Árboles (CARTs), Bagging and Random Forests

Big Data y Machine Learning para Economía Aplicada

Ignacio Sarmiento-Barbieri

Universidad de los Andes

Agenda

Recap

- Queremos predecir y en funcion de observables $(\mathbf{x}_i)$

$$y = f(\mathbf{x}_i) + u \tag{1}$$

- donde la estimación de f implica la que minimize el riesgo empírico (prediga mejor fuera de muestra):

$$\hat{f} = \underset{f}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i; \Theta)) \right\} \tag{2}$$

Recap

- Por ejemplo en regresión $f(x_i) = \mathbf{x}_i\beta$ y minimizamos

$$\hat{\beta} = \underset{\tilde{\beta}}{\text{argmin}} \left\{ \sum_{i=1}^{n} (y_i - \mathbf{x}_i\beta)^2 \right\} \tag{3}$$

- Por ejemplo en clasificación logistica $p(x_i) = \frac{1}{1+e^{-\mathbf{x}_i\beta}}$

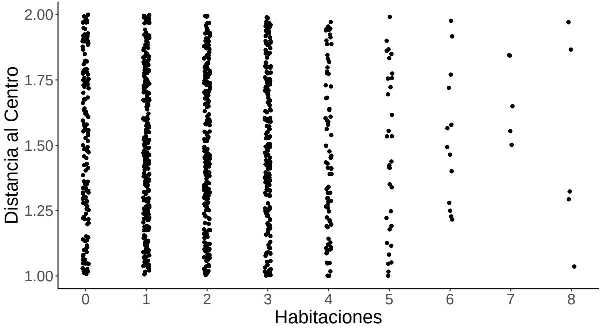
$$\hat{\beta}^{MLE} = \underset{\beta}{argmin} - \left[\sum_{i=1}^{n} log\left(\frac{p_i}{1-p_i}\right)^{y_i} + \sum_{i=1}^{n} log(1-p_i) \right] \tag{4}$$

Agenda

Motivación

- Queremos predecir:

$$Precio = f(habitaciones, Distancia\, al\, CBD) \tag{5}$$



Motivación

- Podemos asumir f es lineal

$$f(\mathbf{x}_i; \beta) = \beta_0 + \beta_1 Habitaciones_i + \beta_2 DCBD_i + u_i \tag{6}$$

- o ajustar un modelo flexible e interpretable como son los arboles

$$f(x_i; \{R_j, \gamma_j\}_1^J) = T(x_i; \{R_j, \gamma_j\}_1^J) \tag{7}$$

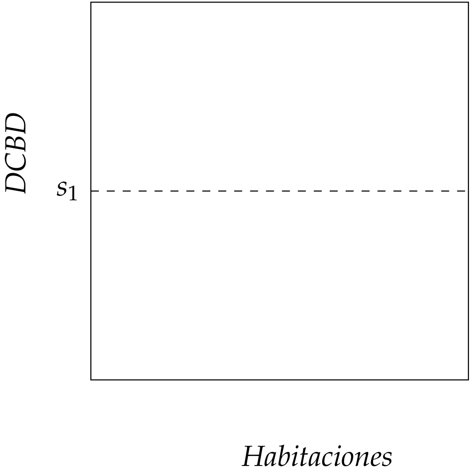
Agenda

¿Qué hacen?

"Recursive binary splitting"

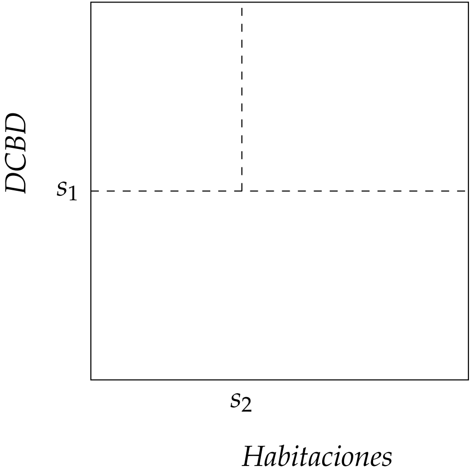
¿Qué hacen?

"Recursive binary splitting"

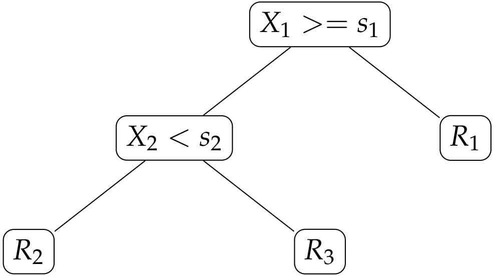


¿Qué hacen?

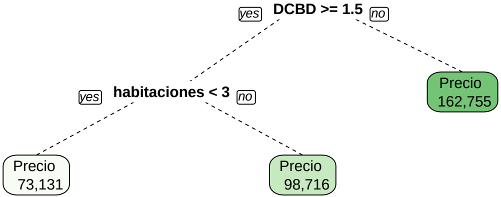
"Recursive binary splitting"

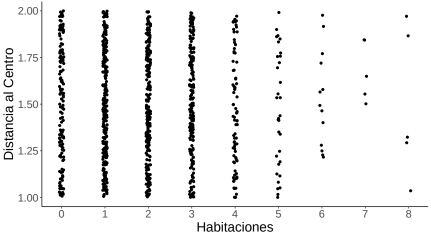


¿Qué hacen?



¿Qué hacen?


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

¿Cómo construimos un árbol?

- El problema de optimización

$$\hat{f} = \underset{f}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i; \Theta)) \right\} \tag{8}$$

- es ahora

$$\{\hat{R}_j, \hat{\gamma}_j\}_1^J = \underset{\{R_j, \gamma_j\}_1^J}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} L(y_i, T(\mathbf{x}_i; \{R_j, \gamma_j\}_1^J)) \right\} \tag{9}$$

¿Cómo construimos un árbol?

- Datos: $y_{n\times 1}$ y $X_{n\times k}$
- Definiciones
  - j es la variable que parte el espacio y s es el punto de partición
  - Defina los siguientes semiplanos

$$R_1(j,s) = \{X|X_j \leq s\} \;\; \& \;\; R_2(j,s) = \{X|X_j > s\} \tag{10}$$

  - El problema: usando una "perdida cuadrática" buscar la variable de partición X_j y el punto s de forma tal que:

$$\underset{j,s}{min}\left[\underset{\gamma_{R_1}}{min}\sum_{x_i\in R_1(j,s)}(y-\gamma_{R_1})^2 + \underset{\gamma_{R_2}}{min}\sum_{x_i\in R_2(j,s)}(y-\gamma_{R_2})^2\right] \tag{11}$$

¿Cómo construimos un árbol?

- ¿Cuál es la solución?

¿Cómo construimos un árbol?

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

Agenda

1. Recap
2. Árboles
   - Árboles de Regresión
   - Árboles de Clasificación
   - Sobreajuste
3. Bagging y Random Forests
   - Bagging
   - Random Forests

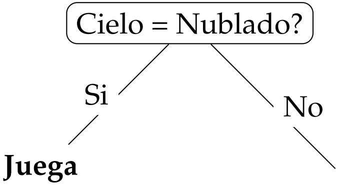
Árboles: Problema

- Jugamos al tenis?

Cielo	Humedad	Tenis?
Sol	Alta	No
Sol	Alta	No
Nublado	Alta	Sí
Sol	Alta	No
Sol	Normal	Sí
Nublado	Alta	Sí
Nublado	Normal	Sí

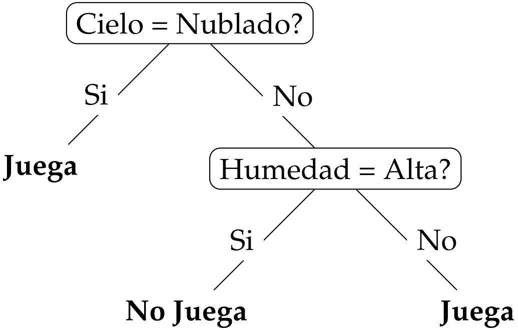
Árboles: Problema

Cielo	Humedad	Tenis?
Sol	Alta	No
Sol	Alta	No
Nublado	Alta	Sí
Sol	Alta	No
Sol	Normal	Sí
Nublado	Alta	Sí
Nublado	Normal	Sí



Árboles: Problema

Cielo	Humedad	Tenis?
Sol	Alta	No
Sol	Alta	No
Nublado	Alta	Sí
Sol	Alta	No
Sol	Normal	Sí
Nublado	Alta	Sí
Nublado	Normal	Sí



¿Cómo construimos un árbol de decisión?

- Regiones lo más “puras” posibles
  - **Regresión**: minima varianza
  - **Clasificación**: ?

¿Cómo construimos un árbol de decisión?

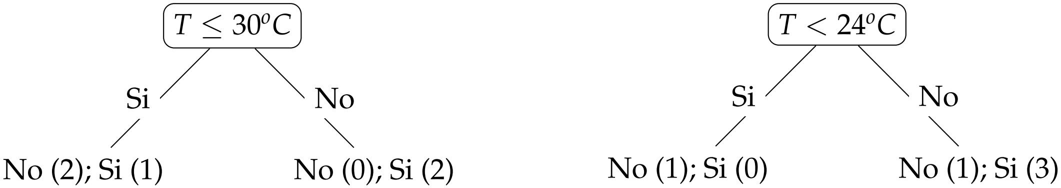
Problemas de clasificación

Temperatura °C	Llovió
23	NO
24	NO
29	SI
31	SI
33	SI

¿Cómo construimos un árbol de decisión?

Problemas de clasificación

- ¿Cuál de los dos cortes es mejor?



¿Cómo construimos un árbol de decisión?

Problemas de clasificación. Medidas de Impureza

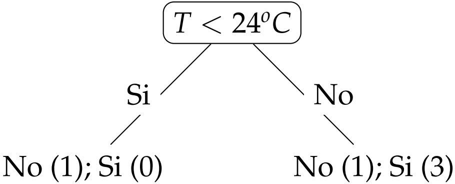
- Medidas de impureza dentro de cada hoja:
  - Índice de Gini : $G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$
  - Entropía : $-\sum_{k=1}^{K} \hat{p}_{mk} log(\hat{p}_{mk})$
- Se define la impureza de un árbol por el promedio ponderado de las impurezas de cada hoja. El ponderador es la fracción de observaciones en cada hoja.

¿Cómo construimos un árbol de decisión?

Problemas de clasificación. Impureza

- ¿Cuál de los dos cortes es mejor?

Temperatura °C	Llovió
31	SI
24	NO
29	SI
33	SI
23	NO

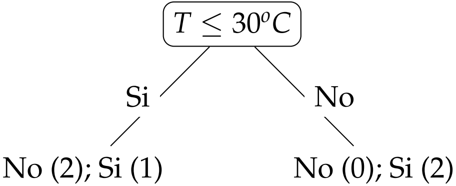


¿Cómo construimos un árbol de decisión?

Problemas de clasificación. Impureza

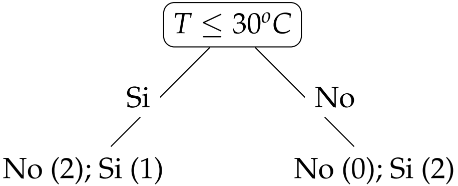
- ¿Cuál de los dos cortes es mejor?

Temperatura °C	Llovió
31	SI
24	NO
29	SI
33	SI
23	NO



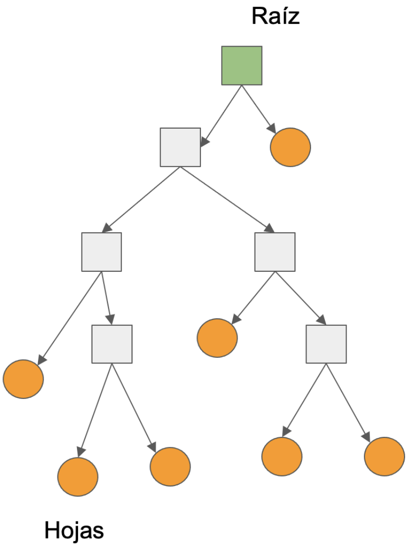
¿Cómo construimos un árbol de decisión?

Problemas de clasificación. Predicción



Agenda

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?

Pruning (poda)

- No es posible calcular el error de predicción usando cross-validation para cada sub-árbol posible
- Solución: *Cost complexity pruning (cortar las ramas mas débiles)*
  - Indexamos los arboles con T.
  - Un sub-árbol $T \in T_0$ es un árbol que se obtuvo colapsando los nodos terminales de otro árbol (cortando ramas).
  - $[T]$ = número de nodos terminales del árbol T

Pruning (poda)

- Cost complexity del árbol T

$$C_\alpha(T) = \sum_{m=1}^{[T]} n_m Q_m(T) + \alpha[T] \tag{12}$$

  - donde $Q_m(T) = \frac{1}{n_m}\sum_{x_i \in R_m}(y_i - \hat{y}_m)^2$ para los árboles de regresión
  - $Q_m(T)$ penaliza la heterogeneidad dentro de la regresión y α el número de regiones
  - Objetivo: para un dado α, encontrar el pruning óptimo que minimice $C_\alpha(T)$

Pruning (poda)

- Mecanismo de búsqueda para T_α (pruning óptimo dado α).

  Resultado: para cada α hay un sub-árbol único T_α que minimiza $C\alpha\ (T)$.

  - Eliminar sucesivamente las ramas que producen un aumento mínimo en $\sum_{m=1}^{[T]} n_m Q_m(T)$
  - Se colapsa hasta el nodo inicial pero va a través de una sucesión de árboles
  - T_α pertenece a esta secuencia. (Breiman et al., 1984)

Pruning (poda)

Algoritmo Completo

1. Utilizamos particiones recursivas binarias para hacer crecer el árbol
2. Para un dado α, aplicamos *cost complexity pruning* al árbol para obtener la secuencia de los subarboles como α.
3. Utilizamos K-fold cross-validation para elegir α.
4. Tenemos entonces una secuencia de subarboles para distintos valores de α
5. Elegimos el α y el subárbol que tienen el menor error de predicción.

Ejemplo

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

Agenda

Agenda

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.

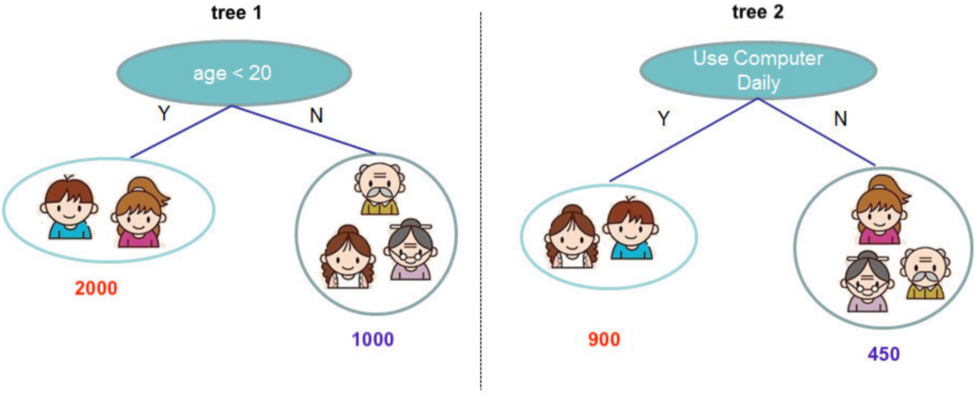
Bagging

- 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

$$\hat{f}_{bag} = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^b(x) \tag{13}$$

- Básicamente estamos suavizando las predicciones.

Bagging



f () = (2000 + 900)/2 = 1450 f () = (1000 + 450)/2 = 725

Bagging

Out-of-Bag Error Estimation

Bagging

Variable Importance Measures

Agenda

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/