Lecture 9: Machine Learning
Bosques

Big Data and Machine Learning en el Mercado Inmobiliario
Educación Continua

Ignacio Sarmiento-Barbieri

Universidad de los Andes

November 7, 2023

Motivación

- Queremos predecir:

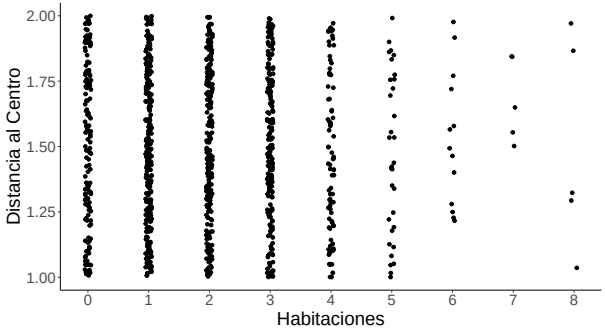
$$Price = f(structural\ attributes, amenities, ...) \tag{1}$$

  - Podemos aplicar linear regression,

$$Price = \beta_0 + \beta_1 Habitaciones + \beta_2 DCBD + u \tag{2}$$

- Aplicar OLS a este problema requiere tomar algunas decisiones.

Motivación

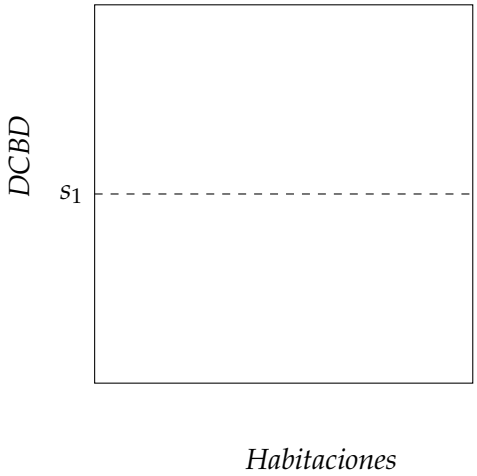

Distancia al Centro
2.00
1.75
1.50
1.25
1.00
0
1
2
3
4
5
6
7
8
Habitaciones

Árboles

"Recursive binary splitting"

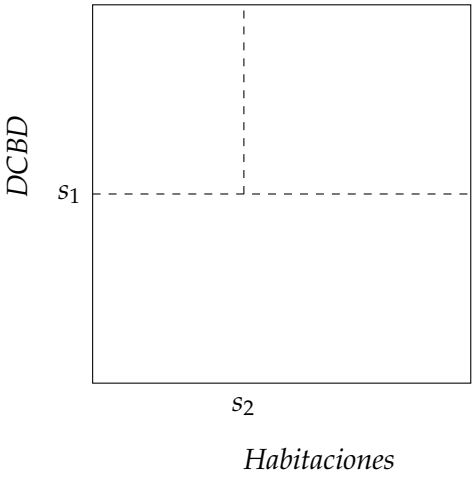
Árboles

"Recursive binary splitting"

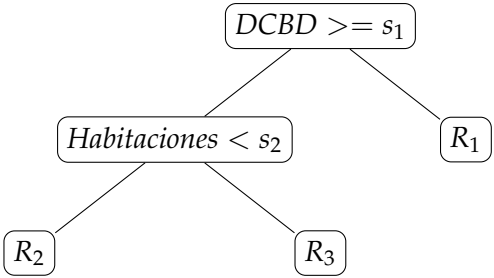


Árboles

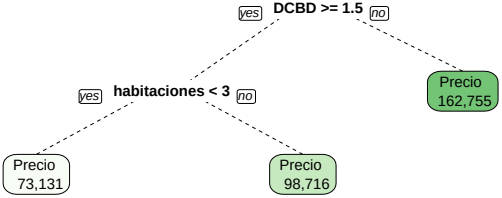
"Recursive binary splitting"

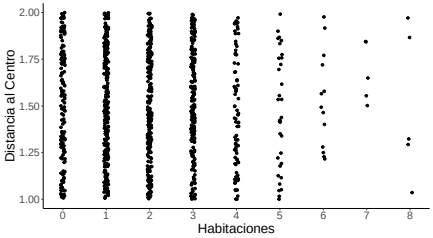


Árboles

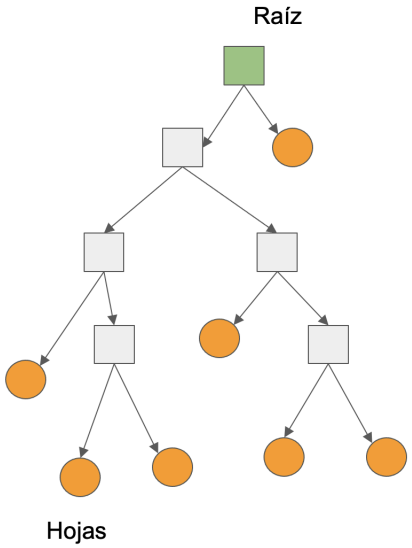


Árboles


yes
DCBD >= 1.5
no
yes
habitaciones < 3
no
Precio
162,755
Precio
73,131
Precio
98,716


Distancia al Centro
2.00
1.75
1.50
1.25
1.00
0
1
2
3
4
5
6
7
8
Habitaciones

Arboles: Sobreajuste



Sobreajuste. Algunas soluciones

- Fijar la profundidad del árbol.
- Fijar la mínima cantidad de datos que están contenidos dentro de cada hoja.
- Pruning (poda).
  - Dejar crecer un árbol muy grande T_0
  - Luego cortarlo obteniendo sub-árbol (*subtree*)
  - Como cortarlo? $\Rightarrow$ *Cost complexity pruning (cortar las ramas mas débiles)*

Ejemplo

photo from https://www.dailydot.com/parsec/batman-1966-labels-tumblr-twitter-vine/

Bagging

- Problema con CART: pocos robustos.
- Podemos mejorar mucho el rendimiento mediante la agregación
- Idea: la varianza del promedio es menor que la de una sola predicción.

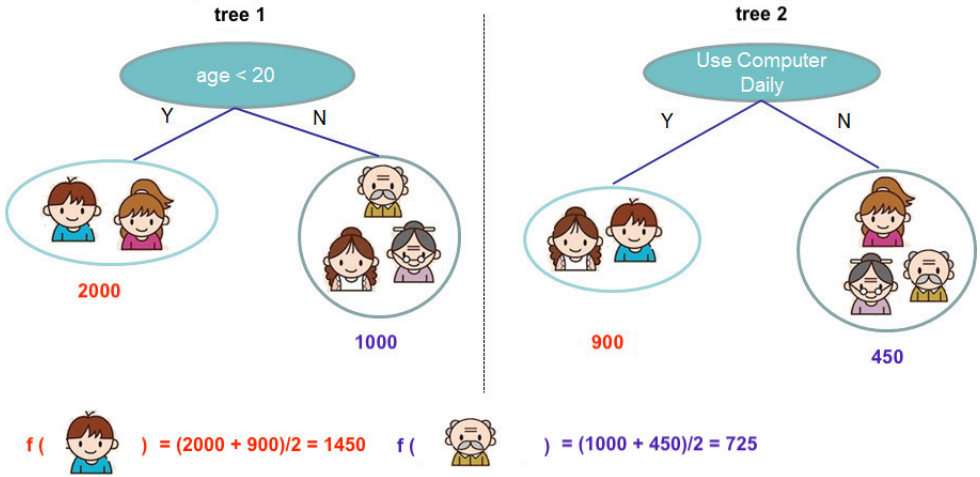
Bagging

- Bagging:
  - Obtenga repetidamente muestras aleatorias $(X_i^b, Y_i^b)_{i=1}^N$ de la muestra observada (bootstrap).
  - Para cada muestra, ajuste un árbol de regresión $\hat{f}^b(x)$
  - Promedie las muestras de bootstrap

$$\hat{f}_{bag} = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^b(x) \tag{3}$$

- Básicamente estamos suavizando las predicciones.

Bagging


tree 1
age < 20
Y
N
2000
1000
tree 2
Use Computer Daily
Y
N
900
450
f ( ) = (2000 + 900)/2 = 1450
f ( ) = (1000 + 450)/2 = 725

Random Forests

- Problema con el bagging: si hay un predictor fuerte, diferentes árboles son muy similares entre sí.
- Bosques (forests): reduce la correlación entre los árboles en el boostrap.
- Si hay p predictores, en cada partición use solo $m < p$ predictores, elegidos al azar.
- Bagging es forests con $m = p$ (usando todo los predictores en cada partición).
- m es un hiper-parámetro, $m = \sqrt{(p)}$ es un benchmark

Ejemplo

photo from https://www.dailydot.com/parsec/batman-1966-labels-tumblr-twitter-vine/

Boosting: Motivation

- Problema con CART: varianza alta.
- Podemos mejorar mucho el rendimiento mediante la agregación
- El boosting toma esta idea pero lo ”encara” de una manera diferente $\rightarrow$ viene de la computación
- Va a usar arboles pequeños y a aprender de los errores

Boosting Trees

- La idea es aprender de los errores lentamente.
- Ajustamos un árbol utilizando los errores del modelo.
- Cada uno de estos árboles puede ser bastante pequeño.
- Esto permite mejorar lentamente aprendiendo $f(.)$ en áreas donde no funciona bien.
- OJO: a diferencia de *bagging*, la construcción de cada árbol depende en gran medida de los árboles que ya han crecido.

Boosting Trees: Algoritmo

1. Iniciamos fijando $\hat{f}(x) = 0$ y $r_i = y_i$ para todos los i del training set
2. Para $m = 1, 2, ..., M$
   1. Ajustamos un árbol $\hat{f}^m$ con d bifurcaciones ($d + 1$ hojas)
   2. Actualizamos $\hat{f}(x)$ con una versión "shrunken" del nuevo árbol

$$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^m(x) \tag{4}$$

   3. Actualizamos los residuales

$$r_i \leftarrow r_i - \lambda \hat{f}^m(x) \tag{5}$$

3. El modelo final es

$$\hat{f}_{boost} = \sum_{m=1}^{M} \lambda \hat{f}^m(x) \tag{6}$$

Boosting Trees: Iteraciones

- Los hiperparámetros a fijar son
  - λ la tasa a la que aprende, los valores típicos son 0.01 o 0.001
  - El tamaño del árbol. Arboles pocos profundos funcionan bien.
  - El número de iteraciones (M) a usar?

Boosting Trees: Iteraciones

- Cuantas iteraciones (M) usar?
  - Cada iteración generalmente reduce el error de ajuste, de modo que para M lo suficientemente grande este error puede hacerse arbitrariamente pequeño (sesgo se va a cero).
  - Sin embargo, ajustar demasiado bien los datos de entrenamiento puede llevar a overfit (sobreajuste)
  - Por lo tanto, hay un número óptimo $M^\star$ que minimiza el error fuera de muestra
  - Una forma conveniente de encontrar $M^\star$ con validación cruzada

Example

photo from https://www.dailydot.com/parsec/batman-1966-labels-tumblr-twitter-vine/