Árboles de Decisión
Combinación de Clasificadores

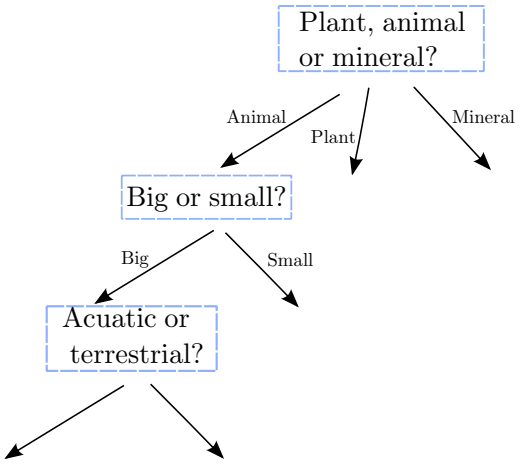
Aprenentatge Automàtic

APA/GEI/FIB/UPC - 2025/2026 1Q

/ Javier Béjar

Introducción

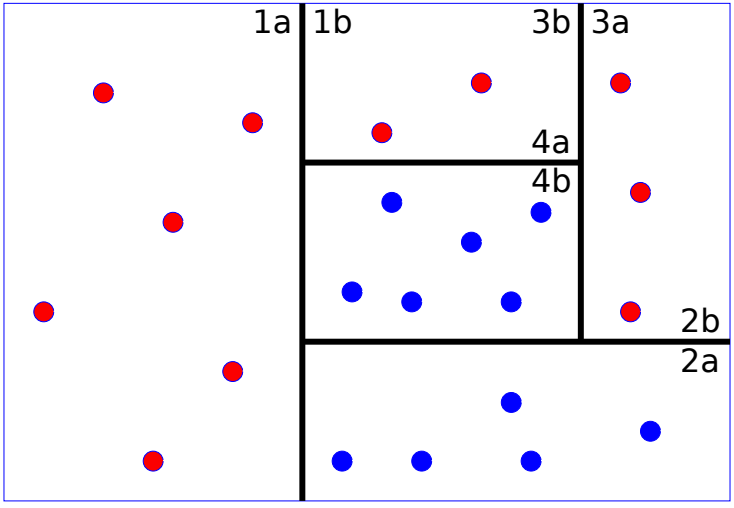
- Podemos abordar el aprendizaje de un concepto como el algoritmo para encontrar el conjunto de preguntas necesario para distinguirlo de otros
- El árbol de preguntas es el lenguaje de representación, cada nodo del árbol es un test sobre un atributo
- Es equivalente a una DNF (2^{2^n} conceptos)
- Para aprender tenemos que realizar una búsqueda en el espacio de árboles de preguntas



- Buscar en el espacio de todas las DNF es demasiado costoso
- Para reducir el costo computacional imponemos un **sesgo** (conceptos preferidos)
- **Restricción:** Queremos el árbol que representa la descripción mínima del concepto objetivo dado un conjunto de ejemplos
- **Justificación:** Este árbol será el mejor para clasificar nuevos ejemplos (se reduce la probabilidad de que tenga preguntas innecesarias)
- **La navaja de Occam**: "*La pluralidad nunca debe postularse sin necesidad*"

Algoritmos de árboles de decisión

- Los algoritmos de árboles de decisión realizan una búsqueda Hill-Climbing en el espacio de los árboles
- Para cada nueva pregunta
  1. Se elige un atributo según un criterio
  2. Se dividen y reparten los ejemplos de acuerdo a sus valores para ese atributo
  3. Se repite recursivamente con cada partición hasta que todos los ejemplos son de la misma clase
- La selección del atributo se decide mediante una función heurística que sesga la selección hacia el árbol mínimo


1a
1b
3b
3a
4a
4b
2b
2a

- Se pueden usar diferentes heurísticas para seleccionar el mejor atributo dado un conjunto de ellos, por ejemplo
  - Ganancia de información (basada en entropía)
  - Índice Gini (Error cuadrático)
- El objetivo básico de estas heurísticas es elegir el atributo más relevante para predecir las clases acorde a la distribución de los datos

- La teoría de la información estudia (entre otras cosas) cómo codificar mensajes y el costo de su transmisión
- Dado un conjunto de mensajes $M = \{m_1, m_2, \ldots, m_N\}$, cada uno con probabilidad $p(m_i)$, podemos definir la cantidad de información (H, entropía) contenida en los mensajes M como:

$$H(M) = \sum_{n=1}^{n} -p(m_n) \log(p(m_n))$$

- Este valor puede interpretarse como la información necesaria para distinguir entre los mensajes de M (número de bits necesarios para codificarlos)

- Podemos establecer una analogía entre el aprendizaje y la codificación de mensajes
  - Las clases son los mensajes
  - La proporción de ejemplos de cada clase son sus probabilidades
- Aprender de un árbol de decisión puede verse como la búsqueda de la mínima codificación que permite distinguir entre clases
- Para cada decisión adicional en el árbol, se evalúa cada atributo para decidir si se incluye en el código (si minimiza el código)

- Dado un conjunto de ejemplos $\mathcal{X}$ y la probabilidad de las clases $p(\mathcal{C}_i)$, su entropía es

$$H(\mathcal{X}) = \sum_{i=1}^{C} -p(\mathcal{C}_i) \log p(\mathcal{C}_i)$$

- Dado un atributo A, siendo $p(A = v_a)$ la probabilidad de tener el valor v_a en el atributo, y con $p(\mathcal{C}_i|A = v_a)$ la probabilidad de ser de clase $\mathcal{C}_i$ teniendo el valor v_i en el atributo, la entropía condicional con respecto a A es

$$H(\mathcal{X}|A) = \sum_{\forall v_a \in A} p(A = v_a) \sum_{i=1}^{C} -p(\mathcal{C}_i|A = v_a) \log p(\mathcal{C}_i|A = v_a)$$

- La ganancia de información de un atributo A es

$$IG(\mathcal{X}|A) = H(\mathcal{X}) - H(\mathcal{X}|A)$$

- Para minimizar el árbol debemos elegir el atributo con la mayor ganancia de información
- Esto significa elegir el atributo que minimiza el número de atributos adicionales necesarios para separar los ejemplos en sus clases
- Una vez que hemos elegido un atributo en una rama del árbol, el atributo se puede descartar

- El índice de Gini es una heurística alternativa
- Se define como:

$$Gini(\mathcal{X}) = \sum_{i=1}^{C} p(\mathcal{C}_i)(1 - p(\mathcal{C}_i)) = 1 - \sum_{i=1}^{C} p(\mathcal{C}_i)^2$$

- Podemos calcular el índice de Gini de un atributo como la suma ponderada del índice de Gini de las particiones que generan
- Este índice mide la dispersión, en este caso buscamos el atributo que la minimiza
- Cuando es mínima todas las particiones tienen solo ejemplos de una clase

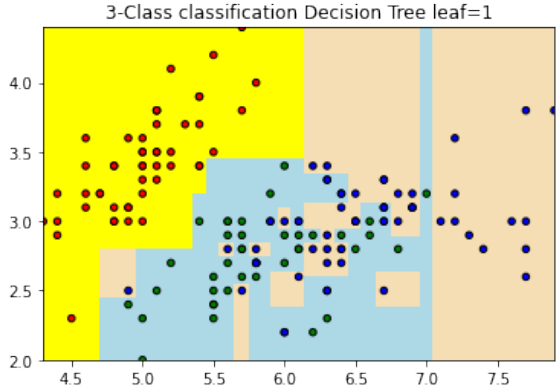
- La ganancia de información y el índice de Gini tienen preferencia por los atributos con mayor número de valores (son sesgados)
- En implementaciones prácticas, los atributos categóricos se transforman a binarios usando one hot encoding
- Esto hace que los árboles de decisión sean siempre árboles binarios en la práctica

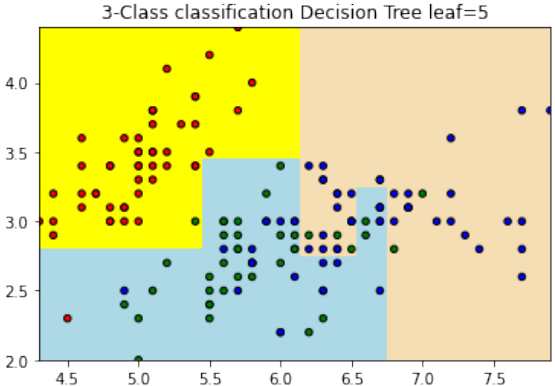
- Estas funciones heurísticas, tal como se definen, solo funcionan para atributos categóricos
- Por lo general, los conjuntos de datos también se describen mediante atributos numéricos
- No es factible tratar estos atributos como categóricos (cada valor como un valor diferente)
- Solución: Elegir la mejor división del rango de valores de un atributo en dos intervalos
- En la práctica esto significa discretizar el atributo (dinámicamente)

- Comenzamos con la secuencia ordenada de valores $A_i = \{v_1, v_2, \ldots, v_n\}$
- La ganancia de información se mide para cada partición de la secuencia ($n-1$ particiones binarias)
- La partición que maximiza la ganancia de información se compara con el resto de los atributos
- La decisión utilizada para el nodo del árbol es $A < \frac{v_i+v_{i+1}}{2}$ y $A \geq \frac{v_i+v_{i+1}}{2}$
- No podemos descartar el atributo de decisiones sucesivas (podemos dividirlo posteriormente en intervalos más pequeños)

- Los árboles de decisión tienden a sobreajustar los datos dado que podemos particionar los ejemplos hasta llegar a un punto donde todas las predicciones son correctas
- A medida que descendemos en el árbol, las decisiones se toman con cada vez menos datos
- Podemos regularizar el aprendizaje controlando los parámetros que definen la complejidad del árbol
- También podemos sobreajustar el árbol y podar el resultado final estimando el error de generalización de las hojas, eliminando los nodos con un error esperado mayor que no tenerlos (Post poda)

- Fijando el número mínimo de ejemplos que puede haber en una hoja
- Fijando el número máximo de nodos/hojas en el árbol
- Fijando la profundidad máxima de las ramas del árbol
- Fijando un umbral para la heurística
- Aproximando el error en los nodos mediante validación cruzada


3-Class classification Decision Tree leaf=1
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5

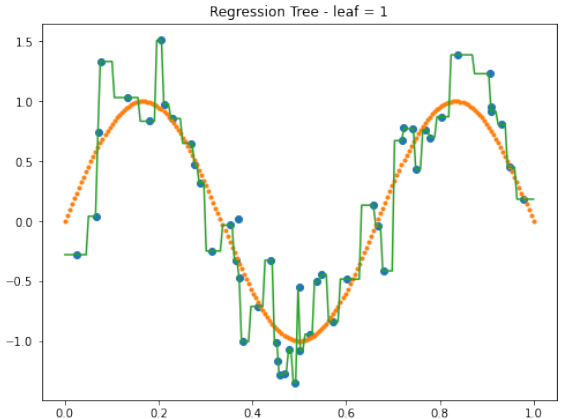

3-Class classification Decision Tree leaf=5
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5

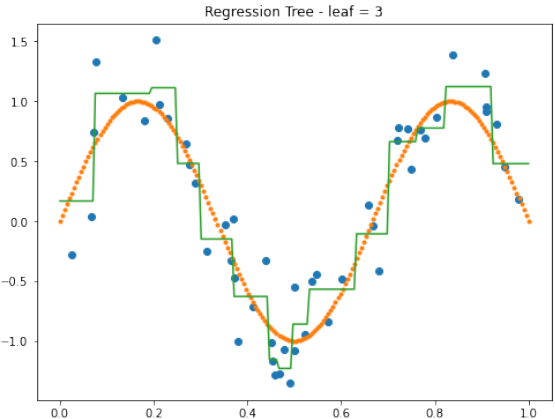
- Podemos usar la misma estrategia para realizar tareas de regresión
- Función de división: Minimización del error cuadrático de la partición dada por un atributo

$$\ell(y_n, \hat{y}_n) = (y_n - \hat{y}_n)^2$$

- La decisión en las hojas para nuevos ejemplos ($\hat{y}_n$) corresponde a la media de los valores de los ejemplos en la hoja
- Se pueden también ajustar modelos más complejos en las hojas, pero el coste computacional es mayor

- Este modelo aproxima la función usando funciones lineales por partes
- Podemos tener problemas de sobreajuste, dado que un árbol con un ejemplo por hoja tiene error cero en los datos de entrenamiento (al igual que los k-vecinos más cercanos)
- Podemos aplicar las mismas estrategias para controlar la complejidad del árbol obteniendo aproximaciones más suaves


Regression Tree - leaf = 1
1.5
1.0
0.5
0.0
−0.5
−1.0
0.0
0.2
0.4
0.6
0.8
1.0


Regression Tree - leaf = 3
1.5
1.0
0.5
0.0
−0.5
−1.0
0.0
0.2
0.4
0.6
0.8
1.0

- Su complejidad computacional es baja
- Realizan la selección automática de características, ya que solo se utilizan los atributos relevantes para la tarea
- Se pueden interpretar más fácilmente que otros modelos
  - Cada rama del árbol es una regla de decisión
  - La posición del atributo en el árbol indica la importancia del atributo para la decisión (general a específico)

- Tienen una gran variabilidad, pequeños cambios en la muestra pueden dar como resultado un modelo completamente diferente (gran variancia)
- Son sensibles al ruido y a las clasificaciones erróneas (la regularización ayuda)
- Aproximan los conceptos por particiones que son paralelas a los atributos, por lo que pueden devolver modelos complejos si las fronteras de clasificación son oblicuas

Combinación de clasificadores

Teorema del jurado de Condorcet

- Jurado de votantes que necesitan tomar una decisión binaria
- Cada votante tiene una probabilidad p de acertar
- Cada votante es independiente
- La probabilidad de que la mayoría de los votantes acierte es L
- Entonces
  - $p > 0.5$ implica $L > p$
  - L se acerca a 1, para todo $p > 0.5$ a medida que el número de votantes se acerca al infinito

Esto significa que clasificadores suficientemente independientes ligeramente mejores que una decisión al azar pueden predecir casi perfectamente combinando sus decisiones

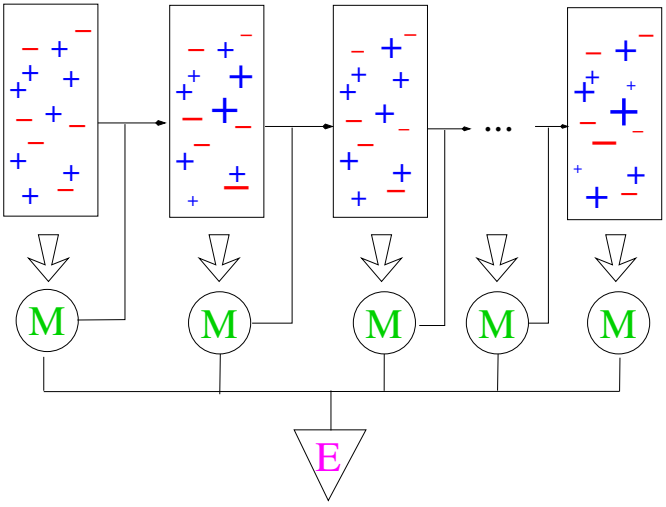
- Diversidad de opinión: Cada miembro debe tener información privada aunque sea sólo una interpretación excéntrica de los hechos conocidos
- Independencia: Las opiniones de los miembros no están determinadas por las opiniones de los otros alrededor de ellos
- Descentralización: Los miembros pueden especializarse y sacar conclusiones basadas en conocimiento local
- Agregación: Existe algún mecanismo para convertir los juicios privados en una decisión colectiva

- Podemos mejorar el acierto de clasificadores débiles (gran varianza) combinando las decisiones de un grupo de ellos (conjuntos de clasificadores)
- Un conjunto de modelos simples construidos usando diferentes conjuntos de datos puede obtener mejor acierto que un modelo más complejo
- Cada modelo tiene un *punto de vista* diferente del problema, su combinación da una visión más completa
- Reducimos la varianza con su combinación (y minimizamos el error)
- Los árboles de decisión son especialmente adecuados para estas técnicas, pero se pueden usar otros clasificadores, e incluso podemos mezclar diferentes métodos

- Conjunto de entrenamiento: Un conjunto de datos etiquetado que se utiliza para el entrenamiento de conjunto
- Inductor Base: Un algoritmo de inducción que recibe un conjunto de entrenamiento y forma un clasificador que representa la relación generalizada entre los atributos de entrada y el atributo objetivo
- Generador de Diversidad: Responsable de generar el conjunto de clasificadores diversos
- Combinador: Responsable de combinar las clasificaciones del conjunto

Métodos

- **Boosting** genera una secuencia de clasificadores
- Asume que los ejemplos tienen un peso que determina su importancia
- Se generan clasificadores sucesivos aumentando el peso de los ejemplos que no son predichos correctamente y disminuyendo el peso de los ejemplos que se predijeron correctamente
- Cada clasificador se especializa en los ejemplos difíciles para el clasificador anterior
- Diversidad se genera modificando el peso de los ejemplos en cada iteración
- La predicción se obtiene votando o promediando

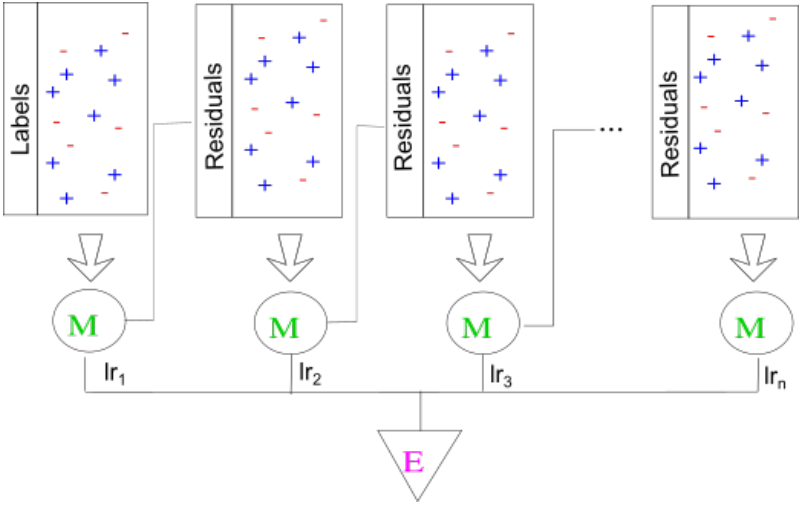

M
M
M
M
M
E

- Gradient Boosting aborda todos los problemas como tareas de regresión usando una función de pérdida diferenciable y árboles de regresión
- Para clasificación consideramos que estamos aproximando la distribución de probabilidad en las hojas
- Se puede ajustar con diferentes funciones de pérdida
  - Para regresión: MSE, MAE, cuantil ...
  - Para clasificación: entropía cruzada

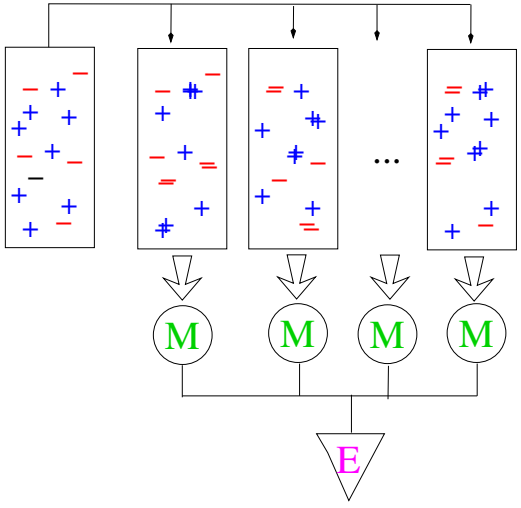
- Cada árbol está construido para predecir el residuo de la pérdida restante de todos los árboles anteriores
- Diversidad se puede obtener submuestreando los datos
- La predicción está compuesta por la suma de las predicciones de todos los árboles multiplicada por una tasa de aprendizaje

$$f(x_n) = f_0(x_n) + w_1 f_1(x_n) + \cdots w_t f_t(x_n) \quad w_1 > w_2 > \cdots > w_t$$

- Esta tasa de aprendizaje se reduce monótonamente en cada iteración, por lo que el modelo no sobre ajusta


Labels
Residuals
Residuals
...
Residuals
M
M
M
M
lr1
lr2
lr3
lrn
E

- **Bagging (Bootstrapping Aggregation)** genera un conjunto de clasificadores independientes
- Todos los clasificadores se entrenan de la misma manera, esto significa que se puede paralelizar
- Diversidad se obtiene generando conjuntos de datos a partir del original utilizando muestreo con o sin reemplazo (Boostrapping) y seleccionando subconjuntos de características
- La predicción se obtiene mediante votación o promedio


...
M
M
M
M
E

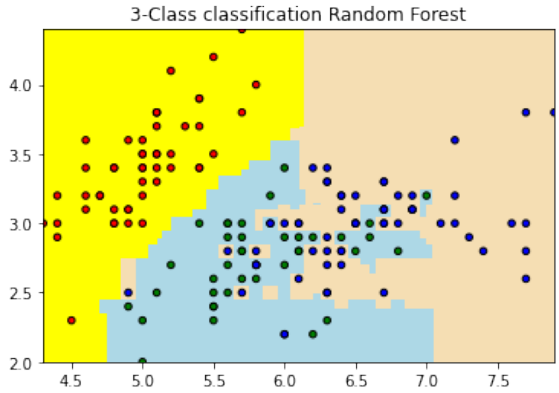
- Se construye un conjunto de árboles con bootstrapping con los datos
- Diversidad se obtiene al construir los árboles eligiendo aleatoriamente los atributos para cada nodo
- Random Forest: Para un nodo, del conjunto de atributos se elige un subconjunto, y se calcula la mejor división para cada atributo, usando para el nodo el atributo con la puntuación más alta
- Extremely Randomized Trees: Lo mismo que random forest, pero la división de cada atributo se decide ahora al azar, el atributo con la mejor puntuación se usa para el nodo

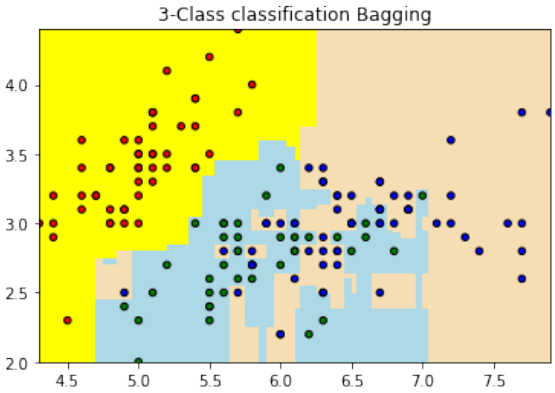
- Las estrategias generales para conjuntos de clasificadores (no específicas para árboles) se pueden usar con otros modelos como clasificador base siempre que generen modelos diversos
- Tampoco estamos obligados a usar un solo modelo en un conjunto
- Diferentes algoritmos tienen diferentes sesgos, por lo que tendrán una perspectiva diversa del problema

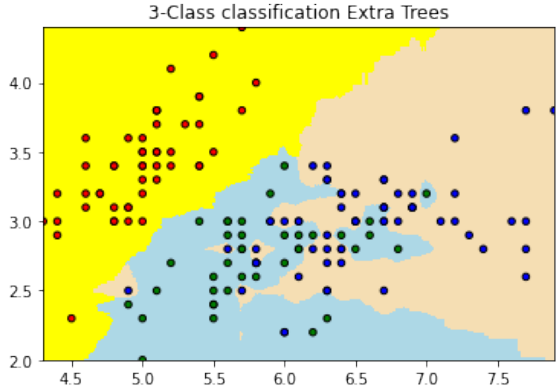
Voting: Entrenar N clasificadores diferentes (diferentes algoritmos) y usar el voto por mayoría para obtener el resultado

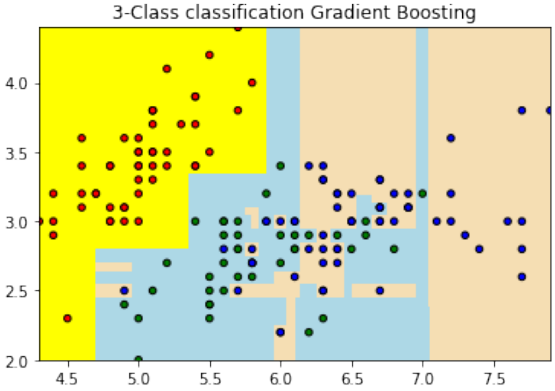
- Votación dura = mejor clase por clasificador
- Votación blanda = Promedio de las probabilidades de cada clasificador (softmax), elegir la mejor clase del promedio

Stacking: Entrenar N clasificadores (diferentes algoritmos o diferentes hiperparámetros) para generar N predicciones y usar otro clasificador para aprender a combinar las decisiones (dura/blanda)


3-Class classification Random Forest
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5


3-Class classification Bagging
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5


3-Class classification Extra Trees
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5


3-Class classification Gradient Boosting
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5

Este notebook muestra la superficie de predicción de árboles de decisión y conjuntos de clasificadores usando diferentes hiperparámetros

Interpretabilidad/Explicabilidad

- Los árboles de decisión/regresión son un modelo de caja blanca
- La jerarquía refleja la importancia de las diferentes características en diferentes partes del problema
- Las distintas ramas de los árboles corresponden a reglas de decisión que se pueden interpretar (si no son demasiado largas)
- El resultado del modelo puede ser explicado por el camino que se sigue en el árbol
- Este modelo también se puede usar como modelo sustituto para LIME

- Las combinaciones hacen imposible interpretar las decisiones del modelo (caja negra)
- Estos modelos proporcionan un método para calcular la importancia de cada variable en las decisiones globales del modelo
- Esta se calcula usando la contribución de cada atributo a la pureza de las hojas de los árboles
- Dado el sesgo de las heurísticas utilizadas por los algoritmos de árboles, las importancias deben usarse con cuidado
- Las explicaciones de ejemplo deben obtenerse utilizando métodos independientes del modelo

- Los valores de Shapley son una técnica agnóstica del modelo
- Tienen su base matemática en la teoría de juegos.
- Se utiliza para calcular la atribución a los jugadores en un juego cooperativo
- El objetivo es cuantificar cuánto ha contribuido cada jugador al resultado final del juego.
- Las características se pueden interpretar como jugadores
- El principal problema es que el cálculo exacto es muy costoso, se trata de evaluar un problema combinatorio

- Los valores de Shapley se pueden aproximar usando un modelo aditivo y un kernel binario especializado
- Este cálculo involucra el muestreo de diferentes subconjuntos de características y la estimación de la contribución de cada uno de ellos usando el modelo
- Las muestras se pueden usar para entrenar un modelo lineal que se aproxima al valor de Shapley para cada característica
- Este método todavía es computacionalmente costoso
- Se ha desarrollado un método específico para modelos basados en árboles (TreeSHAP) que reduce el costo computacional

- Explicación de ejemplos (contribución de las características al resultado)
- Importancia global de las características (agregado de valores SHAP para el conjunto de datos)
- Comparación de la importancia de las características con los efectos de las características (como en los gráficos de efectos para la regresión lineal)
- Gráficos de dependencia de características (correspondencia entre valores de características y valores SHAP)
- Valores de interacción SHAP (interacción entre pares de valores)

- Efectos de los valores de las características en la predicción
  - Gráficos de dependencia parcial
  - Expectativas condicionales individuales
  - Efectos locales acumulados
- Explicaciones basadas en ejemplos
  - Explicaciones contrafactuales (qué cambios en la característica hacen que la predicción cambie a un resultado específico)
  - Ejemplos adversarios (qué ejemplos pueden engañar al modelo)
  - Prototipos (ejemplos que representan todos o parte de los datos)

Este notebook muestra un ejemplo de árboles de decisión y conjuntos de clasificadores con una explicación de los modelos

Este notebook muestra un ejemplo de árboles de decisión y conjuntos de clasificadores con una explicación usando valores Shapley