Lecture 10: Machine Learning
Boosting

Big Data and Machine Learning en el Mercado Inmobiliario
Educación Continua

Ignacio Sarmiento-Barbieri

Universidad de los Andes

September 16, 2021

Agenda

1. Recap
2. Boosting
   - Motivation
   - AdaBoost
   - Boosting Trees
   - Boosting Trees: Demo
3. Further Readings
4. Break

Recap

- Problema con CART: varianza alta.
- Podemos mejorar mucho el rendimiento mediante la agregación
- Bagging y Arboles:
  - Obtenga repetidamente muestras aleatorias $(X_i^b, Y_i^b)_{i=1}^N$ de la muestra observada, (Bosques $m = \sqrt{p}$)
  - Para cada muestra de arranque, ajuste un árbol de regresión $\hat{f}^b(x)$
  - Promedie las muestras de bootstrap

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

- Básicamente estamos suavizando las predicciones.
- Idea: la varianza del promedio es menor que la de una sola predicción.

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 clasificadores débiles: clasificador marginalmente mejor que lanzar una moneda (tasa de error ligeramente mejor que .5)
- Ej.: CART con pocas ramas (dos ramas)
- Boosting: promedio ponderado de la sucesión de clasificadores débiles.

Boosting

- El boosting viene de la computación y tiene un lenguaje ligeramente diferente.
- Fue desarrollado para problemas de clasificación pero se puede extender fácilmente a regresión.
- Vocabulario:
  - $y \in -1, 1$, X vector de predictores.
  - $y = G(X)$ (clasificador)
  - $err = \frac{1}{N} \sum_i^N I(y_i \neq G(x_i))$
- Para fijar ideas veamos `AdaBoost`

AdaBoost

1. Comenzamos con ponderadores $w_i = 1/N$
2. Para m = 1 hasta M:
   1. Estimar $G_m(x)$ usando ponderadores w_i .
   2. Computar el error de predicción

$$err_m = \frac{\sum_{i=1}^{N} I(y_i \neq G_m(x_i))}{\sum_{i=1}^{N} w_i} \tag{2}$$

   3. Obtener $\alpha_m = ln\left[\frac{(1-errm)}{errm}\right]$
   4. Actualizar los ponderadores : $w_i \leftarrow w_i c_i$

$$c_i = exp\left[\alpha_m I(yi \neq G_m(x_i))\right] \tag{3}$$

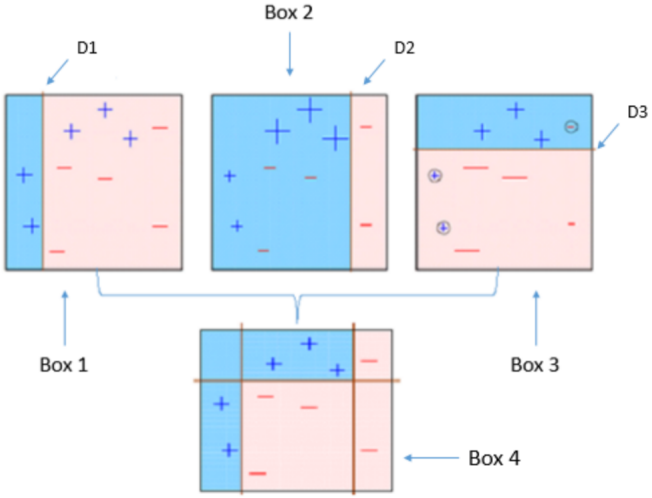
3. Resultado: $G(x) = sgn[\sum_{m=1}^{M} \alpha_m G_m(x)]$

AdaBoost

- $c_i = exp\left[\alpha_m I(y_i \neq G_m(x_i))\right]$
- Si fue correctamente predicho, $c_i = 1$.
- En caso contrario, $c_i = exp(\alpha_m) = \frac{(1-err_m)}{err_m} > 1$
- En cada paso el algoritmo da mas importancia relativa a las predicciones incorrectas.
- Paso final: promedio ponderado de estos pasos

$$G(x) = sgn[\sum_{m=1}^{M} \alpha_m G_m(x)] \tag{4}$$

AdaBoost



Source: https://www.analyticsvidhya.com/blog/2015/11/quick-introduction-boosting-algorithms-machine-learning/

Boosting Trees: Algoritmo

- En el caso de los árboles ($\hat{f}^b(x)$) aprender su estructura es mucho mas difícil
  - Necesitamos saber [T]
  - Es intratable aprender todos los arboles al mismo tiempo
- La estrategia de boosting es aprender iterativamente
- Fijamos lo aprnedido, y agregamos un nuevo arbol en cada paso
- Escribimos el valor de la predicción en cada paso m como $\hat{y}_i^m$.
- Entonces tenemos

$$\begin{aligned} \hat{y}_i^0 &= 0 \\ \hat{y}_i^1 &= \hat{y}_i^0 + f_1(x_i) \\ \hat{y}_i^2 &= \hat{y}_i^1 + f_2(x_i) \\ &\dots \\ \hat{y}_i^M &= \sum_{m=1}^{M} f_m(x_i) = \hat{y}_i^{m-1} + f_m(x_i) \end{aligned} \tag{5}$$

Boosting Trees: Algoritmo

- ¿Qué árbol agregamos en cada paso?
- El que optimice el objetivo

$$obj^m = \sum_{i=1}^{N} L(y_i, \hat{y}_i^{(m)}) \tag{6}$$

$$= \sum_{i=1}^{N} L(y_i, \hat{y}_i^{m-1} + f_m(x_i)) \tag{7}$$

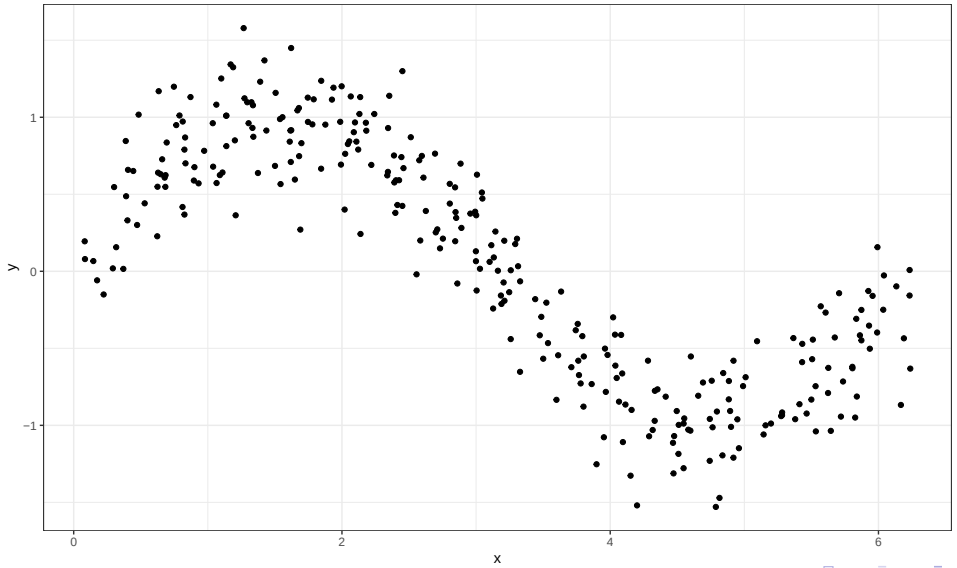
- Si usamos MSE como L la función objetivo es

$$\sum_{i=1}^{N} (y_i - \hat{y}_i^{m-1} + f_m(x_i))^2 \tag{8}$$

Boosting Trees: Iteraciones

- La pregunta se vuelve cuantas iteraciones (M) usar?
  - Cada iteración generalmente reduce el riesgo de entrenamiento $L(.)$, de modo que para M lo suficientemente grande este riesgo 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 estimar $M^\star$ es calcular el error de predicción como una función de M en una muestra de validación. El valor de M que minimiza este riesgo se toma como una estimación de M.

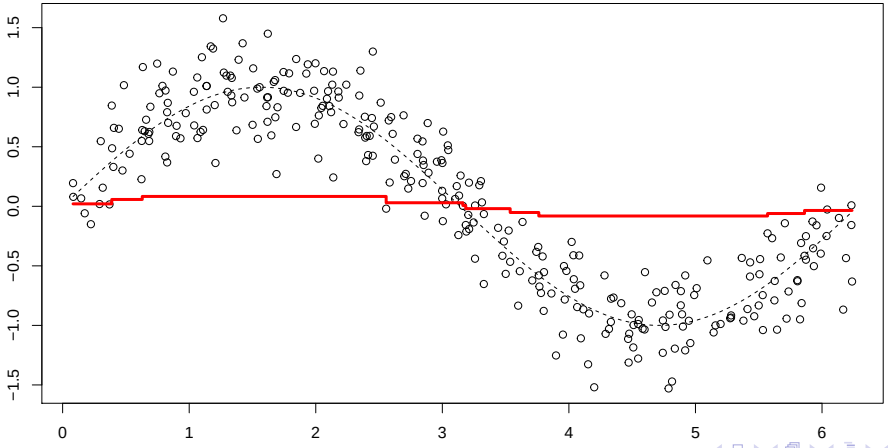
Boosting Trees: Demo



Boosting Trees: Example

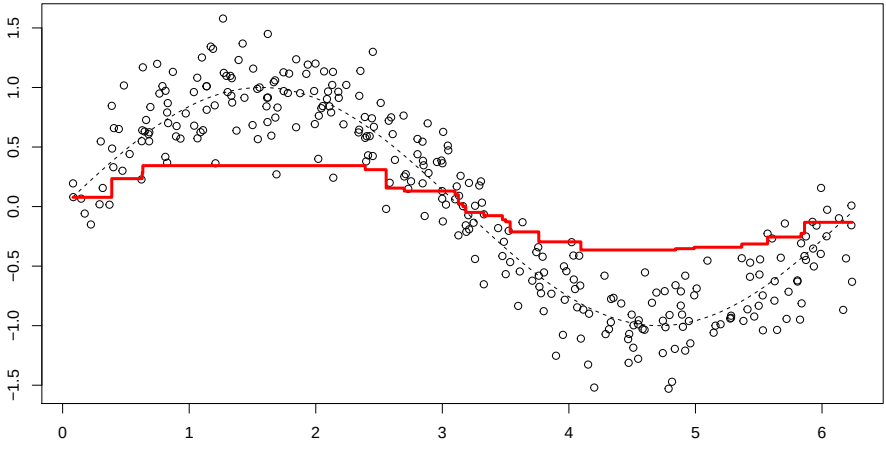
- Algorithm:

```
M<-2
```



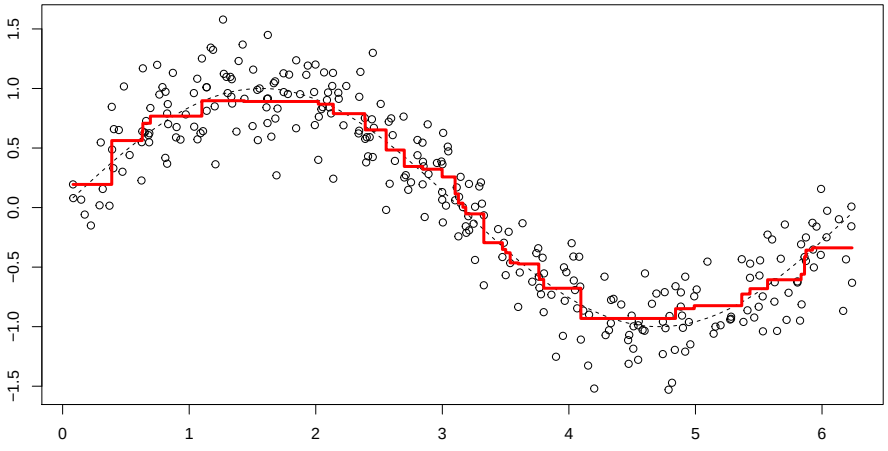
Boosting Trees: Example

```
M<-10
```



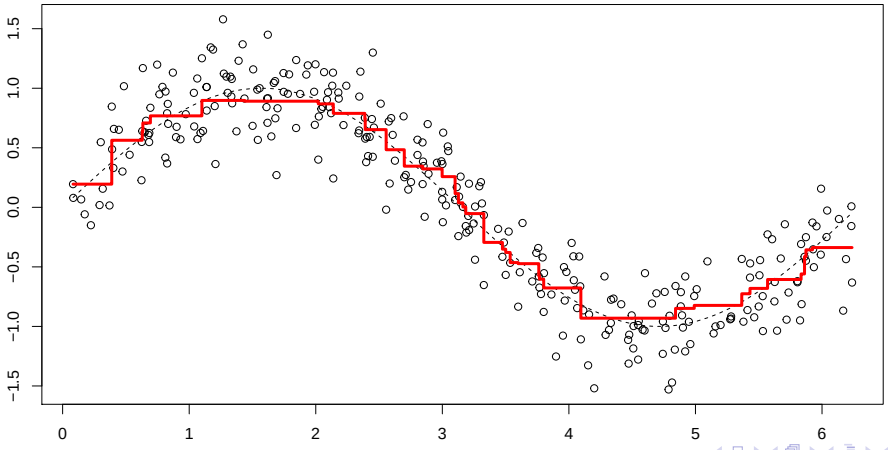
Boosting Trees: Example

```
M<-100
```



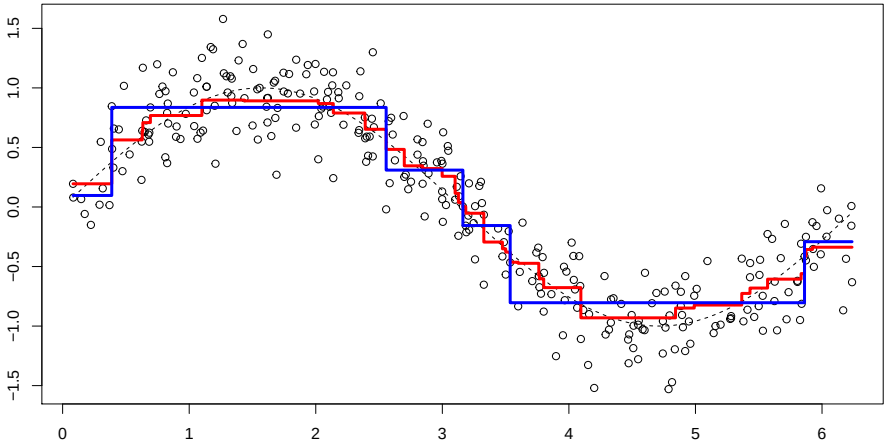
Boosting Trees: Example

```
M<-300
```



Boosting Trees: Example

- Simple tree (blue), boosted tree (red)



XGBoost en un Boosting Tree

- ¿Qué árbol agregamos en cada paso?
- El que optimice la función objetivo

$$\mathcal{L} = \sum_{i=1}^{N} L(y_i, \hat{y}_i) + \sum_{k=1}^{m} \Omega(f_k) \tag{9}$$

- El segundo termino penaliza la complejidad del modelo,

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda ||\omega||_2 \tag{10}$$

XGBoost: palabras finales

- XGBoost es una extensión de boosting trees, hay muchas más.
- Su implementación fue diseñada específicamente para un rendimiento y velocidad óptimos.
- ¿Por qué hablar de XGBoost?
  - Entre las 29 soluciones ganadoras del desafío publicadas en página de Kaggle durante 2015, 17 soluciones utilizaron XGBoost.
  - Entre estas soluciones, ocho utilizaron exclusivamente XGBoost para entrenar el modelo, mientras que la mayoría de las demás combinaron XGBoost con redes neuronales. (El segundo método más popular, las redes neuronales profundas, se utilizó en 11 soluciones)
  - También se vio en la competencia de Minería de Datos y Descubrimiento de Conocimiento de 2015 organizada por ACM (Copa KDD), donde XGBoost fue utilizado por todos los equipos ganadores en el top-10.
  - Históricamente, XGBoost ha funcionado bastante bien para datos tabulares estructurados. Pero, si se trata de datos no estructurados como imágenes, las redes neuronales suelen ser una mejor opción.

Further Readings

- Charpentier, Arthur (2018). Classification from scratch, boosting. https://freakonometrics.hypotheses.org/tag/xgboost
- Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).
- Chen, T., He, T., & Benesty, M. (2018). XGBoost Documentation.
- Friedman, J., Hastie, T., & Tibshirani, R. (2001). The elements of statistical learning (Vol. 1, No. 10). New York: Springer series in statistics.
- Taddy, M. (2019). Business data science: Combining machine learning and economics to optimize, automate, and accelerate business decisions. McGraw Hill Professional.


¡Gracias Totales!

Volvemos en 5 min con Python