventaja de los árboles de decisión frente a otros métodos de regresión es su fácil interpretación y la gran utilidad de su representación gráfica.

3. Entrenamiento del árbol

El proceso de construcción de un árbol de predicción o clasificación se divide en dos etapas:

- División sucesiva del espacio de los predictores generando regiones no solapantes (nodos terminales) $R1$, $R2$, $R3$, …, Rj. Aunque, desde el punto de vista teórico las regiones podrían tener cualquier forma, si se limitan a regiones rectangulares (de múltiples dimensiones), se simplifica en gran medida el proceso de construcción y se facilita la interpretación.
- Predicción de la variable respuesta en cada región.

A pesar de la sencillez con la que se puede resumir el proceso de construcción de un árbol, es necesario establecer una metodología que permita crear las regiones $R1$, $R2$, $R3$, …, Rj, o lo que es equivalente, decidir donde se introducen las divisiones.

En el caso de los árboles de regresión, el criterio más frecuentemente empleado para **identificar las divisiones es el Residual Sum of Squares (RSS).** El objetivo es encontrar las J regiones ($R1$,…, Rj) que minimizan el Residual Sum of Squares (RSS) total:

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

donde $\hat{y}_{R_j}$ es la media de la variable respuesta en la región R_j. Una descripción menos técnica equivale a decir que se busca una distribución de regiones tal que, el sumatorio de las desviaciones al cuadrado entre las observaciones y la media de la región a la que pertenecen sea lo menor posible.

Desafortunadamente, no es posible considerar todas las posibles particiones del espacio de los predictores. Por esta razón, se recurre a lo que se conoce como recursive binary splitting (división binaria recursiva). Esta solución sigue la misma idea que la selección de predictores stepwise (backward o fordward) en regresión lineal múltiple, no evalúa todas las posibles regiones, pero, alcanza un buen balance computación-resultado.

4. Recursive binary splitting

El objetivo del método recursive binary splitting es encontrar en cada iteración el predictor X_j y el punto de corte (umbral) s tal que, si se distribuyen las observaciones en las regiones $\{X|Xj<s\}$ y $\{X|Xj\geq s\}$, se consigue la mayor reducción posible en el RSS. El algoritmo seguido es:

1. El proceso se inicia en lo más alto del árbol, donde todas las observaciones pertenecen a la misma región.
2. Se identifican todos los posibles puntos de corte (umbrales) s para cada uno de los predictores ($X1$, $X2$,..., Xp). En el caso de predictores cualitativos, los posibles puntos de corte son cada uno de sus niveles. Para predictores continuos, se ordenan de menor a mayor sus valores, el punto intermedio entre cada par de valores se emplea como punto de corte.
3. Se calcula el RSS total que se consigue con cada posible división identificada en el paso 2.

$$\sum_{i:x_i \epsilon R_1(j,s)} \left(y_i - \hat{y}_{R_1}\right)^2 + \sum_{i:x_i \epsilon R_2(j,s)} \left(y_i - \hat{y}_{R_2}\right)^2$$

   donde el primer término es el RSS de la región 1 y el segundo término es el RSS de la región 2, siendo cada una de las regiones el resultado de separar las observaciones acordes al predictor j y valor s.

4. Se selecciona el predictor X_j y el punto de corte S que resulta en el menor RSS total, es decir, que da lugar a las divisiones más homogéneas posibles. Si existen dos o más divisiones que consiguen la misma mejora, la elección entre ellas es aleatoria.
5. Se repiten de forma iterativa los pasos 1 a 4 para cada una de las regiones que se han creado en la iteración anterior hasta que se alcanza alguna norma de stop. Algunas de las más empleadas son: que ninguna región contenga un mínimo de n observaciones, que el árbol tenga un máximo de nodos terminales o que la incorporación del nodo reduzca el error en al menos un % mínimo.

Está metodología conlleva dos hechos:

- Que cada división óptima se identifica acorde al impacto que tiene en ese momento. No se tiene en cuenta si es la división que dará lugar a mejores árboles en futuras divisiones.
- En cada división se evalúa un único predictor haciendo preguntas binarias (si, no), lo que genera dos nuevas ramas del árbol por división. A pesar de que es posible evaluar divisiones más complejas, hacer una pregunta sobre múltiples variables a la vez es equivalente a hacer múltiples preguntas sobre variables individuales.

Nota: Los algoritmos que implementan recursive binary splitting suelen incorporar estrategias para evitar evaluar todos los posibles puntos de corte. Por ejemplo, para predictores continuos, primero se crea un histograma que agrupa los valores y luego se evalúan los puntos de corte de cada región del histograma (bin).

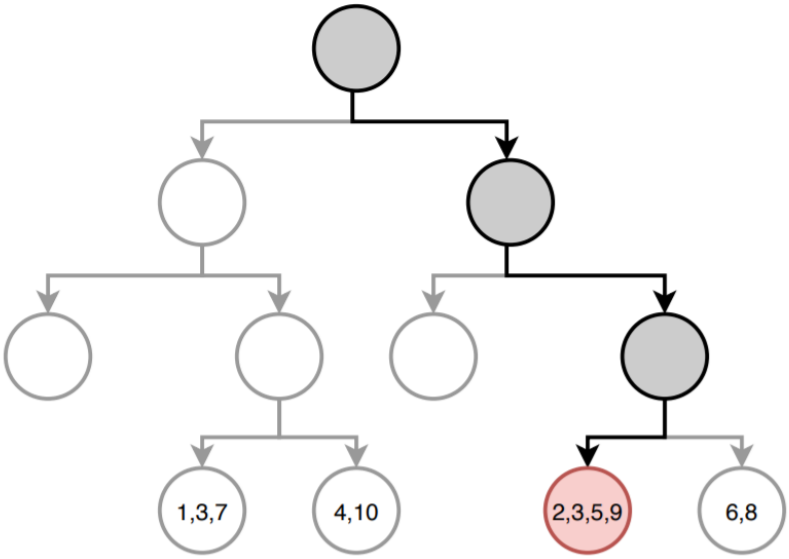
5. Predicción del árbol

Tras la creación de un árbol, las observaciones de entrenamiento quedan agrupadas en los nodos terminales. Para predecir una nueva observación, se recorre el árbol en función de los valores que tienen sus predictores hasta llegar a uno de los nodos terminales. En el caso de regresión, el valor predicho suele ser la media de la variable respuesta de las observaciones de entrenamiento que están en ese mismo nodo. Si bien la media es valor más empleado, se puede utilizar cualquier otro (mediana, cuantil...).

Supóngase que se dispone de 9 observaciones, cada una con un valor de variable respuesta Y y unos predictores X.

<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	
id	1	2	3	4	5	6	7	8	9
Y	10	18	24	8	2	9	16	10	20
X	...	...	...	...	...	...	...	...	...

La siguiente imagen muestra cómo sería la predicción del árbol para una nueva observación. El camino hasta llegar al nodo final está resaltado. En cada nodo terminal se detalla el índice de las observaciones de entrenamiento que forman parte.



El valor predicho por el árbol es la media de la variable respuesta Y de las observaciones con id: 2, 3, 5, 9.

$$\hat{u} = \frac{18 + 24 + 2 + 20}{4} = 16$$

Aunque la anterior es la forma más común de obtener las predicciones de un árbol de regresión, existe otra aproximación. La predicción de un árbol de regresión puede verse como una variante de vecinos cercanos en la que, solo las observaciones que forman parte del mismo nodo terminal que la observación predicha, tienen influencia. Siguiendo esta aproximación, la predicción del árbol se define como la media ponderada de todas las observaciones de entrenamiento, donde el peso de cada observación depende únicamente de si forma parte o no del mismo nodo terminal.

$$\hat{u} = \sum_{i=1}^{n} w_i Y_i$$

El valor de las posiciones del vector de pesos w es 1 para las observaciones que están en el mismo nodo y 0 para el resto. En este ejemplo sería:

$$w = (0,1,1,0,1,0,0,0,1,0)$$

Para que la suma de todos los pesos sea 1, se dividen por el número total de observaciones en el nodo terminal seleccionado, en este caso 4.

$$w = (0,\frac{1}{4},\frac{1}{4},0,\frac{1}{4},0,0,0,\frac{1}{4},0)$$

Así pues, el valor predicho es:

$$w = (0x10) + \left(\frac{1}{4}x18\right) + \left(\frac{1}{4}x24\right) + (0x8) + \left(\frac{1}{4}x2\right) + (0x9) + (0x16) + (0x19) + \left(\frac{1}{4}x20\right) + (0x14) = 16$$

Evitar el overfitting

El proceso de construcción de árboles descrito en las secciones anteriores tiende a reducir rápidamente el error de entrenamiento, es decir, el modelo se ajusta muy bien a las observaciones empleadas como entrenamiento. Como consecuencia, se genera un overfitting que reduce su capacidad predictiva al aplicarlo a nuevos datos. La razón de este comportamiento radica en la facilidad con la que los árboles se ramifican adquiriendo estructuras complejas. De hecho, si no se limitan las divisiones, todo árbol termina ajustándose perfectamente a las observaciones de entrenamiento creando un nodo terminal por observación. Existen dos estrategias para prevenir el problema de overfitting de los árboles: limitar el tamaño del árbol (parada temprana o early stopping) y el proceso de podado (pruning).

Controlar el tamaño del árbol (parada temprana)

El tamaño final que adquiere un árbol puede controlarse mediante reglas de parada que detengan la división de los nodos dependiendo de si se cumplen o no determinadas condiciones. El nombre de estas condiciones puede variar dependiendo del software o librería empleada, pero suelen estar presentes en todos ellos.

- **Observaciones mínimas para división**: define el número mínimo de observaciones que debe tener un nodo para poder ser dividido. Cuanto mayor el valor, menos flexible es el modelo.
- **Observaciones mínimas de nodo terminal:** define el número mínimo de observaciones que deben tener los nodos terminales. Su efecto es muy similar al de observaciones mínimas para división.
- **Profundidad máxima del árbol**: define la profundidad máxima del árbol, entendiendo por profundidad máxima el número de divisiones de la rama más larga (en sentido descendente) del árbol.
- **Número máximo de nodos terminales:** define el número máximo de nodos terminales que puede tener el árbol. Una vez alcanzado el límite, se detienen las divisiones. Su efecto es similar al de controlar la profundidad máxima del árbol.
- **Reducción mínima de error**: define la reducción mínima de error que tiene que conseguir una división para que se lleve a cabo.

A todos estos parámetros se les conoce como hiperparámetros porque no se aprenden durante el entrenamiento del modelo. Su valor tiene que ser especificado por el usuario en base a su conocimiento del problema y mediante el uso de validación cruzada.

Tree pruning

La estrategia de controlar el tamaño del árbol mediante reglas de parada tiene un inconveniente, el árbol se crece seleccionando la mejor división en cada momento.

Al evaluar las divisiones sin tener en cuenta las que vendrán después, nunca se elige la opción que resulta en el mejor árbol final, a no ser que también sea la que genera en ese momento la mejor división. A este tipo de estrategias se les conoce como greedy. Un ejemplo que ilustra el problema de este tipo de estrategia es el siguiente: supóngase que un coche circula por el carril izquierdo de una carretera de dos carriles en la misma dirección. En el carril que se encuentra hay muchos coches circulando a 100 km/h, mientras que el otro carril se encuentra vacío. A cierta distancia se observa que hay un vehículo circulando por el carril derecho a 20 km/h. Si el objetivo del conductor es llegar a su destino lo antes posible tiene dos opciones: cambiarse de carril o mantenerse en el que está. Una aproximación de tipo greedy evaluaría la situación en ese instante y determinaría que la mejor opción es cambiarse de carril y acelerar a más de 100 km/h, sin embargo, a largo plazo, esta no es la mejor solución, ya que una vez alcance al vehículo lento, tendrá que reducir mucho su velocidad.

Una alternativa no greedy que consigue evitar el overfitting consiste en generar árboles grandes, sin condiciones de parada más allá de las necesarias por las limitaciones computacionales, para después podarlos (pruning), manteniendo únicamente la estructura robusta que consigue un test error bajo. La selección del sub-árbol óptimo puede hacerse mediante cross-validation, sin embargo, dado que los árboles se crecen lo máximo posible (tienen muchos nodos terminales) no suele ser viable estimar el test error de todas las posibles sub-estructuras que se pueden generar. En su lugar, se recurre al cost complexity pruning o weakest link pruning.

Cost complexity pruning es un método de penalización de tipo Loss + Penalty, similar al empleado en ridge regression o lasso. En este caso, se busca el sub-árbol T que minimiza la ecuación

$$\sum_{j=1}^{|T|} \sum_{i \in R_j} \left(y_i - \hat{y}_{R_j}\right)^2 + \alpha |T|$$

donde |T| es el número de nodos terminales del árbol.

El primer término de la ecuación se corresponde con el sumatorio total de los residuos cuadrados RSS. Por definición, cuantos más nodos terminales tenga el modelo menor será esta parte de la ecuación. El segundo término es la restricción, que penaliza al modelo en función del número de nodos terminales (a mayor número, mayor penalización). El grado de penalización se determina mediante el tuning parameter α. Cuando α=0, la penalización es nula y el árbol resultante es equivalente al árbol original. A medida que se incrementa α la penalización es mayor y, como consecuencia, los árboles resultantes son de menor tamaño. El valor óptimo de α puede identificarse mediante cross validation**.**

6. Algoritmo para crear un árbol de regresión con pruning

1. Se emplea recursive binary splitting para crear un árbol grande y complejo (To) empleando los datos de training y reduciendo al máximo posible las

condiciones de parada. Normalmente se emplea como única condición de parada el número mínimo de observaciones por nodo terminal.

2. Se aplica el cost complexity pruning al árbol To para obtener el mejor sub-árbol en función de α. Es decir, se obtiene el mejor sub-árbol para un rango de valores de α.
3. Identificación del valor óptimo de α mediante k-cross-validation. Se divide el training data set en K grupos. Para k=1, …, k=K:
   – Repetir pasos 1 y 2 empleando todas las observaciones excepto las del grupo ki.
   – Evaluar el mean squared error para el rango de valores de α empleando el grupo ki.
   – Obtener el promedio de los K mean squared error calculados para cada valor α.
4. Seleccionar el sub-árbol del paso 2 que se corresponde con el valor α que ha conseguido el menor cross-validation mean squared error en el paso 3.

En el caso de los árboles de clasificación, en lugar de emplear la suma de residuos cuadrados como criterio de selección, se emplea alguna de las medidas de homogeneidad vistas en apartados anteriores.

Ventajas y desventajas

Ventajas

- Los árboles son fáciles de interpretar aun cuando las relaciones entre predictores son complejas. Su estructura se asemeja a la forma intuitiva en que clasificamos y predecimos las personas, además, no se requieren conocimientos estadísticos para comprenderlos.
- Los modelos basados en un solo árbol (no es el caso de random forest, boosting…) se pueden representar gráficamente aun cuando el número de predictores es mayor de 3.
- Los árboles pueden manejar tanto predictores cuantitativos como cualitativos sin tener que crear variables dummy.
- Al tratarse de métodos no paramétricos, no es necesario que se cumpla ningún tipo de distribución específica.
- Por lo general, requieren mucha menos limpieza y pre procesado de los datos en comparación a otros métodos de aprendizaje estadístico.
- No se ven muy influenciados por outliers.
- Si para alguna observación, el valor de un predictor no está disponible, a pesar de no poder llegar a ningún nodo terminal, se puede conseguir una

predicción empleando todas las observaciones que pertenecen al último nodo alcanzado. La precisión de la predicción se verá reducida pero al menos podrá obtenerse.

- Son muy útiles en la exploración de datos, permiten identificar de forma rápida y eficiente las variables más importantes.
- Son capaces de seleccionar predictores de forma automática.

Desventajas

- La capacidad predictiva de los modelos de regresión y clasificación basados en un único árbol es bastante inferior a la conseguida con otros modelos debido a su tendencia al overfitting. Sin embargo, existen técnicas más complejas que, haciendo uso de la combinación de múltiples árboles (bagging, random forest, boosting), consiguen mejorar en gran medida este problema.
- Cuando tratan con variables continuas, pierden parte de su información al categorizarlas en el momento de la división de los nodos. Por esta razón, suelen ser modelos que consiguen mejores resultados en clasificación que en regresión.
- Tal y como se describe más adelante, la creación de las ramificaciones de los árboles se consigue mediante el algoritmo de recursive binary splitting. Este algoritmo identifica y evalúa las posibles divisiones de cada predictor acorde a una determinada medida (RSS, Gini, entropía…). Los predictores continuos o predictores cualitativos con muchos niveles tienen mayor probabilidad de contener, solo por azar, algún punto de corte óptimo, por lo que suelen verse favorecidos en la creación de los árboles.

7. Paquetes R

Debido a sus buenos resultados, Random Forest y Gradient Boosting, se han convertido en los algoritmos de referencia cuando se trata con datos tabulares, de ahí que se hayan desarrollado múltiples implementaciones. Cada una tiene unas características que las hacen más adecuadas dependiendo del caso de uso. Algunos de los paquetes más empleados en R son:

- tree y rpart: permiten crear y representar árboles de regresión y clasificación.
- rpart.plot: permite crear representaciones detalladas de modelos creados con rpart.
- randomForest: dispone de los principales algoritmos para crear modelos random forest. Destaca por su fácil uso, pero no por su rapidez.