

Árboles (CARTs), Bagging and Random Forests

Big Data y Machine Learning para Economía Aplicada

Ignacio Sarmiento-Barbieri

Universidad de los Andes

Motivación: Recap

- Queremos predecir:

$$Price = f(\text{structural attributes}, \text{amenities}, \dots) \quad (1)$$

- Podemos aplicar linear regression,

$$Price = \beta_0 + \beta_1 \text{Habitaciones} + \beta_2 \text{DCBD} + u \quad (2)$$

- Aplicar OLS a este problema requiere tomar algunas decisiones.

Motivación: Recap



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

Agenda

1 Bagging - Random Forests

2 Boosting

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.

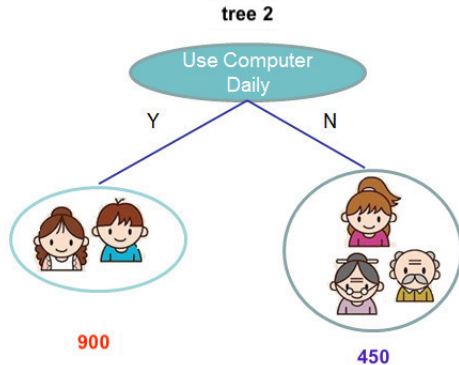
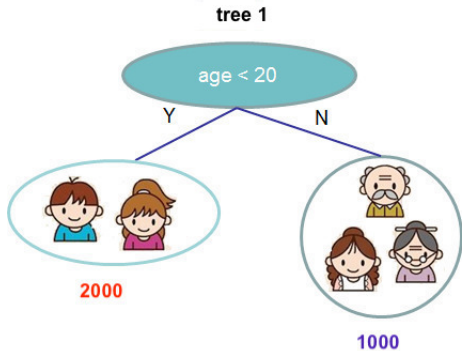
Bagging

- ▶ 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

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

- ▶ Básicamente estamos suavizando las predicciones.

Bagging



$f(\text{boy}) = (2000 + 900)/2 = 1450$ $f(\text{old man}) = (1000 + 450)/2 = 725$

Agenda

- 1 Bagging
 - Random Forests
- 2 Boosting

Random Forests

- ▶ Problema con el bagging: si hay un predictor fuerte, diferentes árboles son muy similares entre sí. Si hay alta correlación, ¿está realmente reduciendo la varianza?
- ▶ Bosques (forests): reduce la correlación entre los árboles en el bootstrap.
- ▶ 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).
- ▶ Tipicamente $m = \sqrt{p}$

Ejemplo



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

Agenda

1 Bagging - Random Forests

2 Boosting

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 → 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 $m = 1, 2, \dots, M$

1 Ajustamos un árbol \hat{f}^m con d bifurcaciones ($d + 1$ hojas)

2 Actualizamos $\hat{f}(x)$ con una versión "shrunk" del nuevo árbol

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

3 Actualizamos los residuales

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

3 El modelo final es

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

Example: Default



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

Boosting Trees: Iteraciones

- ▶ La primer pregunta es sobre 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^* que minimiza el error fuera de muestra
 - ▶ Una forma conveniente de encontrar M^* con validación cruzada

Boosting Trees: Iteraciones

- ▶ Los otros hiperparámetros a fijar son
 - ▶ λ la tasa a la que aprende, los valores típicos son 0.01 o 0.001
 - ▶ El tamaño del árbol. Árboles pequeños (stumps) funcionan bien.

Example: Default



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