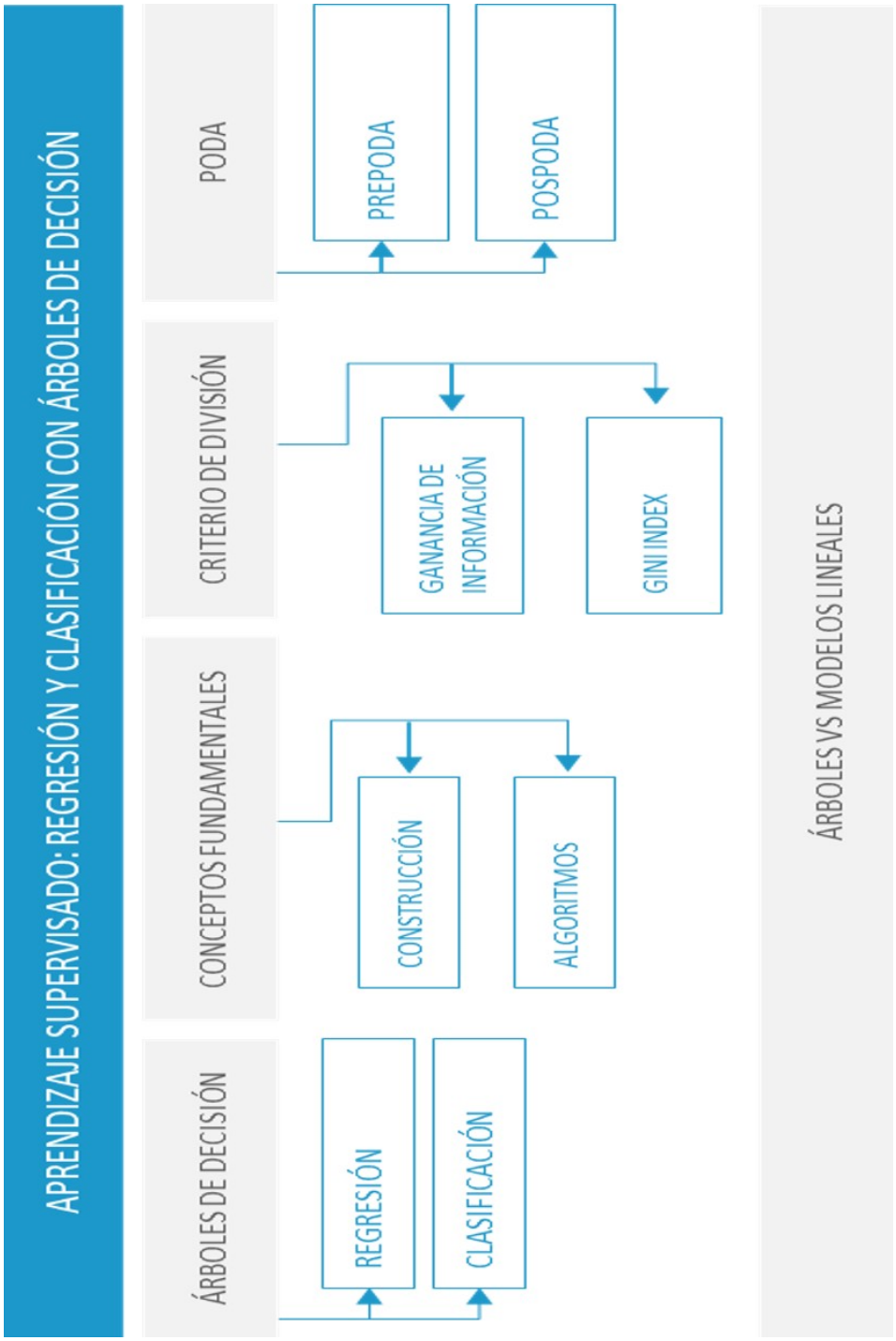
Técnicas de Aprendizaje Automático

Tema 6. Aprendizaje supervisado. Regresión y clasificación con árboles de decisión

Índice

Esquema



6.1. Introducción a los árboles de decisión

En este tema describimos los métodos basados en árboles para las tareas de regresión y clasificación. Estos modelos implican estratificar o segmentar el espacio de características en una serie de regiones simples. Para hacer una predicción para una observación determinada, normalmente se utilizará la media o la moda de las observaciones de entrenamiento en la región a la que pertenece. Dado que el conjunto de reglas de división utilizadas para segmentar el espacio de predictores se puede resumir en un árbol, estos tipos de enfoques se conocen como métodos de árbol de decisión. Los árboles de decisión son una de **las técnicas más populares dentro del campo del aprendizaje automático**. Se llevan utilizando durante muchos años en el área de la minería de datos. Se trata de **métodos simples y fáciles de interpretar**.

Como se ha indicado más arriba, los árboles de decisión son un tipo de algoritmo de aprendizaje automático que se utiliza tanto para problemas de clasificación como de regresión.

El proceso de construcción de un árbol de decisión se lleva a cabo de manera recursiva y se puede resumir en los siguientes pasos:

- **Selección de la característica de división:** seleccionar la característica y el punto de división que mejor separa las clases (o reduce la varianza en el caso de la regresión). Esto se hace evaluando diferentes criterios de división, como la ganancia de información, la reducción de la impureza de Gini o la reducción de la varianza.
- **División del nodo:** dividir el nodo en dos nodos hijos utilizando la característica y el punto de división seleccionados en el paso anterior.

- **Repetición del proceso:** repetir los pasos 1 y 2 para cada nodo hijo creado, de manera recursiva, hasta que se cumpla algún criterio de parada, como la profundidad máxima del árbol, el número mínimo de muestras en un nodo o la falta de ganancia de información significativa.

- **Poda del árbol (opcional):** después de construir el árbol, se puede realizar una poda para evitar el sobreajuste. La poda implica la eliminación de ciertas ramas del árbol que no contribuyen significativamente a mejorar su rendimiento en datos no vistos.

Los árboles de decisión son altamente interpretables y fáciles de visualizar, lo que los hace populares en una variedad de aplicaciones. Además, pueden manejar tanto variables numéricas como categóricas, como características de entrada, lo que los hace versátiles en diferentes tipos de problemas de aprendizaje automático.

El objetivo de este tema es obtener una mayor comprensión de los modelos de árboles de decisión. A continuación, se detallan los objetivos específicos:

- Describir qué son los árboles de decisión, cómo funcionan y cuándo se utilizan en problemas de aprendizaje automático.

- Identificar y diferenciar entre árboles de decisión para clasificación y regresión, así como comprender cómo se construyen y evalúan estos tipos de árboles.

- Comprender los diferentes criterios utilizados para dividir los nodos en árboles de decisión, como la ganancia de información, la impureza de Gini o la reducción de varianza, y cómo afectan la construcción del árbol.

- Evaluar la interpretabilidad de los árboles de decisión y su capacidad para generalizar a nuevos datos, así como explorar técnicas de poda para evitar el sobreajuste y mejorar la capacidad de generalización del modelo.

6.2. Conceptos fundamentales de los árboles de decisión

Un árbol de decisión es un clasificador expresado como una partición recursiva del espacio de instancias. El árbol de decisión, tal y como podemos ver en la Figura 1, consta de nodos que forman un árbol enraizado, lo que significa que es un árbol dirigido con un nodo llamado «raíz» que no tiene aristas entrantes. Todos los demás nodos tienen exactamente un borde de entrada. Un nodo con bordes salientes se denomina nodo interno o de prueba. Todos los demás nodos se denominan hojas (también conocidos como nodos terminales o de decisión). En un árbol de decisión, cada nodo interno divide el espacio de instancia en dos o más subespacios de acuerdo con una determinada función discreta de los valores de los atributos de entrada.

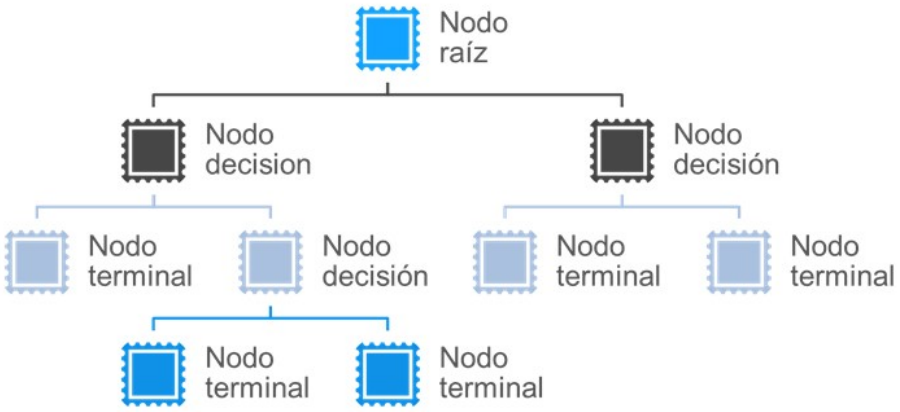


Figura 1. Representación de un árbol de decisión donde se indican cuál es el nodo raíz, cuáles son los nodos de decisión y cuáles son los nodos terminales. Fuente: elaboración propia.

En el caso más simple y frecuente, cada prueba considera un único atributo, de modo que el espacio de la instancia se divide según el valor del atributo. En el caso

de atributos numéricos, la condición hace referencia a un rango.

Cada hoja se asigna a una clase que representa el valor objetivo más apropiado. Alternativamente, la hoja puede contener un vector de probabilidad que indique la probabilidad de que el atributo objetivo tenga un valor determinado. Las instancias se clasifican navegando desde la raíz del árbol hasta una hoja, según el resultado de las pruebas a lo largo del camino.

Cada nodo está etiquetado con el atributo que prueba y sus ramas están etiquetadas con sus valores correspondientes. En la Figura 2 se puede ver un ejemplo de un árbol de decisión.

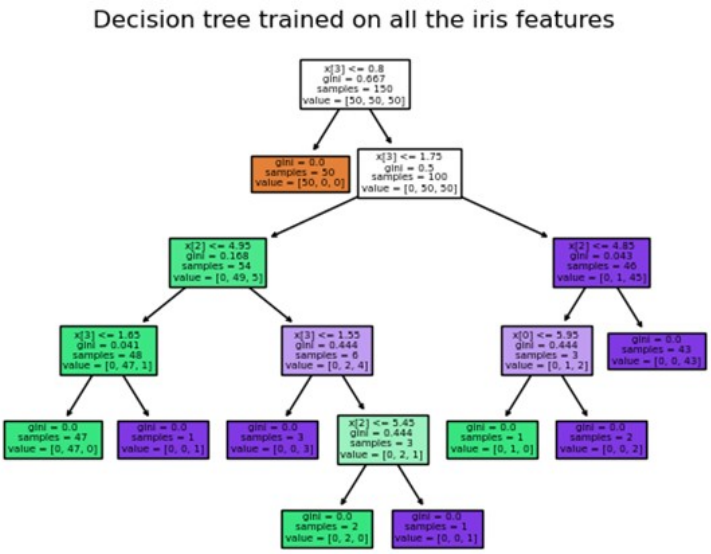


Figura 2. Representación gráfica de un árbol de decisión para la clasificación de especies de flores basado en características morfológicas. Fuente: scikit learn, s.f.

En esta representación del árbol de decisión, cada nodo del árbol representa una pregunta sobre una característica específica, mientras que las ramas indican las posibles respuestas y las hojas muestran las clases de flores resultantes. El árbol

proporciona una estructura visual intuitiva para comprender cómo se toman las decisiones de clasificación en el modelo.

En el caso de atributos numéricos, los árboles de decisión se pueden interpretar geométricamente como una colección de hiperplanos, cada uno ortogonal a uno de los ejes. Según Breiman *et al.* (1984), la complejidad del árbol tiene un efecto crucial en su precisión. La complejidad del árbol está controlada explícitamente por los criterios de parada utilizados y el método de poda empleado. Por lo general, la complejidad del árbol se mide mediante una de las siguientes métricas: el número total de nodos, el número total de hojas, la profundidad del árbol y el número de atributos utilizados. La inducción de árboles de decisión está estrechamente relacionada con la inducción de reglas. Cada camino desde la raíz de un árbol de decisión hasta una de sus hojas se puede transformar en una regla simplemente uniendo las pruebas a lo largo del camino para formar la parte antecedente y tomando la predicción de clase de la hoja como valor de clase.

Construcción de un árbol de decisión

El árbol de decisión se obtiene por medio de un algoritmo que elige primero aquella variable que es más predictiva de la variable objetivo. A continuación, los ejemplos de entrenamiento se dividen en grupos con distintos valores para las clases de esta primera variable. El algoritmo continúa dividiendo los nodos con la elección de la mejor variable en cada iteración hasta que se alcance el criterio de parada. En cada iteración el algoritmo elige aquella variable que mejor predice la variable objetivo, por tanto, se trata de un algoritmo de tipo *greedy*. El criterio de parada puede venir dado por algunas de estas situaciones:

- Todos, o casi todos, los ejemplos del nodo son de la misma clase.
- No existen variables para distinguir entre los ejemplos.
- El árbol ha alcanzado un tamaño predefinido.

Ideas clave

El principal desafío en la implementación de un árbol de decisión es identificar los atributos que consideramos como el nodo raíz en cada nivel. En el apartado 6.3 se profundizará sobre ello.

Algoritmos de árboles de decisión

Existen numerosas implementaciones de algoritmos para la construcción de árboles, las más ampliamente utilizadas en el ámbito del *machine learning* son: ID3, C4.5, C5.0 y CART. Cada algoritmo tiene sus propias características y variaciones, pero todos ellos tienen como objetivo construir árboles de decisión para modelar las relaciones entre las características de entrada y la variable objetivo.

- **ID3** (Iterative Dichotomiser 3): ID3 fue uno de los primeros algoritmos de árbol desarrollados por Ross Quinlan (1986). Está diseñado específicamente para tareas de clasificación y funciona dividiendo recursivamente los datos en función de atributos categóricos. ID3 selecciona el mejor atributo en cada nodo utilizando la ganancia de información como criterio de división, con el objetivo de maximizar la homogeneidad (o pureza) de los subconjuntos resultantes.

- **C4.5:** C4.5 es una extensión de ID3, también desarrollada por Ross Quinlan (1993). Eliminó la restricción de que las características deben ser categóricas al definir dinámicamente un atributo discreto (basado en variables numéricas) que divide el valor del atributo continuo en un conjunto discreto de intervalos. C4.5 convierte los árboles entrenados (es decir, la salida del algoritmo ID3) en conjuntos de reglas si-entonces . Luego se evalúa la precisión de cada regla para determinar el orden en el que deben aplicarse. La poda se realiza eliminando la condición previa de una regla si la precisión de la regla mejora sin ella.

- **C5.0:** es la última versión desarrollada por Quinlan bajo una licencia propietaria como sucesor comercial de C4.5. Utiliza menos memoria y crea conjuntos de reglas más pequeños que C4.5 y, al mismo tiempo, es más preciso.

- **CART** (Classification and Regression Trees): CART es un algoritmo de árbol desarrollado por Leo Breiman y otros. A diferencia de ID3, C4.5 y C5.0, que se utilizan principalmente para clasificación, CART admite tanto tareas de clasificación como de regresión. CART construye árboles binarios dividiendo recursivamente los datos en función de divisiones de características que maximizan la reducción de impureza (para clasificación) o varianza (para regresión) en cada nodo. Los árboles CART a menudo se podan después de la construcción para evitar el sobreajuste y mejorar la generalización.

Ventajas y desventajas de los árboles de decisión

Enumeramos a continuación algunas **ventajas** de los árboles de decisión, recogidas por Schmid, H. (2010) y Pedregosa *et al.* (2011):

- Son sencillos de entender e interpretar. Se pueden visualizar árboles.
- Requieren poca preparación de datos. Otras técnicas a menudo requieren la normalización de datos, es necesario crear variables ficticias y eliminar valores en blanco. Algunas combinaciones de árboles y algoritmos admiten valores faltantes.
- El costo de usar el árbol (es decir, predecir datos) es logarítmico en el número de puntos de datos utilizados para entrenar el árbol.
- Capaz de manejar datos tanto numéricos como categóricos.
- Capaz de manejar problemas de múltiples salidas.
- Es posible validar un modelo mediante pruebas estadísticas. Eso permite dar cuenta de la confiabilidad del modelo.
- Los árboles de decisión son un método de clasificación rápido tanto en términos de tiempo de entrenamiento como de tiempo de procesamiento.

Por otro lado, los árboles de decisión también presentan algunas **desventajas**:

Ideas clave

- Los árboles de decisión pueden crear árboles demasiado complejos que no generalizan bien los datos. En otras palabras, los árboles de decisión son propensos al sobreajuste.
- Los árboles de decisión pueden ser inestables porque pequeñas variaciones en los datos pueden generar un árbol completamente diferente. Esto puede hacer que los árboles de decisión sean menos estables en comparación con otros modelos.
- Se sabe que el problema de aprender un árbol de decisión óptimo es NP-completo en varios aspectos de optimización e incluso para conceptos simples. En consecuencia, los algoritmos prácticos de aprendizaje de árboles de decisiones se basan en algoritmos heurísticos como el algoritmo *greedy*, donde se toman decisiones localmente óptimas en cada nodo. Dichos algoritmos no pueden garantizar que se devuelva el árbol de decisión globalmente óptimo.
- Hay conceptos que son difíciles de aprender porque los árboles de decisión no los expresan fácilmente, como por ejemplo los problemas XOR, de paridad o de multiplexor.
- Los modelos de árboles de decisión crean árboles sesgados si algunas clases dominan. Por lo tanto, se recomienda equilibrar el conjunto de datos antes de ajustarlo al árbol de decisión.

A pesar de estas desventajas, las ventajas previamente enumeradas convierten a los árboles de decisión en una de las técnicas eficaces dentro del área del aprendizaje automático debido a su simplicidad, interpretabilidad y capacidad para manejar una variedad de tipos de datos y problemas.

6.3. El criterio de división

Cada una de las divisiones del árbol da como resultado dos grupos. Es decir, los datos resultantes de la división por la variable elegida y con el punto de corte determinado están en uno de los dos grupos. Cuando los segmentos de una división contienen valores de una única clase se considera que es una **división pura.**

- A la hora de identificar los criterios para realizar el *split* o división existen varias métricas de pureza. Un criterio de división utilizado frecuentemente es la **ganancia de información**. La ganancia de información es la diferencia entre la entropía del conjunto de datos en el nodo actual y la entropía en los dos subconjuntos inducidos por la prueba. La entropía de un conjunto de datos mide hasta qué punto los datos están dispersos en varias clases. Si un conjunto de datos es puro, es decir, si todos los elementos pertenecen a la misma clase, la entropía es 0. Si la mitad de los datos pertenecen a la clase A y la mitad de los datos a la clase B, la entropía es 1. Dicho de otra manera, un valor de entropía igual a 0 indica que la muestra es completamente homogénea, mientras que un valor de 1 indica desorden completo.

La **entropía** se define con la siguiente fórmula matemática (Shanon, 1948):

$$Entropy\,(S) = \sum_{i=1}^{c} -p_i \log_2 (p_i)$$

Donde p_i es la frecuencia relativa de la clase i en el conjunto de datos.

A continuación, se calcula la ganancia de información (*information gain* o IG) que es la diferencia entre la entropía en el segmento antes de hacer el corte (S1) y la partición resultante de hacer el corte (S2), es decir:

```
InfoGain(F)= Entropy(S_1 )- Entropy (S_2)
```

Después de un corte, los datos se pueden dividir en más de una partición, por tanto, se considera la entropía a lo largo de las N particiones, ponderando la entropía de cada partición con el número de instancias de esa partición:

$$InfoGain(F) = Entropy(S_1) - Entropy(S_2)$$

El número de árboles de decisión posibles para un conjunto de datos determinado suele ser demasiado grande para compararlos entre sí y encontrar el árbol más simple y preciso. Esta es la razón por la que la mayoría de los algoritmos de aprendizaje automático de árboles de decisión aplican la estrategia de búsqueda *greedy* basada en la ganancia de información (o un criterio similar). Esta estrategia encuentra rápidamente un árbol de decisión cuyo tamaño sea cercano (pero no necesariamente igual) al óptimo.

Otro criterio de división ampliamente utilizado es el que utiliza el índice de Gini, que mide la impureza de un conjunto de datos. El índice Gini es la suma de productos de todos los pares de probabilidades de clase.

$$Gini = 1 - \sum_{i=1}^{c} (p_i)^2$$

El índice de Gini calcula cuál es la probabilidad de que una característica específica se clasifique incorrectamente cuando se selecciona al azar. Si todos los elementos están vinculados con una sola clase, entonces se puede llamar puro. El mayor valor de homogeneidad se da cuando Gini es igual a 0. Por lo tanto, el índice Gini alcanza su máximo cuando todas las frecuencias de clases son iguales. Es cero si todos los elementos de datos pertenecen a la misma clase. El criterio de división basado en el índice Gini selecciona la prueba para la cual la suma de los índices Gini ponderados de los subconjuntos de datos es mínima.

Ideas clave

Se ha demostrado que el criterio utilizado para realizar las particiones afecta a la calidad del modelo de aprendizaje (Aning y Przybyła-Kasperek, 2022). Por lo tanto, se plantea la cuestión de cuál de las dos métricas elegir. La elección del criterio de división dependerá del tipo de problema (clasificación o regresión), la naturaleza de los datos y las características específicas del conjunto de datos. En general, no hay un criterio de división universalmente superior, y es recomendable probar varios criterios y evaluar el rendimiento del modelo en conjuntos de datos de prueba para determinar cuál es el más adecuado para una tarea dada.

6.4. Poda de los árboles

El proceso anterior de generación de cortes recursivos denominado *top-down greedy* puede generar buenas predicciones en el conjunto de entrenamiento, pero también sufre el **problema del sobreajuste**. Realizar múltiples particiones en el espacio de características puede ocasionar que el árbol resultante sea muy complejo y que se ajuste exactamente a los datos con los que ha sido entrenado. Sin embargo, un árbol más pequeño puede dar lugar a una menor **varianza** en el modelo y mejor interpretación con el coste de un pequeño sesgo. El proceso por el cual en los árboles de decisión se obtienen árboles de menor complejidad se llama **poda**.

La poda en los árboles de decisión es un proceso que busca evitar el sobreajuste y mejorar la capacidad de generalización del modelo. La poda consiste en eliminar ciertas ramas del árbol que no contribuyen significativamente a mejorar su rendimiento en datos no vistos, es decir, que no mejoran la precisión de las predicciones en el conjunto de test.

La poda puede llevarse a cabo de dos maneras:

- **Prepoda** (*Early stopping or pre-pruning*): durante la construcción del árbol se aplican criterios de detención temprana para evitar que el árbol crezca más allá de ciertos límites. Estos criterios pueden incluir restricciones en la profundidad máxima del árbol, el número mínimo de muestras requeridas en un nodo para continuar dividiéndolo, o la ganancia mínima de información necesaria para realizar una división.

- **Pospoda** (*post-pruning*): después de construir el árbol completo, se evalúa la contribución de cada nodo al rendimiento del modelo en un conjunto de datos de validación o prueba. Se identifican y eliminan las ramas que no mejoran la precisión del modelo en el conjunto de prueba, lo que puede implicar fusionar nodos o eliminar subárboles enteros.

Ahora bien, la cuestión es: ¿Cómo seleccionamos el subárbol? Una buena solución es utilizar aquel subárbol que proporcione un menor error de test en el conjunto de datos de prueba con validación cruzada (*cross validation*) o bien en el conjunto de validación.

La selección del árbol resultante de la poda en *machine learning* se basa en la evaluación del rendimiento del modelo en conjuntos de datos de validación y la elección del árbol que maximice la precisión y la capacidad de generalización sin comprometer en exceso la complejidad del modelo. Algunas estrategias habituales para buscar el árbol óptimo son:

- **Validación cruzada:** utilizar técnicas de validación cruzada para evaluar el rendimiento de diferentes árboles podados en conjuntos de datos de validación. Se selecciona el árbol que proporciona el mejor rendimiento promedio en varias divisiones del conjunto de datos.

- **Curva de validación:** graficar la precisión del modelo en el conjunto de validación en función de la complejidad del árbol (por ejemplo, el número de nodos o la profundidad del árbol) y seleccionar el punto donde se maximiza la precisión o se minimiza el error.

- **Tamaño del árbol:** seleccionar un árbol con un tamaño que maximice el rendimiento del modelo en el conjunto de validación sin comprometer en exceso la simplicidad del modelo. Esto puede implicar la elección de un árbol con una profundidad o un número máximo de nodos predefinidos.

- **Pruebas empíricas:** realizar pruebas empíricas utilizando diferentes árboles podados en conjuntos de datos de validación y seleccionar el árbol que proporciona el mejor rendimiento en términos de precisión y generalización en datos no vistos.

Por ejemplo, en la Figura 4 se muestra la obtención de un árbol sin podar. En la Figura 5 se muestra el error obtenido por este árbol en función de la profundidad para los conjuntos de entrenamiento (*train*), test y validación cruzada (*cross validation*). Se observa que el punto óptimo es utilizar tres niveles.

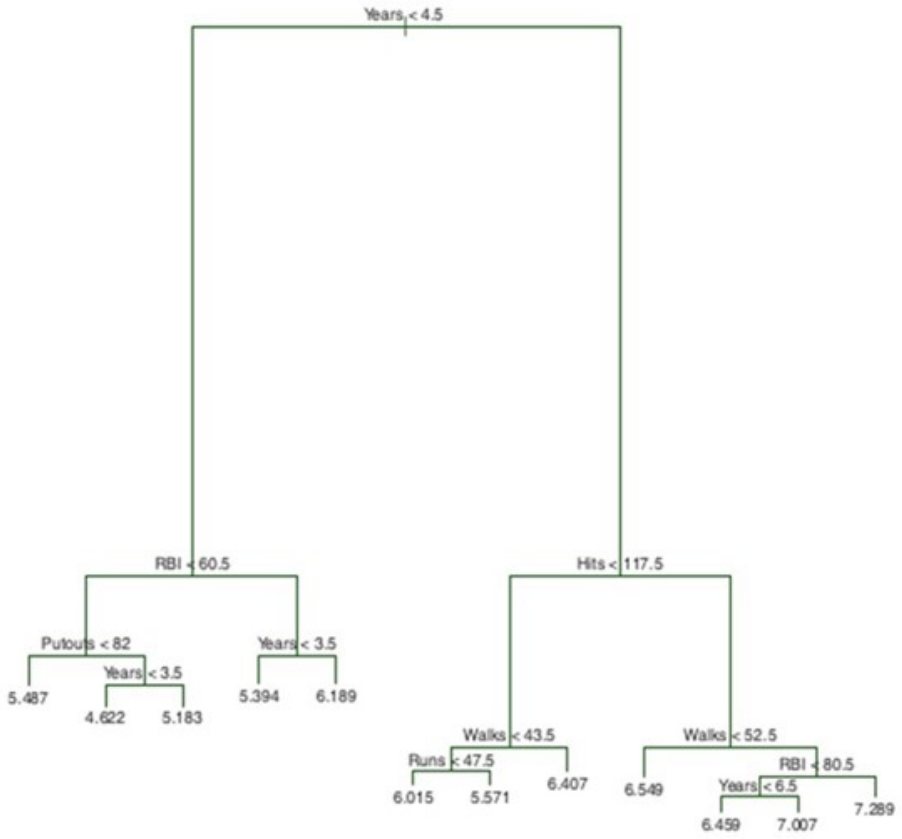


Figura 4. Ejemplo de un árbol sin podar con bastantes ramas. Fuente: James *et al.*, 2017.

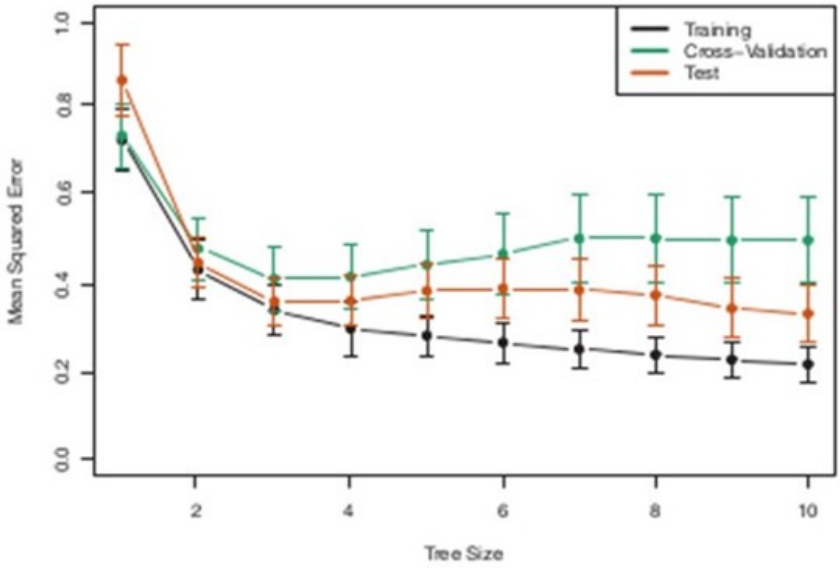


Figura 5. Ejemplo de la evolución del error del árbol de la Figura 4 en función del número de niveles y para 3 conjuntos de datos. Fuente: James *et al.*, 2017.

6.5. Árboles vs. modelos lineales

Para finalizar el tema, presentamos una comparativa entre los árboles de decisión y los modelos lineales.

Un **modelo lineal** en el contexto del aprendizaje automático es un tipo de modelo estadístico que asume una relación lineal entre las variables de entrada y la variable de salida. Con esta premisa, un modelo lineal intentará encontrar la mejor línea (o plano, en espacios de dimensiones más altas) que se ajuste a los datos para predecir la variable de salida en función de las características de entrada. Los modelos lineales son ampliamente utilizados en una variedad de aplicaciones de aprendizaje automático debido a su simplicidad, interpretabilidad y eficiencia computacional. Sin embargo, también tienen limitaciones en su capacidad para modelar relaciones no lineales entre las características y la variable objetivo, lo que puede requerir el uso de técnicas de transformación de características o modelos más complejos en ciertos escenarios.

Los árboles de decisión, por su parte, son modelos que imitan el proceso de toma de decisiones similar a un árbol, donde cada nodo representa una característica, cada rama representa una decisión basada en esa característica, y cada hoja representa el resultado final o la predicción.

Si comparamos ambos modelos, podemos decir que los árboles de decisión otorgan una flexibilidad mayor a la hora de resolver un problema, ya que pueden capturar relaciones no lineales en los datos sin requerir transformaciones complejas de las características. Sin embargo, los modelos lineales pueden no capturar relaciones no lineales en los datos tan bien como los árboles de decisión. Además, requieren que las características se transformen adecuadamente si la relación entre las características y la variable objetivo no es lineal, lo que puede requerir conocimiento previo del dominio o un análisis exploratorio exhaustivo.

Ideas clave

En cuanto a las **características**, los árboles de decisión pueden manejar tanto características numéricas como categóricas, y no requieren suposiciones sobre la distribución de los datos. Por su parte, los modelos lineales requieren que las características sean numéricas y además que exista una linealidad de la relación entre las características y la variable de salida, la independencia entre las características, la normalidad y homocedasticidad de los errores de predicción, y la linealidad en los parámetros del modelo.

Sin embargo, los árboles de decisión tienden a ser propensos al sobreajuste si se les permite crecer sin restricciones, lo que puede conducir a un rendimiento deficiente en datos no vistos. Además, pueden ser sensibles a pequeñas variaciones en los datos de entrenamiento, lo que puede llevar a modelos bastante diferentes con pequeños cambios en los datos. Por su parte, los modelos lineales son menos propensos al sobreajuste en comparación con los árboles de decisión, especialmente cuando se regularizan adecuadamente.

¿Qué modelo es mejor? Depende de lo que se esté **modelando**. Si las relaciones entre las variables de entrada y la variable respuesta se pueden aproximar por un modelo lineal, una regresión lineal funciona bien y supera a un árbol de regresión. Esto se debe a que un árbol de regresión no es capaz de modelar esta estructura. Sin embargo, si la naturaleza del problema es no lineal y con relaciones complejas entre las variables, los árboles han demostrado que funcionan mejor.

Los árboles de decisión son ideales cuando se busca **interpretabilidad y manejo flexible de características**, especialmente en problemas con relaciones no lineales en los datos. Por otro lado, los modelos lineales son eficientes y estables, lo que los hace adecuados para problemas donde se espera una relación lineal entre las características y la variable objetivo. La elección entre estos dos enfoques depende del contexto específico del problema y de las necesidades del proyecto.

6.6. Cuaderno de ejercicios

Ejercicio 1. Dado el conjunto de datos de la tabla (*Buy Computer data*), construye un árbol de decisión y predice la clase para el siguiente ejemplo: age = joven , income=medium , student=yes , credit-rating=fair .

id	Age	Income	Student	Credit_rating	Buy_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_age	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_age	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_age	medium	no	excellent	yes
13	middle_age	high	yes	fair	yes
14	senior	medium	no	excellent	no

SOLUCIÓN

Para construir el árbol de decisión tenemos que conocer cuáles son los atributos que proporcionan mayor ganancia de información en cada nivel. Esto nos servirá para dividir el conjunto de entrenamiento en base a dicho atributo. Por lo tanto, necesitamos calcular la ganancia de información esperada para clasificar el conjunto y la entropía de cada atributo. La ganancia de información esperada es esta información mutua menos la entropía:

The mutual information of the two classes I(SYes, SNo) = I(9,5)= -9/14 log2(9/14) – 5/14 log2(5/14)= 0.94

Ahora podemos calcular la ganancia de información de cada atributo. Tenemos que calcularlo para *age*, *income*, *student* y *credit_rating*.

- Age. Tenemos tres valores posibles age = youth (2 yes, 3 no)/middle_age(4 yes, 0 no)/senior(3 yes, 2 no) .

```
import math

# a * (b + c)

entropy_age_youth = (5/14) * (((-2/5) * math.log2(2/5)) + ((-3/5) *
math.log2(3/5)))

a = 5/14

b = (-2/5) * math.log2(2/5)

c = (-3/5) * math.log2(3/5)

entropy_age_youth = a * (b + c)

a = 5/14

b = (-3/5) * math.log2(3/5)

c = (-2/5) * math.log2(2/5)

entropy_age_middle = (4/14)*0

entropy_age_senior = a * (b + c)

entropy_age = entropy_age_youth + entropy_age_middle + entropy_age_senior

gain_age = 0.94 - entropy_age

print(gain_age)
```

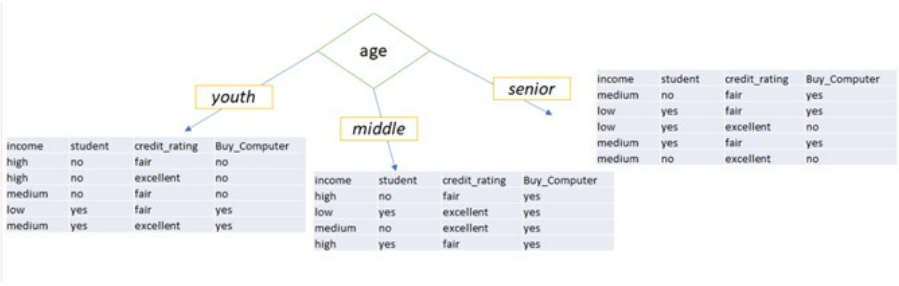
- Income. Tenemos tres valores posibles income = high (2 yes, 2 no)/medium(4 yes, 2 no)/low(3 yes, 1 no) . Se realizan los cálculos de manera similar a *age*, dando como resultado:

```
a = 4/14

b = (-2/4) * math.log2(2/4)

c = (-2/4) * math.log2(2/4)

entropy_high = a * (b + c)

a = 6/14

b = (-4/6) * math.log2(4/6)

c = (-2/6) * math.log2(2/6)

entropy_medium = a * (b + c)

a = 4/14

b = (-3/4) * math.log2(3/4)

c = (-1/4) * math.log2(1/4)

entropy_low = a * (b + c)

entropy_income = entropy_high + entropy_medium + entropy_low

gain_income = 0.94 - entropy_income

print(gain_income)
```

- Student. Tenemos dos valores posibles student = yes (6 yes, 1 no)/ no(3 yes, 4 no) . Se realizan los cálculos de manera similar a *age*, dando como resultado: gain_student = 0.1516.

- Credit_rating. Tenemos dos posibles valores cre dit_rating = fair (6 yes, 2 no) / excellent (3 yes / 3 no). Se realizan los cálculos de manera similar a los anteriores, dando como resultado: gain_credit_rating = 0.0479.

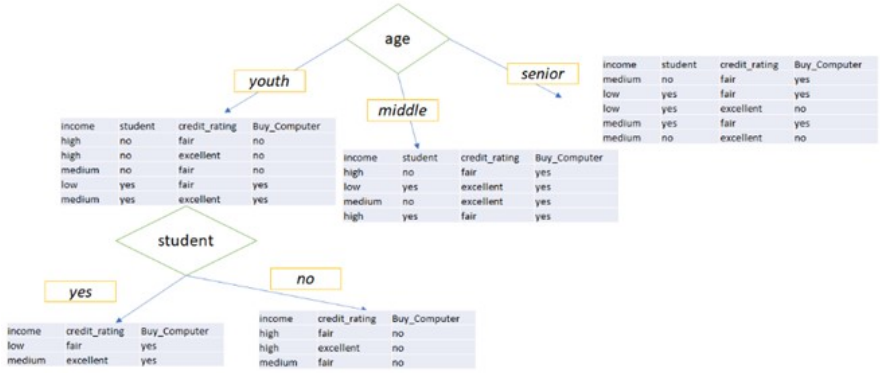
De todas las ganancias de información, la que tiene mayor valor es la de *age*; por lo tanto, ese será el atributo con el que se realizará la primera partición de datos, dando como resultado el siguiente árbol.



Si observamos este árbol, veremos que en la rama correspondiente a age = middle , todas las clases son *YES*; por lo tanto, podemos sustituir la hoja por class = YES .

En el resto de nodos debemos volver a particionar los datos. Se repite el mismo proceso de calcular la ganancia de información para cada atributo.

Para el nodo *youth* hay que calcular la ganancia de los atributos: *income*, *student* y *credit_rating*: I(Syes, Sno) = I(2,3) = -2/5log2(2/5) - 3/5log2(3/5) = 0.97 . Repetimos los cálculos de la sección anterior y obtenemos: gain_income = 0.57 gain_student = 0.97 , y no haría falta calcular la última ganancia de información porque ya hemos obtenido un valor máximo para *student*. Se utiliza dicho valor para particionar, y el resultado del particionado de esa rama sería:



Si observamos las particiones, vemos que cada nodo puede ser hoja, ya que las clases son similares. Por lo tanto, transformamos la partición student = yes en Class = YES y student = no en Class = NO .

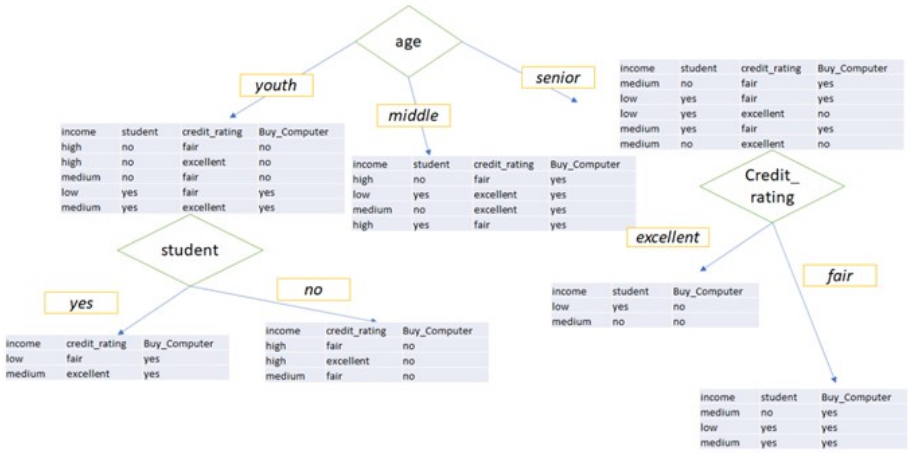
Finalmente, nos queda particionar el nodo age = senior . Seguimos el mismo proceso que antes.

```
I (Syes, Sno) = I (3, 2) = 0.97
```

Hay que calcular la ganancia de información de *income*, *student* y *credit_rating* en este caso. Es importante recordar , aunque sean los mismos atributos que en la rama anterior, los valores no tienen por qué coincidir porque las probabilidades cambian.

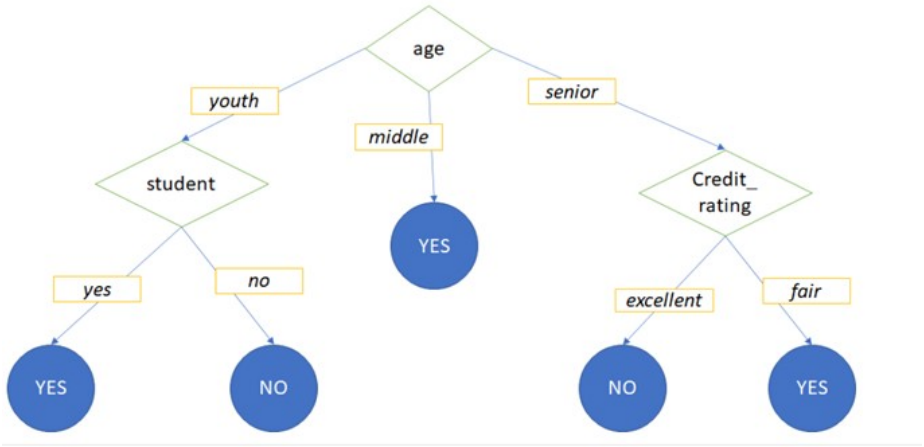
```
gain_income = 0.02 gain_student = 0.02 gain_credit_rating = 0.97
```

La máxima ganancia de información nos la da el atributo *credit_rating*. El particionado quedaría de la siguiente manera.



En este caso transformamos en hoja, credit_rating = excellent en class = NO y credit_rating = fair en class =YES .

El árbol de decisión resultante sería de la siguiente manera:



Ejercicio 2. Considera una tarea de clasificación de dos categorías con los siguientes datos de entrenamiento. Construye un árbol de decisión completo (sin podar) para estos datos usando la ganancia de información como criterio de división. Muestra todos los cálculos de entropía:

attr1	attr2	attr3	attr4	class
a	1	c	-1	c1
b	0	c	-1	c1
a	0	c	1	c1
b	1	c	1	c1
b	0	c	1	c2
a	0	a	-1	c2
a	1	a	-1	c2
b	1	c	-1	c2

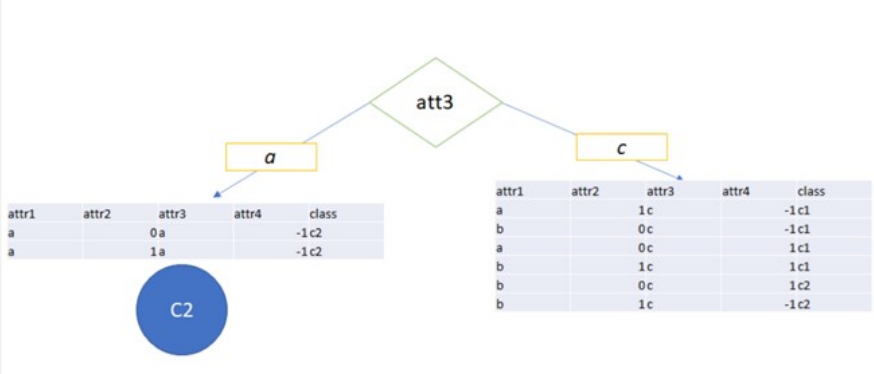
```
import math

I_class = - 4/8 * math.log2(4/8) - 4/8 * math.log2(4/8)

# dos posibles valores, a (c1 2 y c2 2) y b (c1 2 y c2 2)

entropy_att1 = (4/8 * ((-2/4) * math.log2(2/4)- (2/4)* math.log2(2/4))) +
(4/8 * ((-2/4) * math.log2(2/4)- (2/4)* math.log2(2/4)))

gain_att1 = I_class - entropy_att1

print(gain_att1)

# dos posibles valores, 1 (c1 2 y c2 2) y 0 (c1 2 y c2 2)

entropy_att2 = (4/8 * ((-2/4) * math.log2(2/4)- (2/4)* math.log2(2/4))) +
(4/8 * ((-2/4) * math.log2(2/4)- (2/4)* math.log2(2/4)))

gain_att2 = I_class - entropy_att2

print(gain_att2)
```

dos posibles valores, c (c1 4 y c2 2) y a (c1 0 y c2 2)

```
entropy_att3 = (6/8 * ((-4/6) * math.log2(4/6)- (2/6)* math.log2(2/6))) +
(2/8 * ((-0/2) * 0- (2/42)* math.log2(2/2)))
```

```
gain_att3 = I_class - entropy_att3

print(gain_att3)

# dos posibles valores, -1 (c1 2 y c2 3) y 1 (c1 2 y c2 1)

entropy_att4 = (5/8 * ((-2/5) * math.log2(2/5)- (3/5)* math.log2(3/5))) +
(3/8 * ((-2/3) * math.log2(2/3)- (1/3)* math.log2(1/3)))

gain_att4 = I_class - entropy_att4

print(gain_att4)

0.04879494069539847
```

La mayor ganancia de información en el primer nivel es att3.



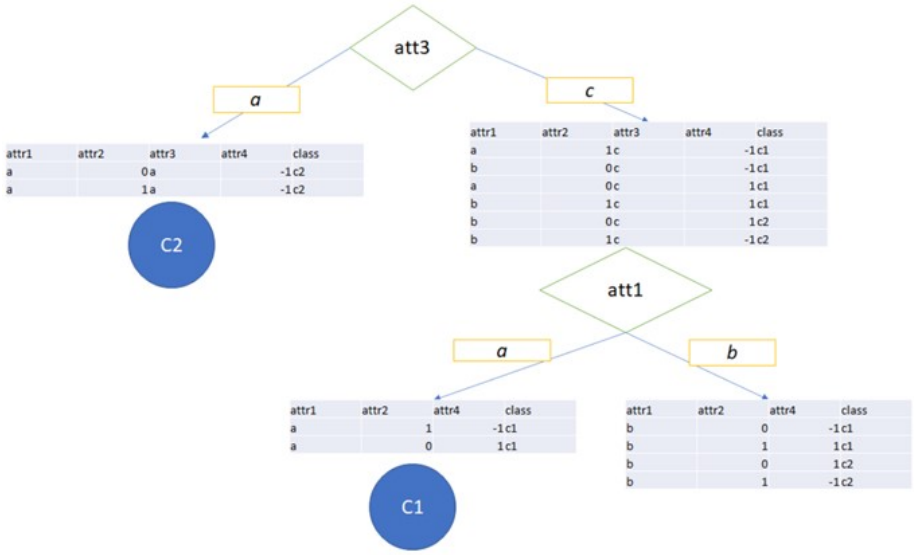
La rama de la partición con att3 = a se convierte en hoja class = C2. Seguimos particionando por la rama att3 = 3 ; hay que calcular la ganancia de información de los siguientes atributos *gain_att1*, *gain_att2*, *gain_att4*

```
I_class = (- (4/6) * math.log2(4/6)) - (2/6 * math.log2(2/6))

# dos posibles valores, a (c1 2 y c2 0) y b (c1 2 y c2 2)

entropy_att1 = (2/6 * ((-2/2) * math.log2(2/2)- 0)) + (4/6 * ((-2/4) *
math.log2(2/4)- (2/4)* math.log2(2/4)))

gain_att1 = I_class - entropy_att1
```

```
print(gain_att1)

# dos posibles valores, 1 (c1 2 y c2 1) y 0 (c1 2 y c2 0)

entropy_att2 = (34/5 * ((-2/3) * math.log2(2/3)- (1/3)* math.log2(1/3))) +
(2/5 * ((-2/2) * math.log2(2/2)- 0))

gain_att2 = I_class - entropy_att2

print(gain_att2)

# dos posibles valores, -1 (c1 2 y c2 1) y 1 (c1 2 y c2 1)

entropy_att4 = (3/6 * ((-2/3) * math.log2(2/3)- (1/3)* math.log2(1/3))) +
(3/6 * ((-2/3) * math.log2(2/3)- (1/3)* math.log2(1/3)))

gain_att4 = I_class - entropy_att4

print(gain_att4)
```

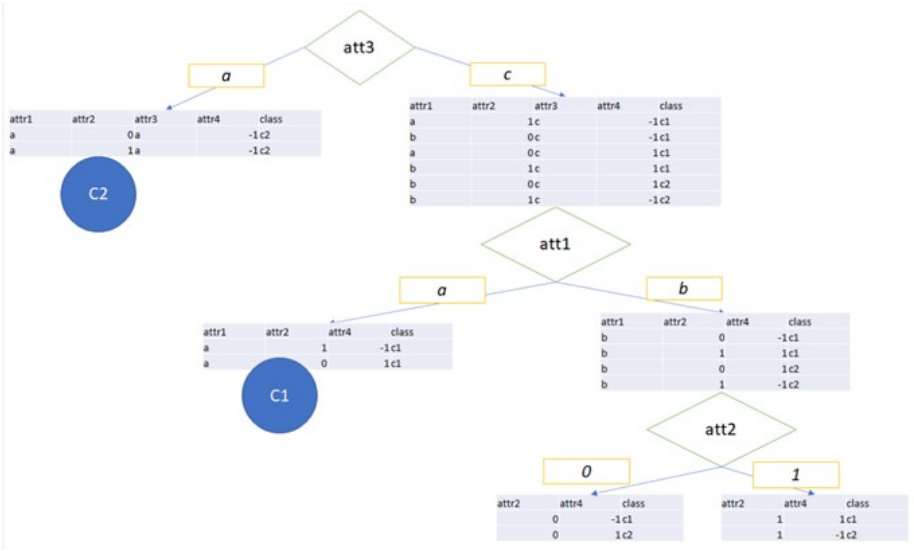
La máxima ganancia de información la da el atributo att1.

La partición quedaría de la siguiente manera:



La partición con el atributo att1= a da como resultado una hoja, class = C1 . Quedaría seguir particionando por la rama de att1= b .

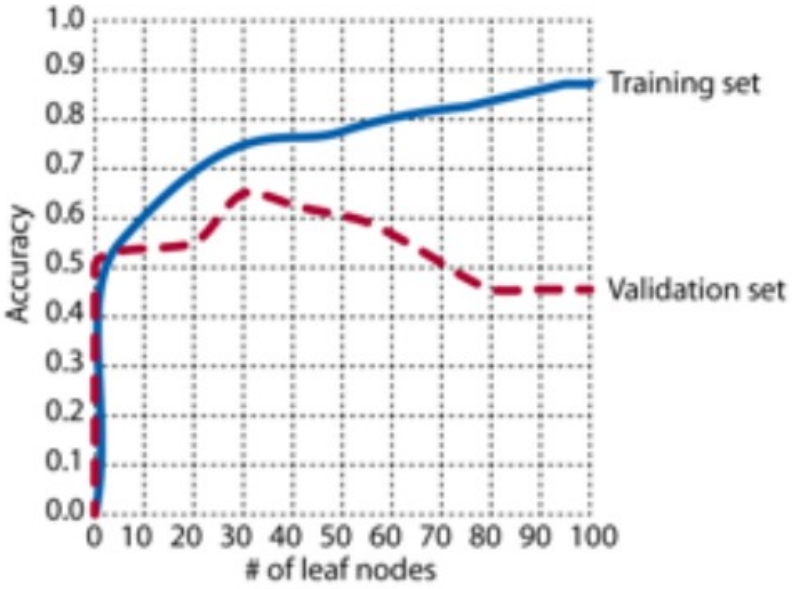
Hay que calcular la ganancia de información de los siguientes atributos *gain_att2*, *gain_att4*. En estos dos casos no haría falta hacer cálculos porque vemos que la representación de instancias en ambos casos es de 1 de cada variable y combinación, sabemos que cada atributo nos proporciona 0 de ganancia de información. Esto significa que no hay uno mejor que otro. Entonces, nuestro criterio de implementación será seleccionar el siguiente atributo por orden de presentación, es decir, *att2*. Dando como resultado el siguiente árbol.



En este árbol vemos que la siguiente partición tendrá que ser por *att4*, dando como resultado al árbol final.

Ejercicio 3. Los árboles de decisión tienden a sobreajustarse al conjunto de datos de entrenamiento si no se hace ninguna modificación durante o después del proceso de construcción del árbol. Una forma sencilla de evitar el sobreajuste es utilizar un conjunto de pruebas de validación. Un enfoque de reducción de errores es encontrar el número óptimo de nodos de hoja verificando las puntuaciones de precisión en los conjuntos de pruebas de capacitación y validación. ¿Cuál sería el número óptimo de

nodos de hojas cuando los valores de precisión aparecen como en el siguiente gráfico?



SOLUCIÓN

El número óptimo de nodos hoja sería el punto donde la precisión del conjunto de validación comienza a disminuir o a estabilizarse, mientras que la precisión del conjunto de entrenamiento continúa aumentando. Esto indica un equilibrio entre la complejidad del modelo y la generalización.

Alrededor del valor 30 (número de hojas), el gráfico muestra que la precisión del conjunto de entrenamiento continúa aumentando mientras que la precisión del conjunto de validación disminuye o se estabiliza, esto sugiere que el modelo se está sobreajustando. En este caso, el número óptimo de nodos hoja probablemente sería aquel donde la precisión del conjunto de validación es mayor antes de que comience a disminuir.

Ejercicio 4. ¿Cuál es la principal ventaja de la pospoda frente a la prepoda?

SOLUCIÓN

La principal ventaja de la pospoda sobre la prepoda es que la pospoda permite que el árbol de decisiones crezca hasta alcanzar su máximo potencial, capturando potencialmente relaciones más complejas en los datos. La prepoda, por otro lado, implica detener el crecimiento del árbol temprano en función de ciertos criterios, como una profundidad máxima o un número mínimo de muestras necesarias para dividir un nodo.

Al permitir que el árbol crezca completamente antes de la poda, la pospoda puede adaptarse mejor a la complejidad del conjunto de datos, lo que podría dar como resultado un modelo más preciso y generalizado. Además, la pospoda tiende a ser computacionalmente menos costosa en comparación con la prepoda, ya que no requiere evaluar múltiples divisiones potenciales en cada nodo durante la fase de construcción del árbol.

Ejercicio 5. Haciendo uso del dataset del Titanic (https://www.kaggle.com/c/titanic), construye un clasificador que realice predicciones sobre la supervivencia de un pasajero. Utiliza las métricas de clasificación aprendidas en el Tema 5 para evaluar el desempeño de tu clasificador.

SOLUCIÓN

No existe una solución única al problema.

6.7. Referencias bibliográficas

Aning, S. y Przybyła-Kasperek, M. (2022). Comparative Study of Twoing and Entropy Criterion for Decision Tree Classification of Dispersed Data. *Procedia Computer Science, 207*, 2434-2443.

Breimann, L., Friedman, J. H., Olshen, R. A. y Stone, C. J. (1984). *Classification and regression trees*. Pacific Grove, Wadsworth.

James, G., Witten, D., Hastie, T. y Tibshirani, R. (2017). *An Introduction to Statistical Learning with Applications*. R. Springer.

Quinclan, J. (1986). *Introduction to decision trees*. Springer.

Quinlan, J. R. (1986). *Induction of Decision Trees. Machine Learning, 1*, 81–106. Kluwer Academic Publishers.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers.

Schmid, H. (2010). Decision trees. En A. Clark, C., Fox y S. Lappin (eds.), *The Handbook of Computational Linguistics and Natural Language Processing* (180-196). Wiley-Blackwell.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal, 27*(3), 379-423.

Explicación visual funcionamiento de los árboles de decisión

R2D3. (s. f.). *A visual introduction to machine learning*. http://www.r2d3.us/visual-intro-to-machine-learning-part-1/

En este artículo de blog encontrarás una explicación visual sobre el funcionamiento de los árboles de decisión con el objetivo de clasificar los alquileres en San Francisco frente a Nueva York.

Decision Tree en Python utilizando Sklearn

Scikit learn. *Documentation. Decision Trees*. http://scikit-learn.org/stable/modules/tree.html

Documentación de la librería scikit learn sobre su implementación de los árboles de decisión.

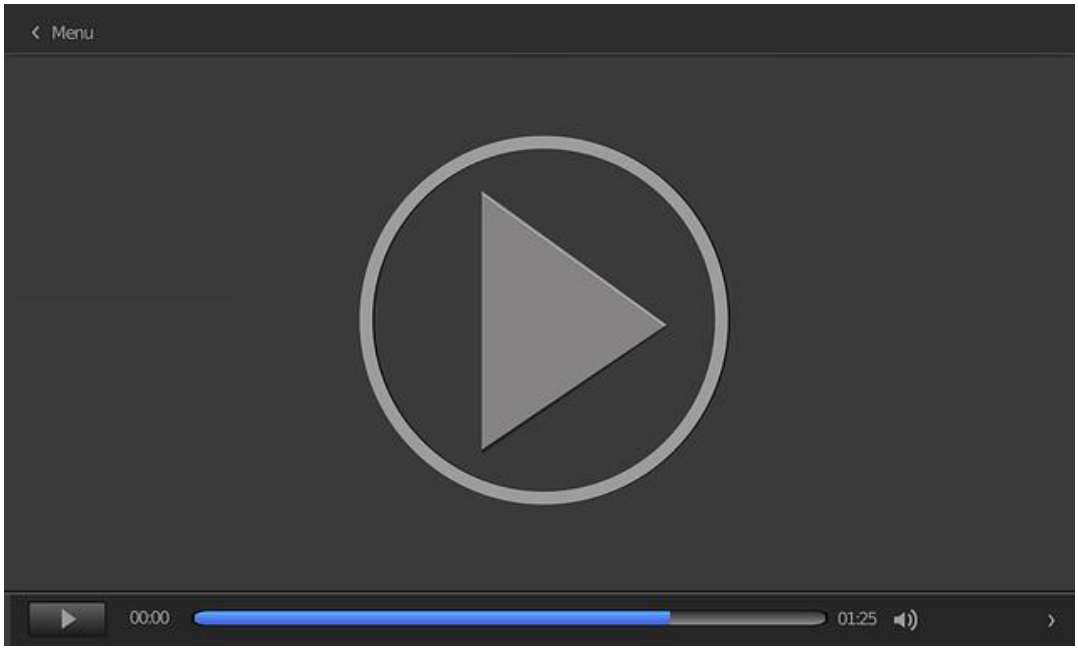
Introducción a los árboles de decisión para Regresión y Clasificación

Lewis, Roger. (2000). *An Introduction to Classification and Regression Tree (CART) Analysis*.
https://www.researchgate.net/publication/240719582_An_Introduction_to_Classification_and_Regression_Tree_CART_Analysis

Artículo científico que presenta, haciendo uso de un caso de uso médico, los árboles de decisión como algoritmos para utilizar en técnicas de clasificación y regresión. Este artículo, de fácil lectura, sintetiza y explica de manera sencilla los conceptos fundamentales de los árboles de decisión aplicados a las tareas de aprendizaje automático.

Poda en los árboles de decisión

Sebastian Mantey. (2020, marzo 11). *Decision Tree Pruning explained (Pre-Pruning and Post-Pruning)* [Vídeo]. YouTube. https://www.youtube.com/watch?v=u4kbPtiVVB8



Accede al vídeo:

https://www.youtube.com/embed/u4kbPtiVVB8

En este vídeo se encuentra una explicación de las técnicas de poda (prepoda y pospoda) para los árboles de decisión. La explicación está acompañada de una visualización que facilita la comprensión.

1. Marca la afirmación falsa sobre los árboles de decisión:

   A. Los árboles de decisión son un método de aprendizaje supervisado que se basa en la estructura de un árbol para tomar decisiones a partir de datos, como en clasificación y regresión.

   B. La construcción de un árbol de decisión implica dividir el conjunto de datos en subconjuntos más pequeños, utilizando diferentes criterios, como la entropía o la ganancia de información.

   C. Los árboles de decisión son fáciles de interpretar y visualizar, lo que los hace útiles para comprender cómo se toman las decisiones en un modelo predictivo.

   D. Los árboles de decisión siempre producen modelos que son robustos frente al sobreajuste, lo que significa que nunca es necesario preocuparse por la generalización del modelo.

2. En un árbol de decisión, ¿de qué es responsable el «nodo raíz»?

   A. Hacer predicciones.

   B. Dividir los datos en subconjuntos según una característica.

   C. Poda del árbol.

   D. Almacenamiento de los valores objetivo reales.

3. Los árboles de decisión:

   A. Transforman las variables de entrada utilizando funciones kernel.

   B. Dividen el espacio de las variables en una serie de regiones.

   C. Ejecutan diversas operaciones modificando las variables de entrada.

   D. Siempre producen modelos que son óptimos en términos de precisión y generalización, independientemente del tamaño y la complejidad del conjunto de datos.

4. ¿Cuáles de las siguientes afirmaciones son verdaderas sobre los árboles de decisión?

A. Solo se pueden utilizar para clasificación.

B. Todas las hojas deben ser puras.

C. La poda generalmente logra una mayor precisión en los datos de test que introducir un criterio de parada.

D. Todas las anteriores son correctas.

5. Un árbol de decisión se obtiene:

A. Por medio de un algoritmo que elige primero una variable aleatoria.

B. Por medio de un algoritmo que elige primero aquella variable más predictiva.

C. Por medio de un algoritmo que elige primero aquella variable menos predictiva.

D. Por medio de una elección aleatoria de las variables más predictivas.

6. El criterio de parada de la construcción de un árbol de decisión viene dado por:

A. Todos, o casi todos, los ejemplos del nodo son de la misma clase.

B. No existen variables para distinguir entre los ejemplos.

C. El árbol ha alcanzado un tamaño predefinido.

D. Todas las anteriores son correctas.

7. ¿Cuál de las siguientes afirmaciones describe incorrectamente el índice Gini en árboles de decisión?

A. El índice Gini mide la impureza de un conjunto de datos, donde un valor más alto indica una mayor impureza y una mezcla más desigual de clases.

B. El índice Gini se utiliza para seleccionar la mejor división en un nodo de un árbol de decisión, buscando minimizar la impureza en los nodos hijos.

C. El índice Gini asigna un valor de 0 cuando todos los ejemplos en un nodo pertenecen a la misma clase, lo que indica una pureza máxima.

D. El índice Gini es una medida de la ganancia de información, utilizada para determinar qué característica proporciona la mayor separación entre las clases objetivo en un conjunto de datos.

8. ¿Cuál de las siguientes afirmaciones describe correctamente la poda de árboles de decisión?

A. La poda en árboles de decisión implica agregar más nodos al árbol para mejorar su capacidad de generalización y precisión en la predicción.

B. La poda se realiza eliminando ramas del árbol que no contribuyen significativamente a mejorar su rendimiento en el conjunto de datos de prueba.

C. La poda de árboles de decisión es un proceso que se lleva a cabo durante la etapa de construcción del árbol para mejorar su capacidad de ajuste a los datos de entrenamiento.

D. La poda de árboles de decisión solo se aplica en casos donde el árbol ha alcanzado un tamaño excesivo y se necesita reducir su complejidad para evitar el sobreajuste.

9. ¿Cuál de las siguientes opciones describe correctamente la ganancia de información en árboles de decisión?

A. La ganancia de información mide la pureza de un nodo en un árbol de decisión, donde un valor más alto indica una mayor homogeneidad en las clases.

B. La ganancia de información se utiliza para seleccionar la característica que divide un conjunto de datos de la manera más equitativa posible, sin favorecer una clase sobre otra.

C. La ganancia de información se calcula como la diferencia entre la entropía del nodo padre y la suma ponderada de las entropías de los nodos hijos después de una división.

D. La ganancia de información asigna un valor de 0 cuando todas las muestras en un nodo pertenecen a la misma clase, lo que indica una falta de variabilidad en las clases objetivo.

10. ¿Cuál de las siguientes afirmaciones describe incorrectamente los árboles de decisión?

A. Los árboles de decisión son un tipo de modelo de aprendizaje no supervisado.

B. Los árboles de decisión pueden manejar tanto variables numéricas como categóricas, como características de entrada.

C. Los árboles de decisión dividen el espacio de características en regiones que representan decisiones basadas en reglas if-else .

D. Los árboles de decisión son propensos al sobreajuste cuando se construyen con poca profundidad y se les permite crecer sin restricciones.