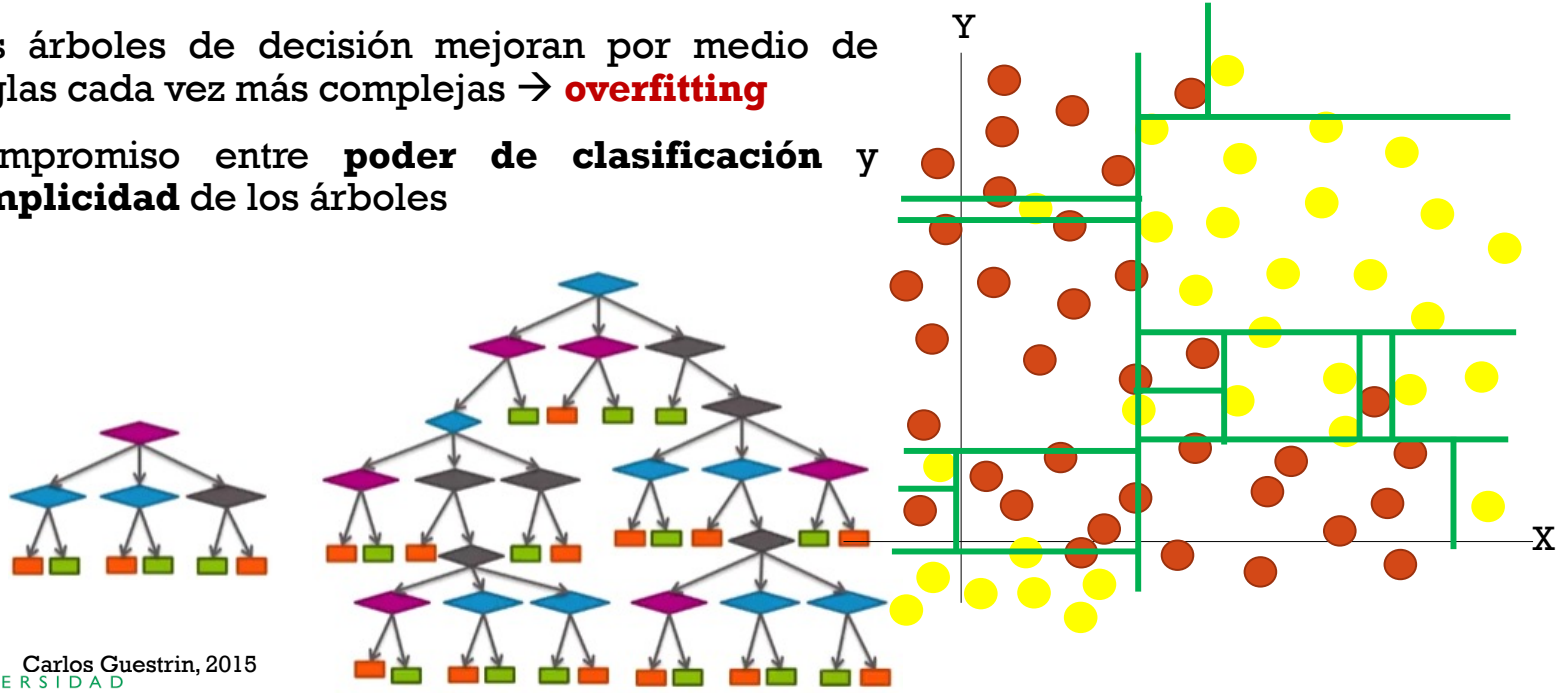
PODA

UNIVERSIDAD ICESI

ÁRBOLES DE DECISIÓN: PODA

Los árboles de decisión mejoran por medio de reglas cada vez más complejas → **overfitting**

Compromiso entre **poder de clasificación** y **simplicidad** de los árboles



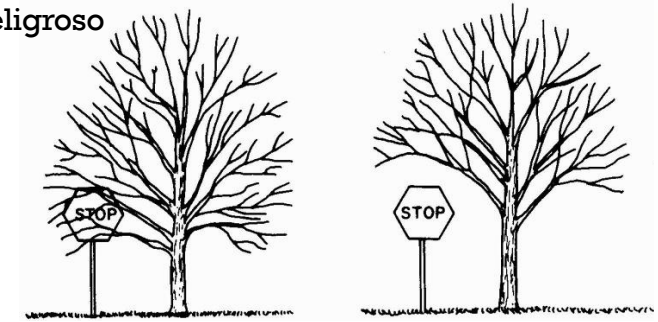
Carlos Guestrin, 2015

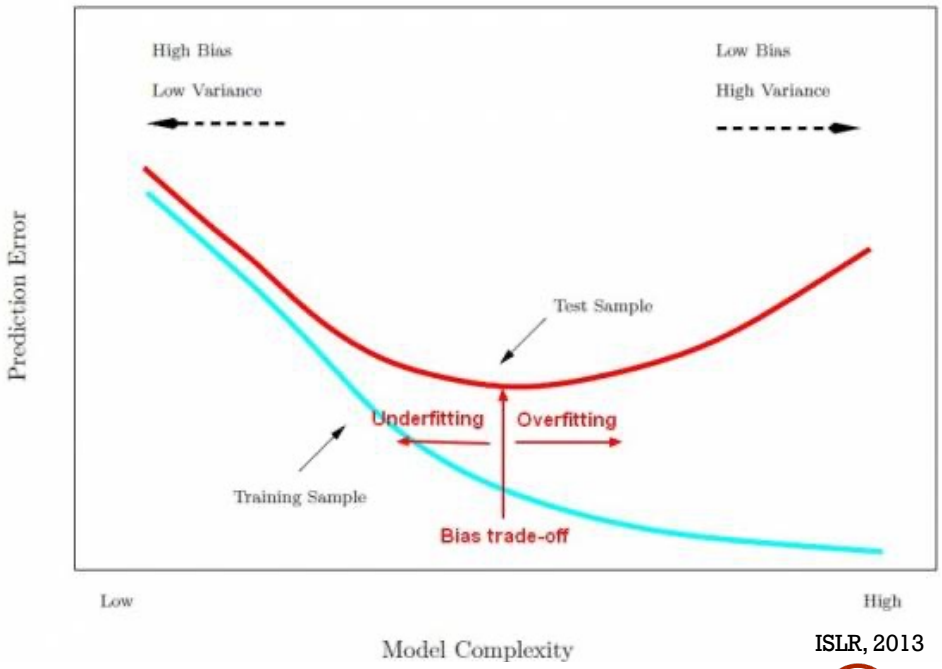
UNIVERSIDAD ICESI

ÁRBOLES DE DECISIÓN: PODA

Pre-Poda: Criterios de **parada temprana**

- Máxima **profundidad**: limita el crecimiento del árbol de manera global → usa CV
- Mínimo **número de instancias** para permitir particionamiento: limita localmente en cada nodo el crecimiento del árbol → usa CV
- No continuar si no se mejora suficiente el criterio de **particionamiento** o el **error** de entrenamiento → peligroso





ISLR, 2013

UNIVERSIDAD ICESI

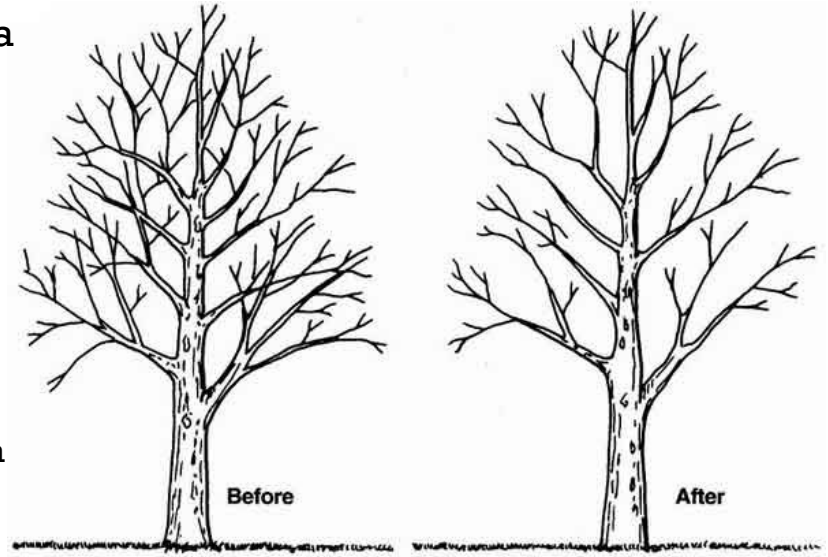
ÁRBOLES DE DECISIÓN: PODA

Post-poda: simplificar el árbol una vez se haya terminado de aprender:

- Preferible a la **pre-poda**
- Complejidad en términos de número de nodos hoja (terminales) L(T), no necesariamente de profundidad
- Podar bottom-up, teniendo en cuenta ahora también la complejidad del árbol:

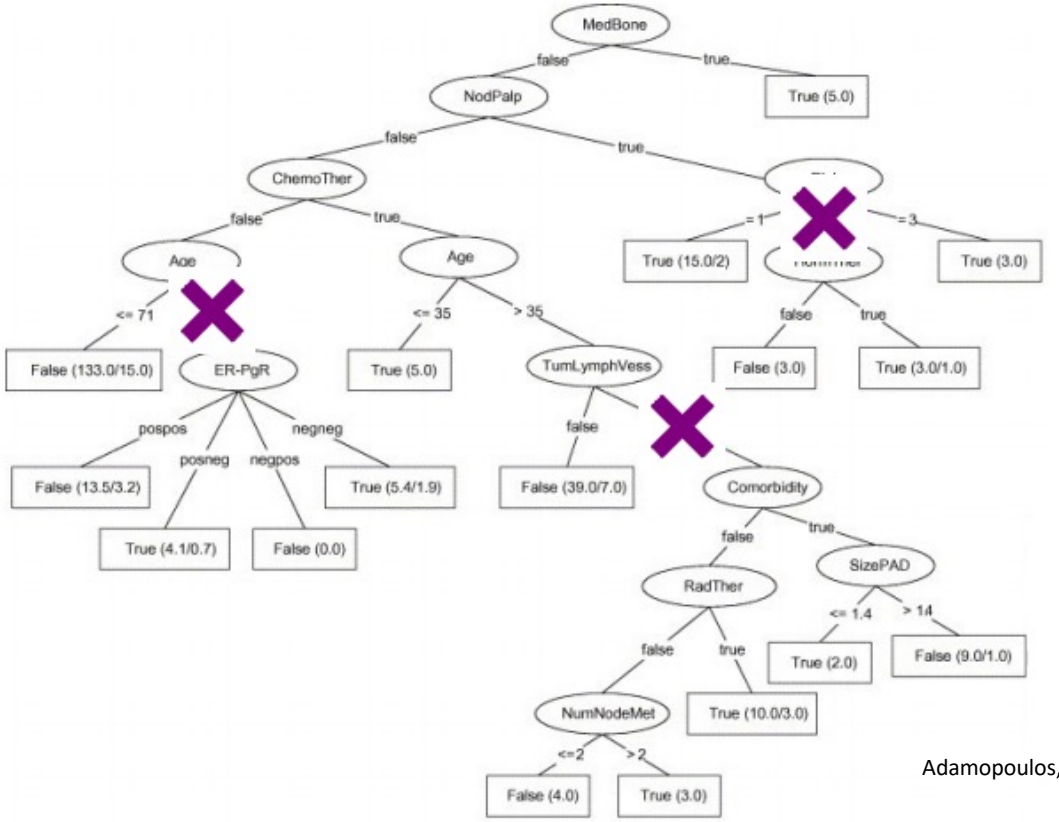
  Costo = Criterio(T)+ α*L(T), donde α controla la complejidad del modelo (α se puede estimar a través de CV).

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



UNIVERSIDAD ICESI

ÁRBOLES DE DECISIÓN: PODA

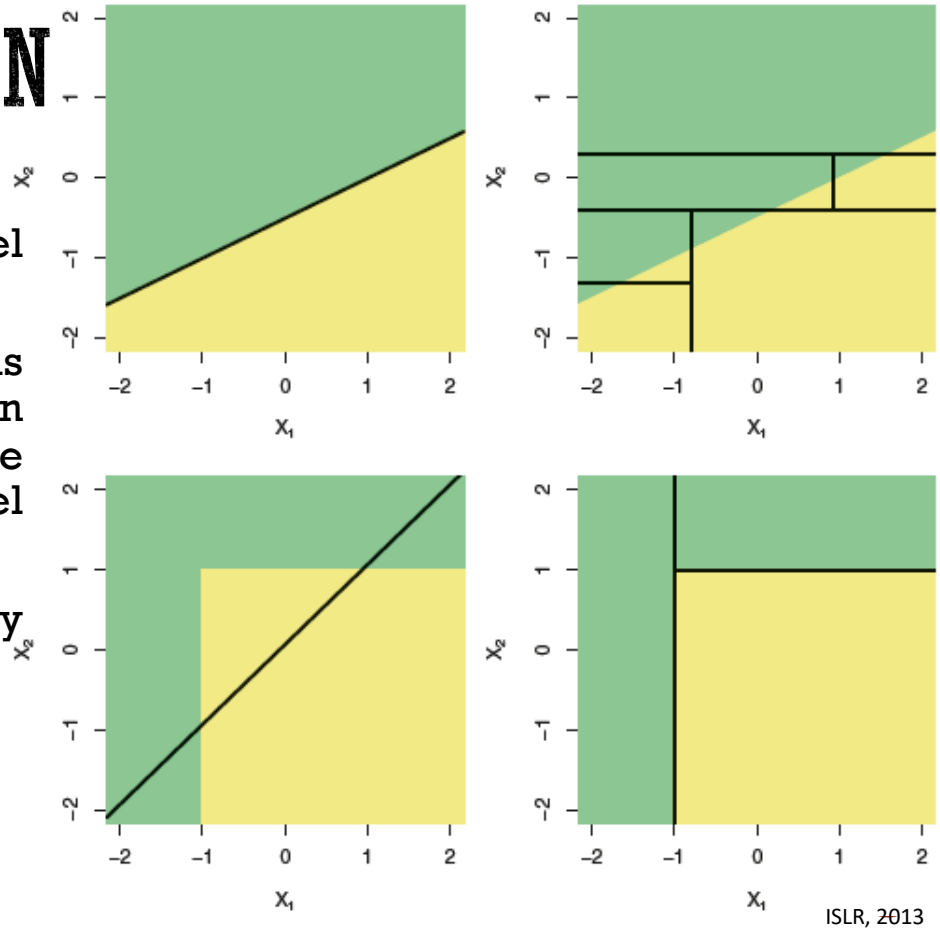


Adamopoulos, NYU, 2014

ÁRBOLES DE DECISIÓN

Consideraciones:

- Su rendimiento depende del comportamiento de los datos
- No son los algoritmos más competitivos, pero producen un modelo simple, altamente interpretable y que asemeja el razonamiento humano
- Tienden al overfitting, por lo que hay que utilizar estrategias de poda



ISLR, 2013

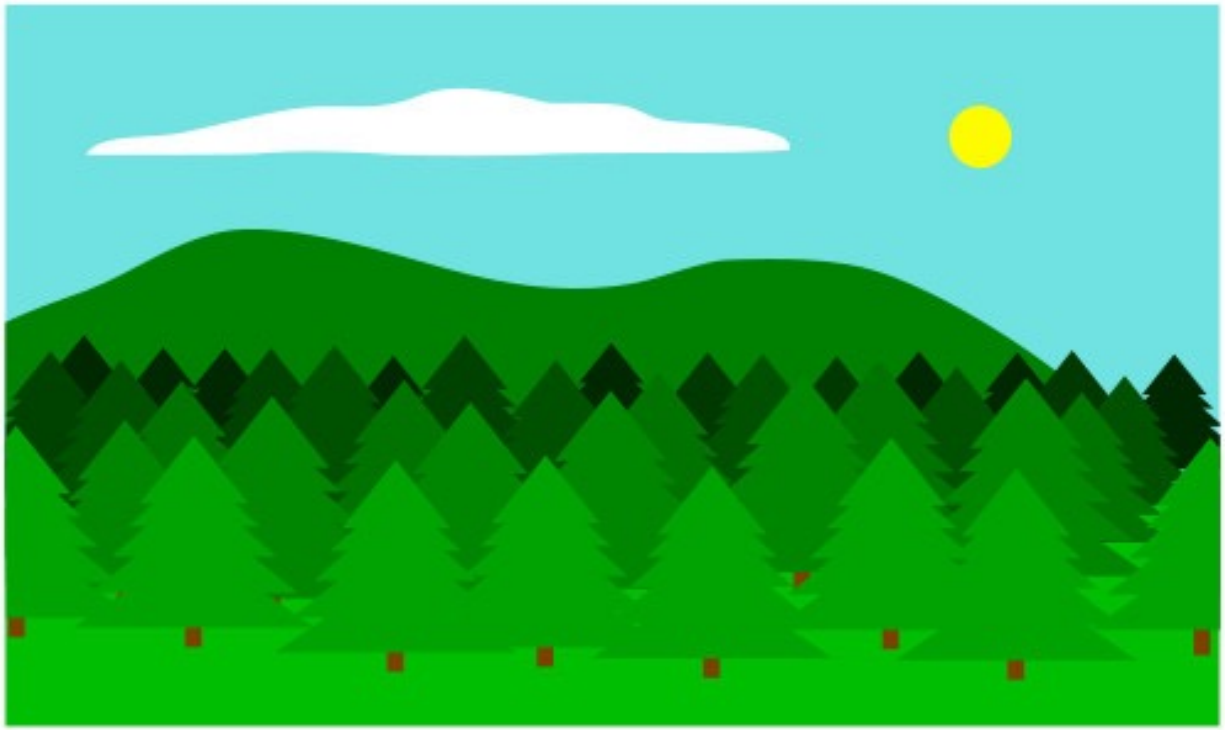
UNIVERSIDAD ICESI

ÁRBOLES DE DECISIÓN

Consideraciones:

- Son indiferentes a escalamiento, distribución de los datos y a datos faltantes
- Permiten la consideración de atributos cualitativos sin necesidad de crear variables adicionales (al menos en R, en Python obligan a variables numéricas)
- Crean un ranking de importancia de todas las variables predictivas
- Excluyen automáticamente variables no óptimas en el particionamiento
- Son sensibles a bases de datos desbalanceadas y a pequeños cambios en los datos
- Sobreaprenden y subaprenden fácilmente
- Son sesgados hacia los atributos con mayor cantidad de valores
- Son la base de modelos de ensamble (Bagging, Boosting, Random forest), que ofrecen un excelente poder predictivo, mientras reducen el error de varianza (overfitting)

ENSEMBLE LEARNING



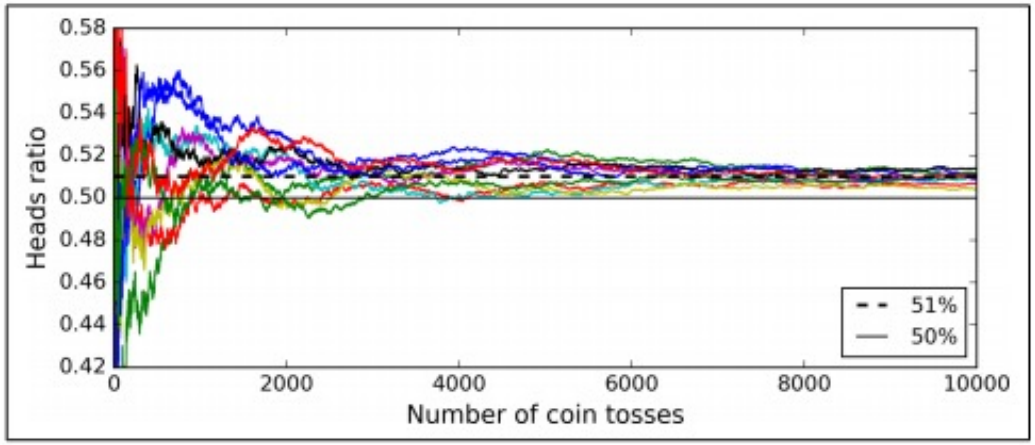
UNIVERSIDAD ICESI

ENSEMBLE LEARNING

Cómo verificar que una moneda no esté trucada?

→ Lanzar la moneda n veces

→ A mayor número de lanzamientos, mejor la estimación



Géron, 2017

UNIVERSIDAD ICESI

ENSEMBLE LEARNING

- Diversificación:
  - Cada modelo tiene un riesgo de mala predicción
  - ¿Por qué limitar la decisión a un solo modelo si podemos construir y utilizar varios?

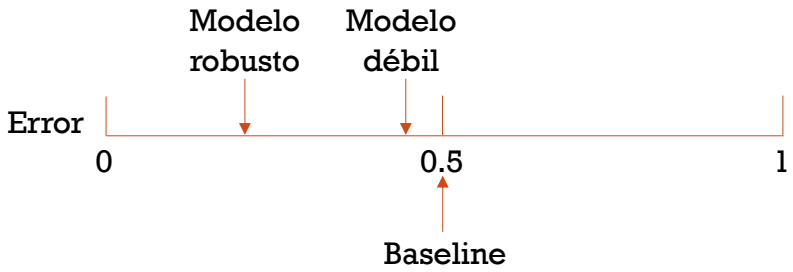
    → Combinar un grupo de clasificadores/regresores

    → Decisión basada en la agregación de varios modelos (max / promedio)
- Reducir la varianza global al agregar un conjunto de modelos supervisados, y obtener mejor generalización
- Se aumenta la calidad de la predicción pero se pierde en interpretabilidad

UNIVERSIDAD ICESI

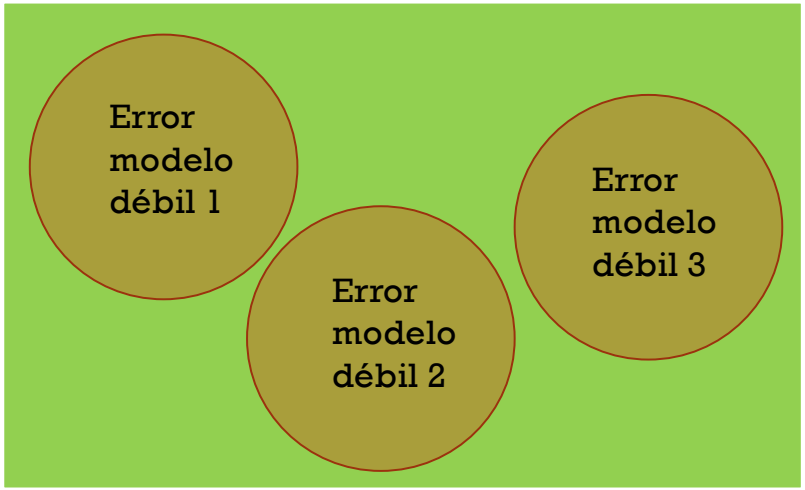
ENSEMBLE LEARNING

- Modelos débiles vs. robustos:
  - Los modelos robustos tenderán a estar de acuerdo en la mayoría de los casos
  - A menudo los datos permiten construir modelos que son apenas mejores al baseline



UNIVERSIDAD ICESI

BAGGING (BOOTSTRAP)



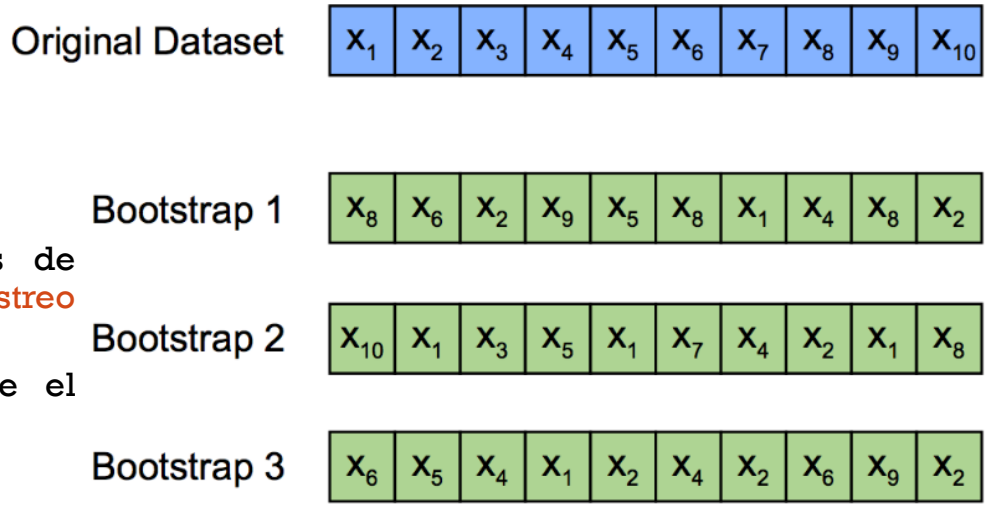
- ¿Cómo es nuestro modelo si ponemos a votar a los 3 modelos?

→ Entre más de-correlacionados estén los modelos individuales, mejor va a ser el meta-modelo agregado

UNIVERSIDAD ICESI

BAGGING

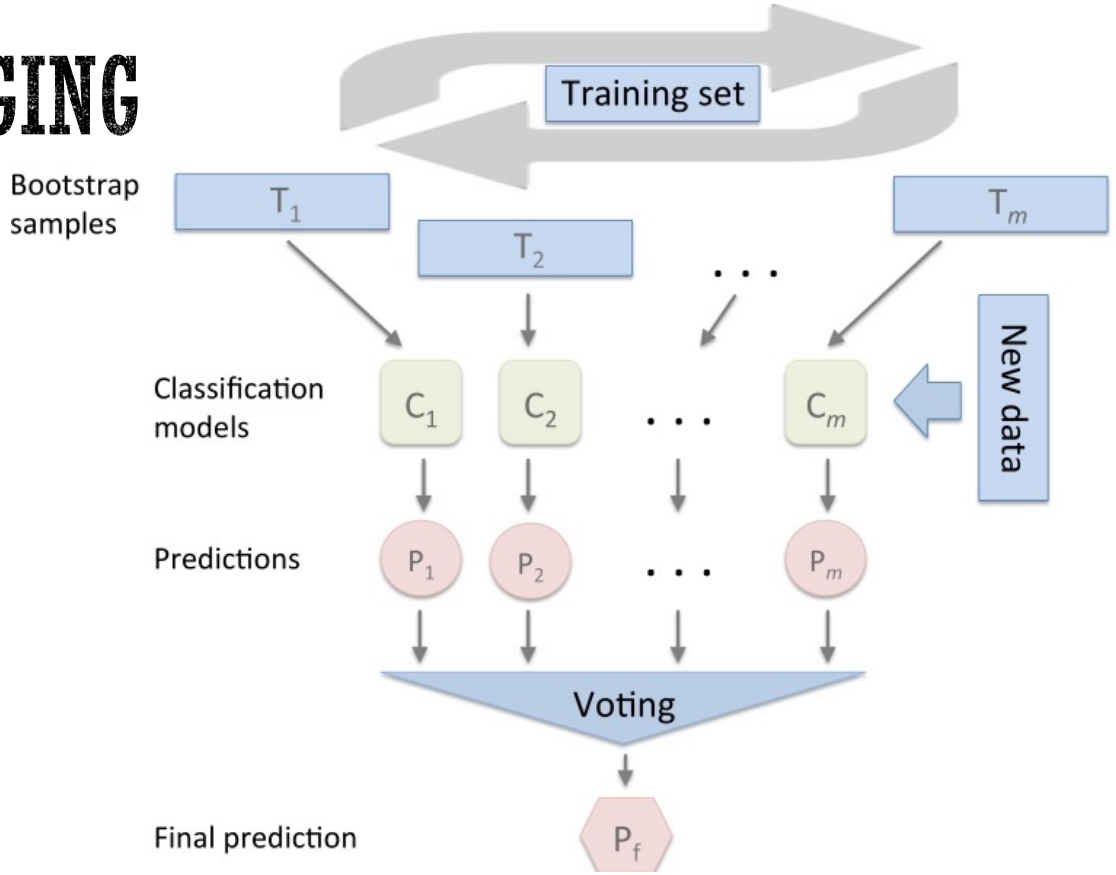
- Bagging = Bootstrap aggregating
- Bootstrap: técnica de muestreo
  - Consideración de varios conjuntos de entrenamiento/test utilizando muestreo con reemplazo
  - Por lo general los muestreos tiene el mismo tamaño del conjunto original
  - Reduce la varianza



Sebastian Raschka, 2015

UNIVERSIDAD ICESI

BAGGING



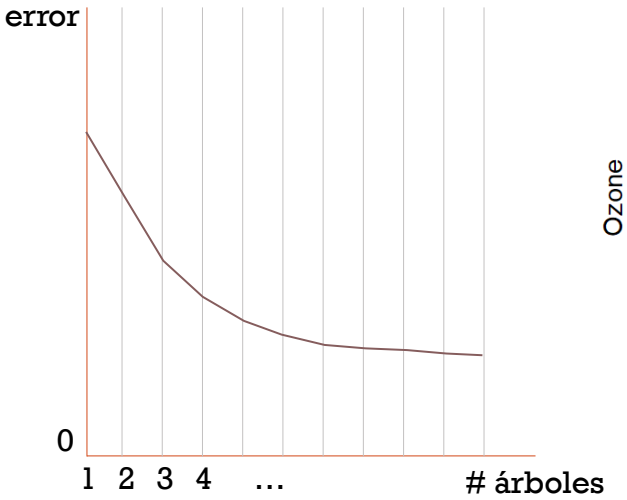
Raschka, 2017

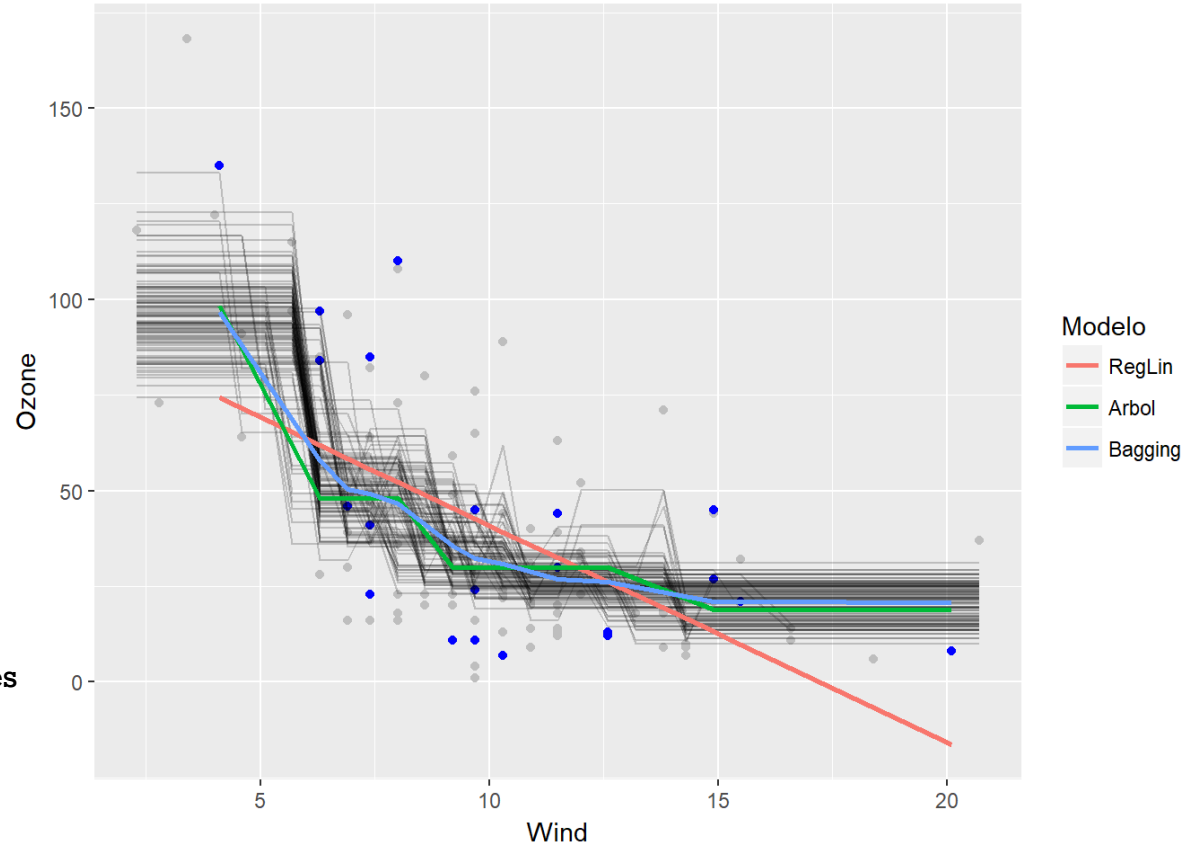
BAGGING

- Algoritmo (Breiman, 1994-1996):
  1. Escoger múltiples subconjuntos de instancias con repetición
  2. Entrenar varias instancias de un mismo tipo de modelo "**robusto**" de aprendizaje (bajo sesgo – alta varianza) basado en cada subconjunto
  3. Agregar las decisiones de cada modelo en una sola decisión global (votación, promedio)
- Muy útil cuando se trata de modelos no lineales (e.g. árboles de decisión)
- Resultado: sesgo similar, reducción de varianza (estabilización de las predicciones) → mejora el accuracy
- Disminuye la interpretabilidad

UNIVERSIDAD ICESI

BAGGING





UNIVERSIDAD ICESI

BAGGING

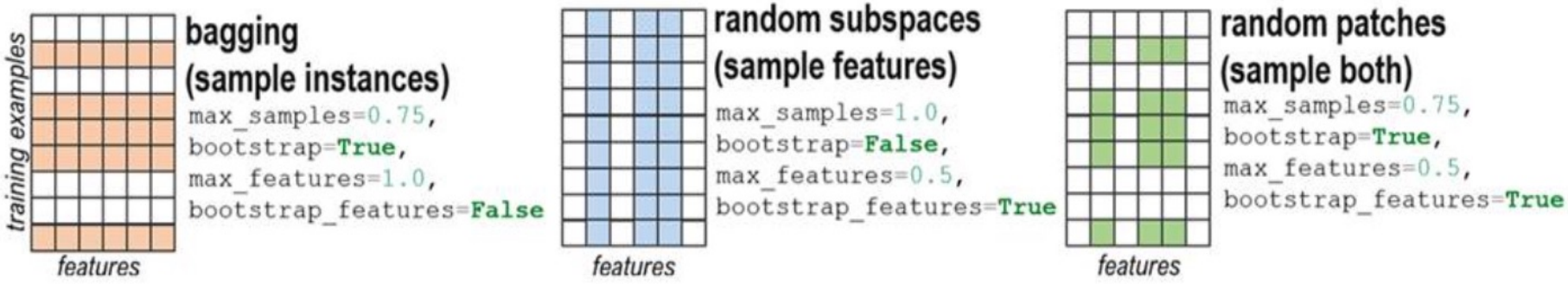
OOB (Out of bag error estimation):

- Un subconjunto bootstrap contiene, en promedio, aproximadamente 2/3 de los datos del conjunto original
- OOB usa el 1/3 restante de datos para calcular métricas de evaluación de los modelos individuales, que luego deben agregarse
- Con una cantidad elevada de árboles, tiende hacia el error LOOCV :
  - Evita tener que dejar un test set independiente
  - Evita el costoso proceso de K-Fold CV

BAGGING

- Se aplica tanto a clasificadores como a regresores

Figure 2.9 Bagging compared to random subspaces and random patches.



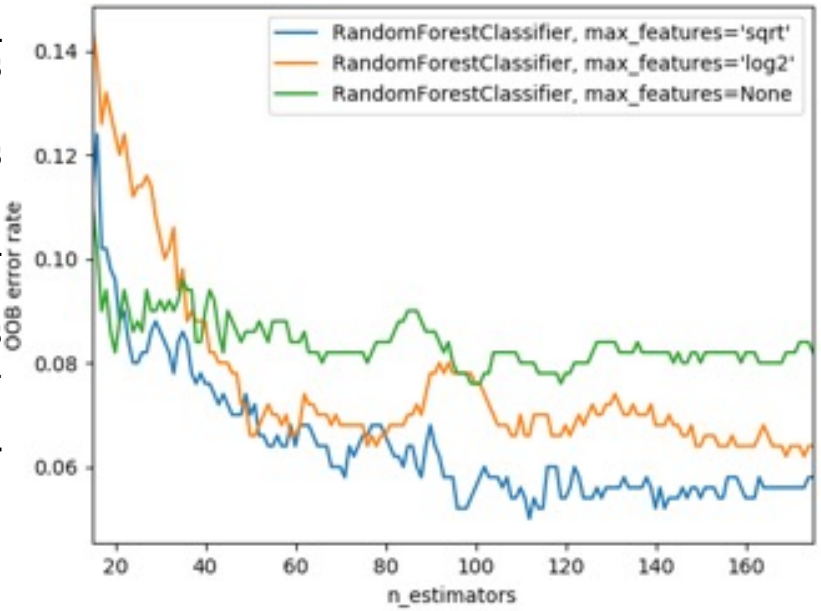
- **Random patches**: subconjuntos tanto de los registros como de las variables independientes

Random patching

UNIVERSIDAD ICESI

RANDOM FOREST

- Junto a boosting, uno de los algoritmos con mejor performance en casos de datos **estructurados**
- Extensión de bagging basado en árboles CART
- Cada árbol es **independiente** y tiene la **misma importancia** en la decisión final
- La idea es minimizar la **correlación** entre los modelos, forzando la diferencia más alla del bootstrap de los datos de entrenamiento
- Limita el **número de atributos** a considerar en cada nodo de particionamiento.
- Valores por defecto:
  - En caso de regresión: m/3
  - En caso de clasificación: $\sqrt{m}$

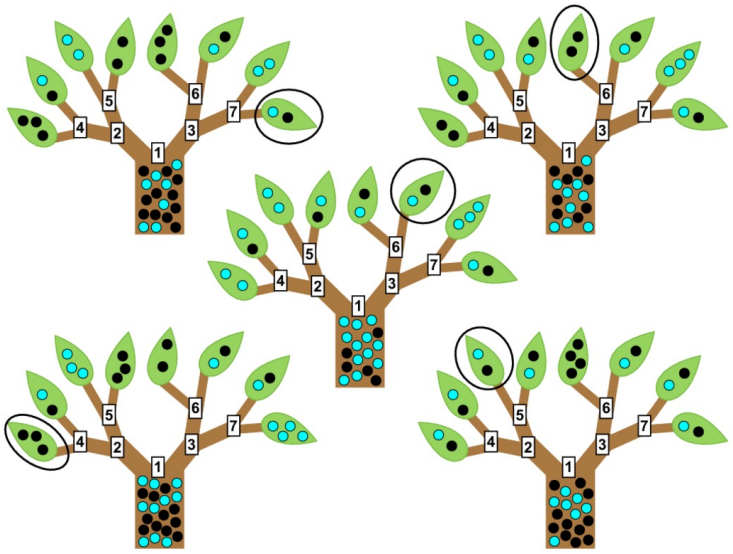


Scikit-learn.org

UNIVERSIDAD ICESI

RANDOM FOREST

- Algoritmo
  1. Crear múltiples árboles de decisión, construidos sobre muestras con reemplazo (bootstrapping) del dataset
  2. En cada punto de partición, considerar solo una muestra aleatoria de todas las variables disponibles
  3. Crecer el árbol
  4. Agregar los resultados de todos los árboles creados



http://inspirehep.net/record/1335130/plots

UNIVERSIDAD ICESI

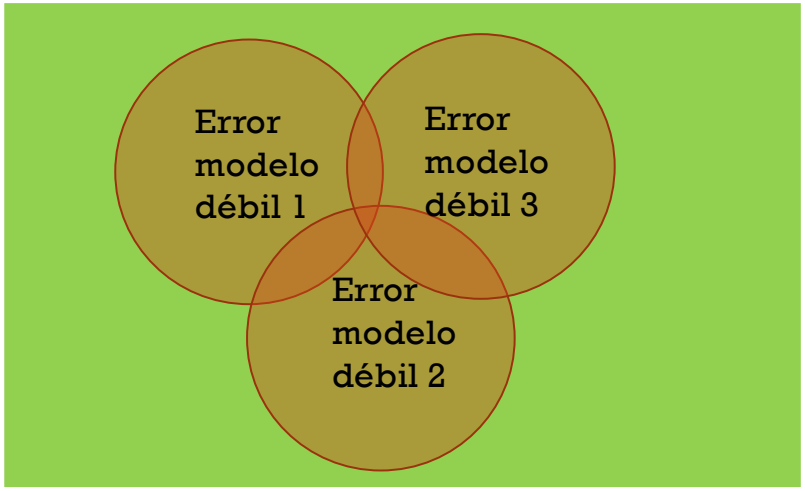
RANDOM FOREST

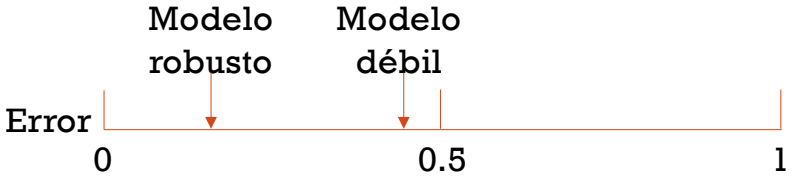
- Consideraciones
  - Al igual que el bagging:
    - **Heredan** cosas buenas de los árboles de decisión (importancia de las variables), mejoran las malas (varianza)
    - **Lento** (gran número de árboles a entrenar) y toman mucho espacio en memoria, pero **escalable** (independencia)
    - No genera **overfitting** con más árboles, pero los árboles individuales pueden tenerlo (limitar su profundidad)
    - Muy difíciles de interpretar
  - Requieren de un buen afinamiento de parámetros (bagging es mas simple)

UNIVERSIDAD ICESI

BOOSTING

- Modelos débiles vs. robustos





- ¿Cómo es nuestro modelo si ponemos a votar a los 3 modelos?
- ¿Hay ciertas regiones mas importantes en cuanto al error que otras?
- ¿Cómo mejoro en las regiones más susceptibles al error donde ningún modelo logra buenos resultados?

UNIVERSIDAD ICESI

BOOSTING

- Agregación de predictores **débiles**, aprendidos **secuencialmente**
- Aprendizaje lento progresivo: reduce el error de sesgo al considerar los errores previos en los modelos subsiguientes.
- La **importancia de cada registro** se modifica según los resultados del último modelo entrenado → Cada modelo es **dependiente** de los modelos anteriores
- Cada modelo tiene **diferente influencia** en la decisión final; este depende de su poder predictivo
- Puede caer en **overfitting**

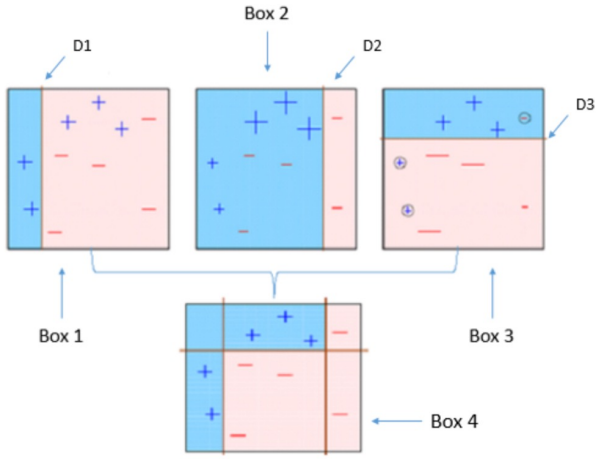
UNIVERSIDAD ICESI

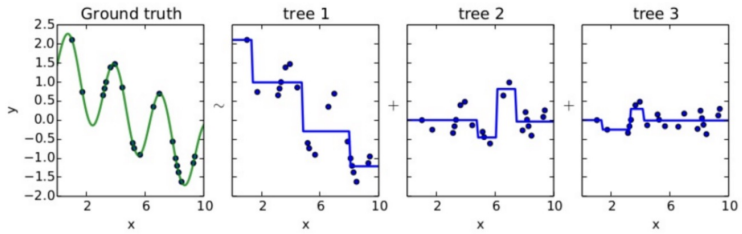
BOOSTING

- Algoritmo
  1. Crear un modelo predictor **débil** inicial sobre el dataset
  2. Asignar un **peso** a la decisión del modelo dado su **error** de entrenamiento
  3. Actualización de los pesos de cada instancia del dataset: se aumenta en las instancias que fueron mal predichas, se reduce en las instancias correctas
  4. Crear un nuevo predictor, influenciado por los errores cometidos por los modelos anteriores
  5. Agregar una decisión ponderada de todos los predictores creados
- Tres parámetros importantes:
  - Número de árboles a desarrollar
  - Learning rate (λ) o shrinking parameter: controla la velocidad del aprendizaje, para limitar el overfitting (valores típicos 0.01 - 0.001)
  - Número de particiones. Controla la complejidad de los árboles. Se conoce como interacción de profundidad máxima

VARIANTES DE BOOSTING

- ADA BOOST
- GRADIENT BOOSTING
- EXTREME GRADIENT BOOSTING (XGBOOST)
  - Hands-On Machine Learning with Scikit-Learn and TensorFlow. Aurélien Géron. 2017. O'Reilly Medi





Source: Quora

UNIVERSIDAD ICESI

TALLER: BAGGING, RANDOM FOREST, BOOSTING

- Vamos a analizar diferentes configuraciones de bagging y random forest para crear un clasificador del conjunto de datos 01_churn.csv que consideren diferentes valores de los parámetros #árboles y depth
- Continuar con el taller 08_Arboles_Churn-STUD.html

UNIVERSIDAD ICESI

REFERENCIAS

- *Introduction to Statistical Learning with Applications in R (ISLR), G. James, D. Witten, T. Hastie & R. Tibshirani, 2014*
- *Practical Data Science with R (2nd Edition)*, Nina Zoumel & John Mount, Manning, 2019
- *R in Action*, Robert I. Kabacoff, Manning, 2015

UNIVERSIDAD ICESI