CLASIFICACIÓN CON ÁRBOLES DE DECISIÓN, I

Abraham Sánchez López
FCC/BUAP
Grupo MOVIS

MOVIS

Facultad de Ciencias de la Computación

Introducción, I

- No hay nada como el aire libre. Vivo en el campo y cuando paseo a mi perro por el bosque, recuerdo cuánto dependemos de los árboles. Los árboles producen la atmósfera que respiramos, crean hábitats para la vida silvestre, nos proporcionan alimento y son sorprendentemente buenos para hacer predicciones.
- Sí, has leído bien: los árboles son buenos para hacer predicciones. Pero antes de que le preguntes al abedul de tu jardín trasero los números de la lotería de la próxima semana, debo aclarar que me refiero a varios algoritmos de aprendizaje supervisado que utilizan una estructura de árbol ramificado.
- Esta familia de algoritmos se puede utilizar para resolver tareas de clasificación y regresión, pueden manejar predictores continuos y categóricos y, naturalmente, es adecuada para resolver problemas de clasificación multiclase.
- **Nota**: Recuerda que una variable predictiva es una variable que creemos que puede contener información sobre el valor de nuestra variable de resultado. Los predictores continuos pueden tener cualquier valor numérico en su escala de medición, mientras que las variables categóricas solo pueden tener valores/categorías finitas y discretas.

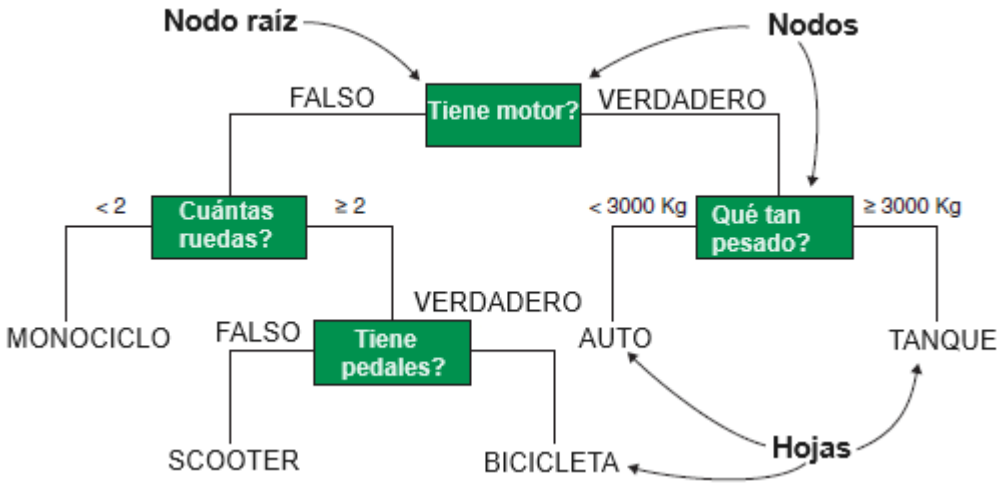
Introducción, II

- La premisa básica de todos los algoritmos de clasificación basados en árboles es que aprenden una secuencia de preguntas que separa los casos en diferentes clases. Cada pregunta tiene una respuesta binaria y los casos se enviarán por la rama izquierda o derecha según los criterios que cumplan.
- Puede haber ramas dentro de ramas; y una vez aprendido el modelo, se puede representar gráficamente como un árbol. ¿Has jugado alguna vez al juego 20 Preguntas, donde tienes que adivinar en qué objeto está pensando alguien haciendo preguntas de sí o no?
- ¿Qué pasa con el juego Guess Who, donde tienes que adivinar el personaje del otro jugador haciéndole preguntas sobre su apariencia? Estos son ejemplos de clasificadores basados en árboles.
- Al final de este tema, verás cómo se pueden utilizar modelos tan simples e interpretables para hacer predicciones. Terminaremos el tema destacando una debilidad importante de los árboles de decisión.

Algoritmo de partición recursiva, I

- En esta parte del documento, aprenderás cómo funcionan los algoritmos de árbol de decisión, y específicamente, el algoritmo de partición recursiva (rpart), para aprender una estructura de árbol.
- Imagine que deseas crear un modelo para representar la forma en que las personas se desplazan al trabajo, dadas las características del vehículo. Recopilas información sobre los vehículos, como cuántas ruedas tienen, si tienen motor y su peso.
- Podrías formular su proceso de clasificación como una serie de preguntas secuenciales. Cada vehículo se evalúa en cada pregunta y se mueve hacia la izquierda o hacia la derecha en el modelo dependiendo de cómo sus características satisfacen la pregunta. Un ejemplo de tal modelo se muestra en la figura del acetato 5.
- La estructura de un árbol de decisión. El nodo raíz es el nodo que contiene todos los datos antes de la división. Los nodos se dividen según un criterio de división en dos ramas, cada una de las cuales conduce a otro nodo. Los nodos que no se dividen más se llaman hojas.

Algoritmo de partición recursiva, II



- Observa que nuestro modelo tiene una estructura ramificada en forma de árbol, donde cada pregunta divide los datos en dos ramas. Cada rama puede generar preguntas adicionales, que tienen ramas propias.

- Las partes de preguntas del árbol se denominan *nodos*, y la primera pregunta/nodo se denomina *nodo raíz*. Los nodos tienen una rama que conduce hacia ellos y dos ramas que se alejan de ellos. Los nodos al final de una serie de preguntas se denominan nodos hoja u hojas. Los nodos de las hojas tienen una sola rama que conduce hacia ellos, pero ninguna rama que se aleje de ellos.

Algoritmo de partición recursiva, III

- Cuando un caso desciende por el árbol hasta llegar a un nodo de hoja, no avanza más y se clasifica como la clase mayoritaria dentro de esa hoja. Puede que te parezca extraño (al menos a mí me lo parece) que la raíz esté en la parte superior y las hojas en la parte inferior, pero así es como se suelen representar los modelos basados en árboles.
- **Nota**: Aunque no se muestra en este pequeño ejemplo, está perfectamente bien (y es común) tener preguntas sobre la misma característica en diferentes partes del árbol.
- Todo esto parece sencillo hasta ahora. Pero en el ejemplo simplista anterior, podríamos haberlo construido nosotros mismos a mano. (¡De hecho, lo hicimos!) Por lo tanto, los modelos basados en árboles no necesariamente se aprenden mediante el aprendizaje automático.
- Un árbol de decisiones podría ser un proceso de recursos humanos establecido para abordar medidas disciplinarias, por ejemplo. Podrías tener un enfoque basado en árboles para decidir qué vuelo comprar (si el precio está por encima de su presupuesto, si la aerolínea es confiable, si la comida es terrible, etc.).

Algoritmo de partición recursiva, IV

- Entonces, ¿cómo podemos aprender automáticamente la estructura de un árbol de decisión para conjuntos de datos complejos con muchas características? Revisa el algoritmo rpart.

- **Nota**: Los modelos basados en árboles se pueden utilizar tanto para tareas de clasificación como de regresión, por lo que es posible que los veas descritos como árboles de clasificación y regresión (CART). Sin embargo, CART es un algoritmo registrado cuyo código es propietario. El algoritmo rpart es simplemente una implementación de código abierto de CART.

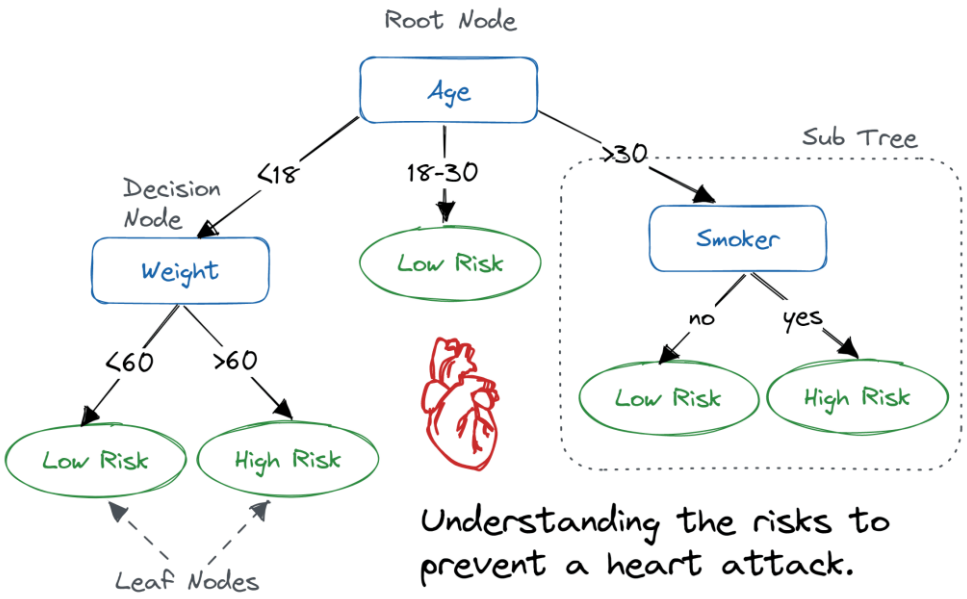
- En cada etapa del proceso de construcción del árbol, el algoritmo rpart considera todas las variables predictoras y selecciona el predictor que hace el mejor trabajo al discriminar las clases.

- Comienza en la raíz y luego, en cada rama, busca nuevamente la siguiente característica que discriminará mejor las clases de los casos que tomaron esa rama. Pero, ¿cómo decide rpart cuál es la mejor característica en cada división? Esto se puede hacer de diferentes maneras y rpart ofrece dos enfoques: la diferencia de entropía (llamada ganancia de información) y la diferencia en el índice de Gini (llamada ganancia de Gini).

Algoritmo de partición recursiva, V

- Los dos métodos suelen dar resultados muy similares; pero el índice de Gini (que lleva el nombre del sociólogo y estadístico Corrado Gini) es un poco más rápido de calcular, por lo que nos centraremos en él.

- *Consejo*: El índice de Gini es el método predeterminado que utiliza rpart para decidir cómo dividir el árbol. Si te preocupa perderte el modelo de mejor rendimiento, siempre puedes comparar el índice de Gini y la entropía durante el ajuste de hiperparámetros.

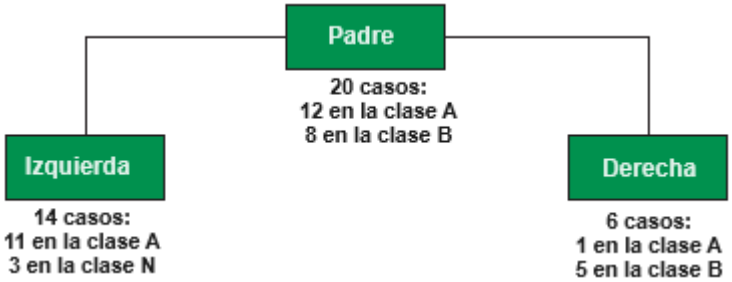


División del árbol con la ganancia de Gini, I

- En esta sección, se mostrará cómo se calcula la ganancia de Gini para encontrar la mejor división para un nodo en particular al hacer crecer un árbol de decisión.
- La entropía y el índice de Gini son dos formas de intentar medir la misma cosa: *la impureza*. La impureza es una medida de cuán heterogéneas son las clases dentro de un nodo.
- **Nota**: Si un nodo contiene solo una clase (lo que lo convertiría en una hoja), se diría que es puro.
- Al estimar la impureza (con cualquier método que elijas) que resultaría del uso de cada variable predictiva para la siguiente división, el algoritmo puede elegir la característica que dará como resultado la impureza más pequeña.
- Dicho de otra manera, el algoritmo elige la característica que dará como resultado nodos posteriores que sean lo más homogéneos posible.
- Entonces, ¿cómo es el índice de Gini?

División del árbol con la ganancia de Gini, II

- La figura muestra un ejemplo de división. Tenemos 20 casos en un nodo padre que pertenece a dos clases, A y B. Dividimos el nodo en dos hojas según algún criterio.
- En la hoja izquierda tenemos 11 cajas de clase A y 3 de clase B. En la hoja derecha tenemos 5 de clase B y 1 de clase A.

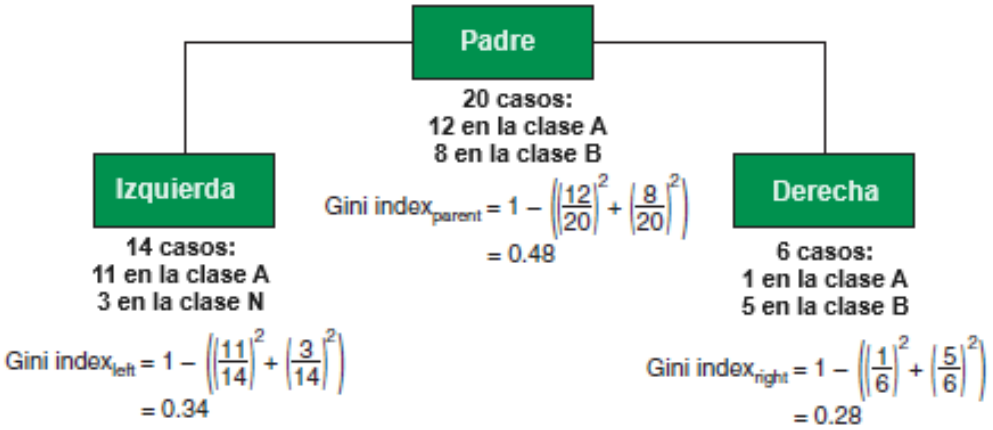


- Queremos saber la ganancia de Gini de esta división. La ganancia de Gini es la diferencia entre el índice de Gini del nodo principal y el índice de Gini de la división. Mirando nuestro ejemplo en la figura, el índice de Gini para cualquier nodo se calcula como

$$\text{Indice Gini} = 1 - (p(A)^2 + p(B)^2)$$

División del árbol con la ganancia de Gini, III

- donde $p(A)$ y $p(B)$ son las proporciones de casos pertenecientes a las clases A y B, respectivamente. Así, los índices de Gini para el nodo padre y las hojas izquierda y derecha se muestran en la siguiente figura.



- Ahora que tenemos los índices de Gini para las hojas izquierda y derecha, podemos calcular el índice de Gini para la división en su conjunto. El índice de Gini de la división es la suma de los índices de Gini izquierdo y derecho multiplicada por la proporción de casos que aceptaron del nodo principal:

$$\text{Indice Gini}_{split} = p(izq) \times \text{Indice Gini}_{izq} + p(der) \times \text{Indice Gini}_{der}$$

División del árbol con la ganancia de Gini, IV

$$\text{Indice Gini}_{split} = \frac{14}{20} \times 0.34 + \frac{6}{20} \times 0.28 = 0.32$$

- Y la ganancia de Gini (la diferencia entre los índices de Gini del nodo padre y la división) es simplemente

$$\text{Ganancia Gini} = 0.48 - 0.32 = 0.16$$

- donde 0.48 es el índice de Gini del padre, como se calcula en la figura anterior.
- La ganancia de Gini en un nodo particular se calcula para cada variable predictiva y el predictor que genera la mayor ganancia de Gini se utiliza para dividir ese nodo. Este proceso se repite para cada nodo a medida que crece el árbol.

Generalización del índice de Gini

- En este ejemplo, hemos considerado sólo dos clases, pero el índice de Gini de un nodo es fácilmente calculable para problemas que tienen muchas clases. En esta situación, la ecuación del índice de Gini se generaliza a

$$\text{Indice Gini} = 1 - \sum_{k=1}^{K} p(clase_k)^2$$

- que es solo una forma elegante de decir que calculamos $p(\text{clase}_K)^2$ para cada clase desde 1 hasta K (el número de clases), las sumamos todas y restamos este valor de 1.

- Si estás interesado, la ecuación de la entropía es

$$\text{entropía} = \sum_{k=1}^{K} -p(clase_k) \times \log_2 p(clase_k)$$

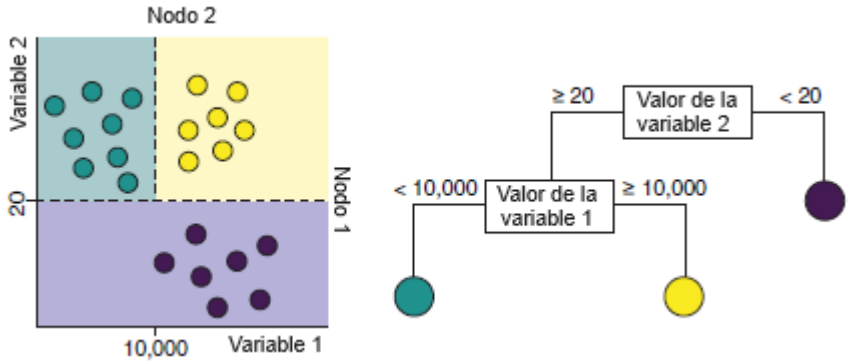
- que es simplemente una forma elegante de decir que calculamos $-p(\text{clase}) \times \log 2\ p(\text{clase})$ para cada clase desde 1 hasta K (el número de clases) y las sumamos todas (lo que se convierte en una resta porque el primer término es negativo).

- En cuanto a la ganancia de Gini, la ganancia de información se calcula como la entropía del padre menos la entropía de la división (que se calcula exactamente de la misma manera que el índice de Gini para la división).

Predictores categóricos continuos y multinivel, I

- ¿Qué pasa con los predictores categóricos continuos y multinivel?
- En esta sección, se mostrará cómo se eligen las divisiones para variables predictoras continuas y categóricas. Cuando una variable predictora es dicotómica (tiene sólo dos niveles), es bastante obvio cómo usarla para una división: los casos con un valor van hacia la izquierda y los casos con el otro valor van hacia la derecha.
- Los árboles de decisión también pueden dividir los casos utilizando variables continuas, pero ¿qué valor se elige como punto de división?
- Echa un vistazo al ejemplo de la figura del acetato 15. Tenemos casos de tres clases graficados contra dos variables continuas.
- El espacio de características se divide en rectángulos por cada nodo. En el primer nodo, los casos se dividen en aquellos con un valor de variable 2, mayor o menor que 20. Los casos que llegan al segundo nodo se dividen en aquellos con un valor de variable 1, mayor o menor de 10,000.

Predictores categóricos continuos y multinivel, II



- Cómo se realiza la división para predictores continuos. Los casos que pertenecen a tres clases se trazan frente a dos variables continuas. El primer nodo divide el espacio de características en rectángulos según el valor de la variable 2. El segundo nodo divide aún más el espacio de características de la variable 2 ≥ 20 en rectángulos según el valor de la variable 1.

- **Nota**: Observa que las variables están en escalas muy diferentes. El algoritmo rpart no es sensible a que las variables estén en diferentes escalas, por lo que no es necesario escalar ni centrar los predictores.

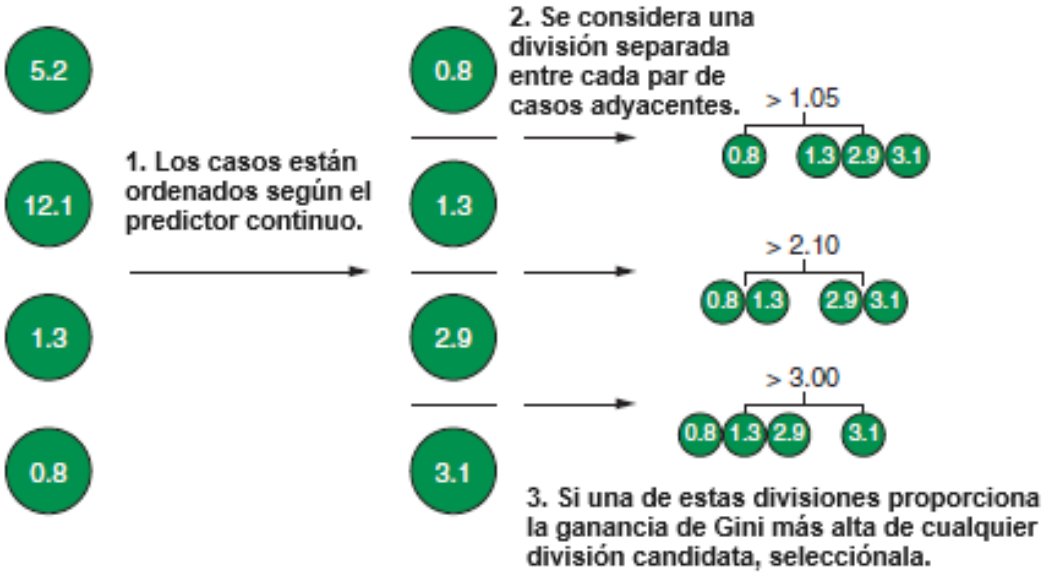
Predictores categóricos continuos y multinivel, III

- Pero, ¿cómo se elige el punto de división exacto para un predictor continuo? Bueno, los casos en el conjunto de entrenamiento están ordenados según la variable continua y la ganancia de Gini se evalúa para el punto medio entre cada par de casos adyacentes.
- Si la mayor ganancia de Gini entre todas las variables predictivas es uno de estos puntos medios, entonces éste se elige como la división para ese nodo. Esto se ilustra en la figura del acetato 18.
- Se utiliza un procedimiento similar para predictores categóricos con más de dos niveles. Primero, se calcula el índice de Gini para cada nivel del predictor (utilizando la proporción de cada clase que tiene ese valor del predictor).
- Los niveles de los factores se organizan según sus índices de Gini y la ganancia de Gini se evalúa para dividirla entre cada par de niveles adyacentes.
- Mira con detalle el ejemplo de la figura del acetato 19.

Predictores categóricos continuos y multinivel, IV

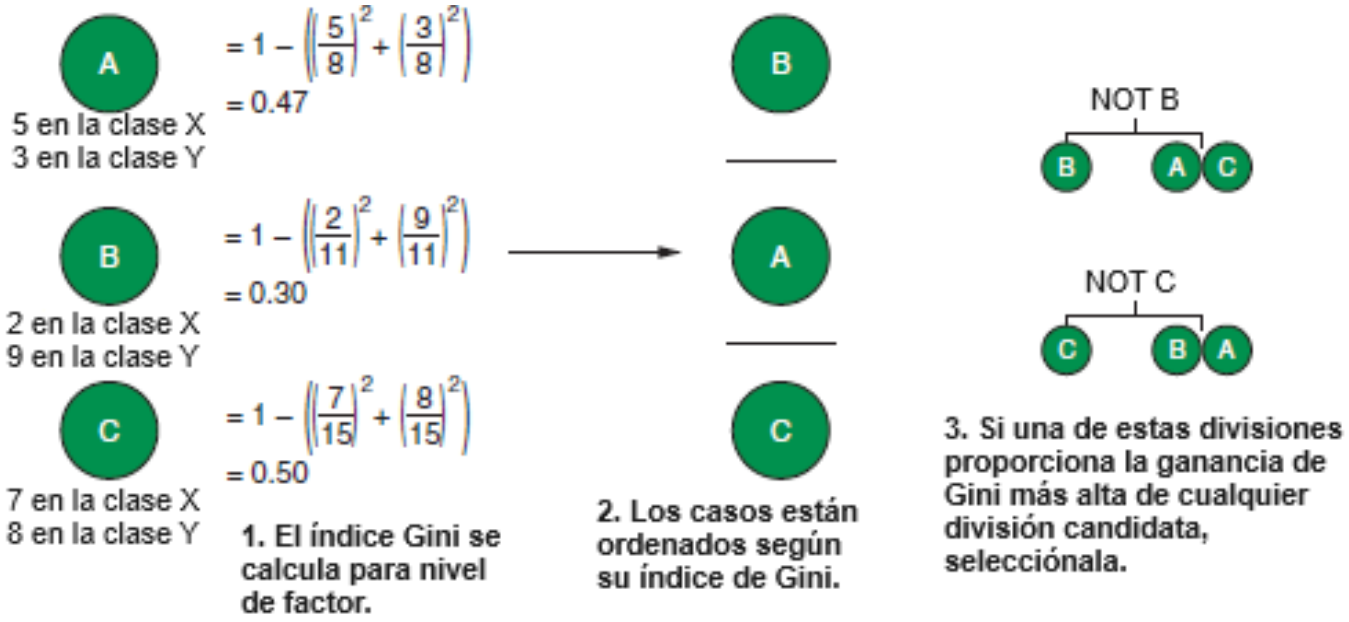
- Tenemos un factor con tres niveles (A, B y C): evaluamos el índice de Gini de cada uno y encontramos que sus valores son B < A < C. Ahora evaluamos la ganancia de Gini para los splits B versus A y C, y C versus B y A.
- De esta manera, podemos crear una división binaria a partir de variables categóricas con muchos predictores sin tener que probar todas las combinaciones posibles de divisiones de niveles (2^{m-1}, donde m es el número de niveles de la variable).
- Si se encuentra que la división B versus A y C tiene la mayor ganancia de Gini, entonces los casos que lleguen a este nodo bajarán por una rama si tienen un valor de B para esta variable, y bajarán por la otra rama si tienen un valor de A o C.
- Cómo se elige el punto de división para predictores continuos. Los casos (círculos) están ordenados según su valor del predictor continuo. El punto medio entre cada par de casos adyacentes se considera una división candidata y la ganancia de Gini se calcula para cada uno. Si una de estas divisiones tiene la ganancia de Gini más alta de cualquier división candidata, se utilizará para dividir el árbol en este nodo.

Predictores categóricos continuos y multinivel, V



- Cómo se elige el punto de división para los predictores categóricos. El índice de Gini de cada nivel de factor se calcula utilizando la proporción de casos de cada clase con ese nivel de factor.
- Los niveles de los factores están ordenados según sus índices de Gini y la ganancia de Gini se evalúa para cada división entre niveles adyacentes.

Predictores categóricos continuos y multinivel, VI



Hiperparámetros del algoritmo rpart, I

- En esta sección, te mostraremos qué hiperparámetros deben ajustarse para el algoritmo rpart, qué hacen y por qué debemos ajustarlos para obtener el árbol con el mejor rendimiento posible.
- Los algoritmos de árboles de decisión se describen como 'voraces' (*greedy*). Por voraces no nos referimos a que tomen una ración extra en la fila del buffet; Queremos decir que buscan la división que funcionará mejor en el nodo actual, en lugar de la que producirá el mejor resultado a nivel global.
- Por ejemplo, una división particular podría discriminar mejor las clases en el nodo actual, pero dar como resultado una separación deficiente más abajo en esa rama.
- Por el contrario, una división que da como resultado una separación deficiente en el nodo actual puede producir una mejor separación más abajo en el árbol.
- Los algoritmos de árboles de decisión nunca elegirían esta segunda división porque solo analizan las divisiones *óptimas localmente*, en lugar de las *óptimas globalmente*.

Hiperparámetros del algoritmo rpart, II

- Hay tres problemas con este enfoque:
  - No se garantiza que el algoritmo aprenda un modelo globalmente óptimo.
  - Si no se controla, el árbol seguirá creciendo más profundamente hasta que todas las hojas sean puras (de una sola clase).
  - Para conjuntos de datos grandes, cultivar árboles extremadamente profundos se vuelve computacionalmente costoso.
- Si bien es cierto que no se garantiza que rpart aprenda un modelo globalmente óptimo, la profundidad del árbol es lo que más nos preocupa. Es muy probable que el costo computacional, además de hacer crecer un árbol en toda su profundidad hasta que todas las hojas sean puras, sobreadapte el conjunto de entrenamiento y cree un modelo con una alta varianza.
- Esto se debe a que a medida que el espacio de características se divide en partes cada vez más pequeñas, es mucho más probable que comencemos a modelar el ruido en los datos. ¿Cómo nos protegemos de una construcción de árboles tan extravagante?

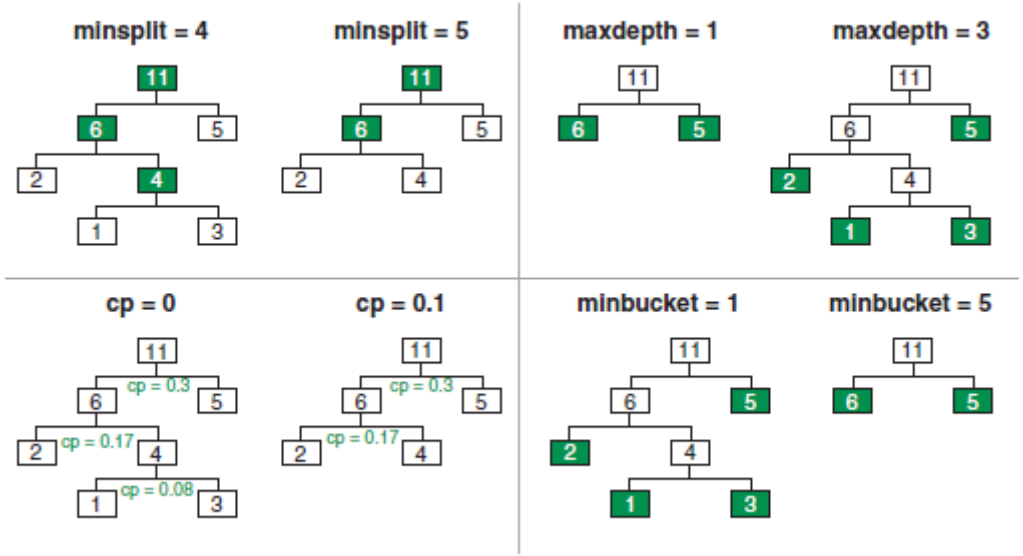
Hiperparámetros del algoritmo rpart, III

- Hay dos formas de hacerlo:
  - Haz crecer un árbol completo y luego pódalo.
  - Emplear *criterios de paro*.
- En el primer enfoque, permitimos que el algoritmo voraz haga crecer su árbol completo y sobreajustado, y luego sacamos nuestras tijeras de podar y eliminamos las hojas que no cumplen con ciertos criterios.
- A este proceso se le llama imaginativamente *poda*, porque terminamos quitando ramas y hojas de nuestro árbol. A esto a veces se le llama poda de abajo hacia arriba porque comenzamos desde las hojas y podamos hacia la raíz.
- En el segundo enfoque, incluimos condiciones durante la construcción del árbol que obligarán a detener la división si no se cumplen ciertos criterios. A esto a veces se le llama poda de arriba hacia abajo porque podamos el árbol a medida que crece desde la raíz.

Hiperparámetros del algoritmo rpart, IV

- Ambos enfoques pueden producir resultados comparables en la práctica, pero existe una ligera ventaja computacional en la poda de arriba hacia abajo porque no necesitamos cultivar árboles completos y luego podarlos nuevamente.
- Por esta razón, utilizaremos el enfoque de criterios de parada.
- Los criterios de parada que podemos aplicar en cada etapa del proceso de construcción del árbol son los siguientes:
  - Número mínimo de casos en un nodo antes de dividirse
  - Profundidad máxima del árbol
  - Mejora mínima en el rendimiento para una división
  - Número mínimo de casos en una hoja.
- Estos criterios se ilustran en la siguiente figura. Para cada división candidata durante la construcción del árbol, cada uno de estos criterios se evalúa y debe cumplirse para que el nodo se divida aún más.

Hiperparámetros del algoritmo rpart, V



- Hiperparámetros de rpart. Los nodos importantes se resaltan en cada ejemplo y los números en cada nodo representan el número de casos. Los hiperparámetros minsplit, maxdepth, cp y minbucket restringen simultáneamente la división de cada nodo.
- El número mínimo de casos necesarios para dividir un nodo se llama minsplit by rpart. Si un nodo tiene menos del número especificado, el nodo no se dividirá más. La profundidad máxima del árbol se llama profundidad máxima (maxdepth) por rpart.

Hiperparámetros del algoritmo rpart, VI

- Si un nodo ya se encuentra a esta profundidad, no se dividirá más. La mejora mínima en el desempeño no es, de manera confusa, la ganancia de Gini de una división.
- En cambio, se calcula una estadística llamada parámetro de complejidad (cp en rpart) para cada nivel de profundidad del árbol. Si el valor cp de una profundidad es menor que el valor umbral elegido, los nodos en este nivel no se dividirán más.
- En otras palabras, si agregar otra capa al árbol no mejora el rendimiento del modelo en cp, no dividas los nodos. El valor de cp se calcula como

$$cp = \frac{p(\text{incorrecto}_{l+1}) - p(\text{incorrecto}_{l})}{n(splits_{l}) - n(splits_{l+1})}$$

- donde *p*(incorrecto) es la proporción de casos clasificados incorrectamente en una profundidad particular del árbol, y *n*(splits, divisiones) es el número de divisiones en esa profundidad. Los índices l y $l + 1$ indican la profundidad actual (l) y una profundidad por encima ($l + 1$).

Hiperparámetros del algoritmo rpart, VII

- Esto se reduce a la diferencia en los casos clasificados incorrectamente en una profundidad en comparación con la profundidad por encima de ella, dividida por el número de nuevas divisiones agregadas al árbol.
- Si esto parece un poco abstracto en este momento, trabajaremos con un ejemplo cuando construyamos nuestro propio árbol de decisiones más adelante.
- Finalmente, el número mínimo de casos en una hoja se llama minbucket por rpart.
- Si dividir un nodo daría como resultado hojas que contienen menos casos que minbucket, el nodo no se dividirá.
- Estos cuatro criterios combinados pueden generar criterios de parada muy estrictos y complicados. Debido a que los valores de estos criterios no se pueden aprender directamente de los datos, son hiperparámetros.
- ¿Qué hacemos con los hiperparámetros? ¡Sintonízalos! Entonces, cuando construimos un modelo con rpart, ajustaremos estos criterios de parada para obtener valores que nos brinden el modelo con mejor rendimiento.

Hiperparámetros del algoritmo rpart, VIII

- *Nota*: Recuerda de los temas anteriores que una variable u opción que controla cómo aprende un algoritmo, pero que no se puede aprender a partir de los datos, se denomina hiperparámetro.