Lecture 22: Bagging, Random Forests, & Boosting

Big Data and Machine Learning for Applied Economics
Econ 4676

Ignacio Sarmiento-Barbieri

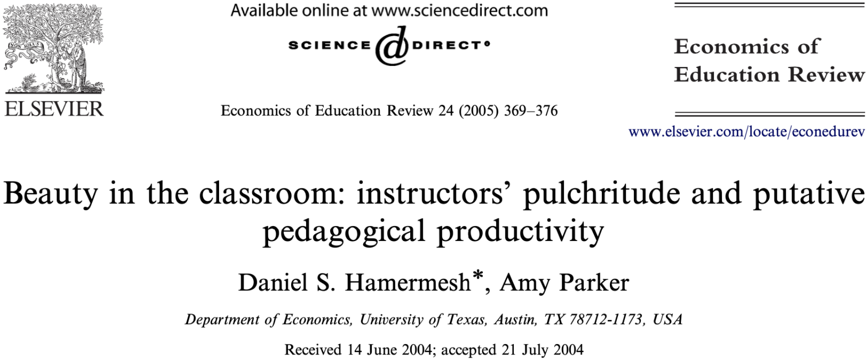
Universidad de los Andes

October 29, 2020

Agenda

1. Recap w. Example: Beauty in the classroom
2. Bagging and Random Forests
3. Boosting
   - AdaBoost
4. Review & Next Steps
5. Further Readings

CART: Recap w. Example: Beauty in the classroom



Available online at www.sciencedirect.com

SCIENCE DIRECT®

Economics of Education Review 24 (2005) 369–376

ELSEVIER

Economics of Education Review

www.elsevier.com/locate/econedurev

Beauty in the classroom: instructors' pulchritude and putative pedagogical productivity

Daniel S. Hamermesh*, Amy Parker

Department of Economics, University of Texas, Austin, TX 78712-1173, USA

Received 14 June 2004; accepted 21 July 2004

CART: Recap w. Example: Beauty in the classroom

Motivation

- An immense literature in social psychology (summarized by Hatfield and Sprecher, 1986) has examined the impact of human beauty on a variety of noneconomic outcomes.
- Economists have considered how beauty affects labor market outcomes, particularly earnings (Hamermesh and Biddle, 1994; Biddle and Hamermesh, 1998).
- The impacts on these monetary outcomes are implicitly the end results of the effects of beauty on productivity;
- But there seems to be no direct evidence of the impacts of beauty on productivity in a context in which we can be fairly sure that productivity generates economic rewards.

CART: Recap w. Example: Beauty in the classroom

Motivation

- A substantial amount of research has indicated that academic administrators pay attention to teaching quality in setting salaries (Becker and Watts, 1999).
- The question is what generates the measured productivity for which the economic rewards are being offered.
- One possibility is simply that descriptive characteristics, such as beauty, trigger positive responses by students and lead them to evaluate some teachers more favorably, so that their beauty earns them higher economic returns.

CART: Recap w. Example: Beauty in the classroom

Motivation

- They take sample of student instructional ratings for a group of university teachers and acquire six independent measures of their beauty, and a number of other descriptors of them and their classes.

```
data("TeachingRatings", package = "AER")
tr <- subset(TeachingRatings, credits == "more")
str(tr)
```

```
## ’data.frame’:    436 obs. of  12 variables:
##  $ minority   : Factor w/ 2 levels "no","yes": 2 1 1 1 1 1 1 1 1 1 ...
##  $ age        : int  36 59 51 40 31 62 33 51 33 47 ...
##  $ gender     : Factor w/ 2 levels "male","female": 2 1 1 2 2 1 2 2 2 1 ...
##  $ credits    : Factor w/ 2 levels "more","single": 1 1 1 1 1 1 1 1 1 1 ...
##  $ beauty     : num  0.29 -0.738 -0.572 -0.678 1.51 ...
##  $ eval       : num  4.3 4.5 3.7 4.3 4.4 4.2 4 3.4 4.5 3.9 ...
##  $ division   : Factor w/ 2 levels "upper","lower": 1 1 1 1 1 1 1 1 1 1 ...
##  $ native     : Factor w/ 2 levels "yes","no": 1 1 1 1 1 1 1 1 1 1 ...
##  $ tenure     : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 1 ...
##  $ students   : num  24 17 55 40 42 182 33 25 48 16 ...
##  $ allstudents: num  43 20 55 46 48 282 41 41 60 19 ...
##  $ prof       : Factor w/ 94 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 9 10 ...
```

CART: Recap w. Example: Beauty in the classroom

OLS

```r
tr_lm <- lm(eval ~ beauty + gender + minority + native + tenure + division,
  data = tr, weights = students)

tr_lm_gender <- lm(eval ~ beauty:gender + minority + native + tenure + division,
  data = tr, weights = students)

tr_lm_male <- lm(eval ~ beauty + minority + native + tenure + division,
  data = tr[tr$gender=="male",], weights = students)

tr_lm_female <- lm(eval ~ beauty + minority + native + tenure + division,
  data = tr[tr$gender=="female",], weights = students)

stargazer::stargazer(tr_lm,tr_lm_gender,tr_lm_male,tr_lm_female,type="text")
```

CART: Recap w. Example: Beauty in the classroom

OLS

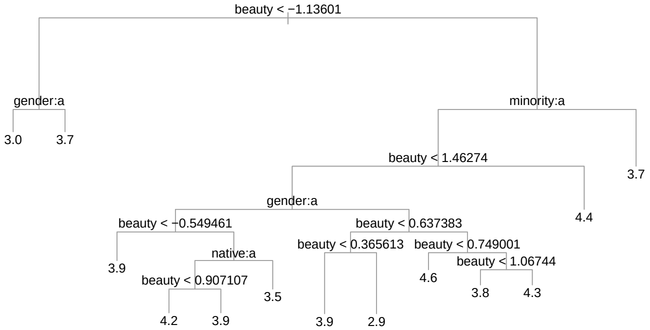
```
## ===================================================================================================================
##                                                           Dependent variable:
##                     -----------------------------------------------------------------------------------------------
##                                                                  eval
##                               (1)                     (2)                     (3)                    (4)
## -------------------------------------------------------------------------------------------------------------------
## beauty                     0.283***                                        0.383***                0.133***
##                             (0.028)                                         (0.037)                (0.045)
## genderfemale               -0.213***
##                             (0.048)
## minorityyes                -0.327***               -0.363***                -0.014                -0.279***
##                             (0.086)                 (0.084)                 (0.194)                (0.098)
## nativeno                    -0.217*                 -0.229*                -0.388**                -0.288
##                             (0.125)                 (0.124)                 (0.188)                (0.175)
## tenureyes                  -0.132**                 -0.032                  -0.053                  -0.064
##                             (0.065)                 (0.065)                 (0.088)                (0.098)
## divisionlower               -0.050                  -0.071                   0.004                -0.244***
##                             (0.044)                 (0.045)                 (0.054)                (0.078)
## beauty:gendermale                                  0.407***
##                                                     (0.038)
## beauty:genderfemale                                0.132***
##                                                     (0.043)
## Constant                   4.216***                4.058***                4.101***                4.027***
##                             (0.068)                 (0.062)                 (0.089)                (0.084)
##
## -------------------------------------------------------------------------------------------------------------------
## Observations                  436                     436                     250                    186
## R2                           0.271                   0.275                   0.335                  0.170
## ===================================================================================================================
## Note:                                                                                   *p<0.1; **p<0.05; ***p<0.01
```

CART: Recap w. Example: Beauty in the classroom

Trees

```
library("tree")

pstree <- tree(eval ~beauty + gender + minority + native + tenure + division, data=tr, mincut=1)
par(mfrow=c(1,1))
plot(pstree, col=8)
text(pstree, digits=2)
```



CART: Recap w. Example: Beauty in the classroom

Trees: Constructing the partition

- How to choose the partition?
- Start with the trivial partition with one element
- Greedy algorithm (CART): Iteratively split an element of the partition, such that the in-sample prediction improves as much as possible.
- That is: Given $(R_1 \ldots, R_M)$
  - For each R_m $m = 1, \ldots, M$ and
  - For each X_j $j = 1, \ldots, p$
  - find the $s_{j,m}$ that minimizes the mean squared error, if we split R_m along variable X_j at $s_{j,m}$
  - then pick the (m, j) that minimizes the MSE and construct a new partition with $M + 1$ elements
  - Iterate

CART: Recap w. Example: Beauty in the classroom

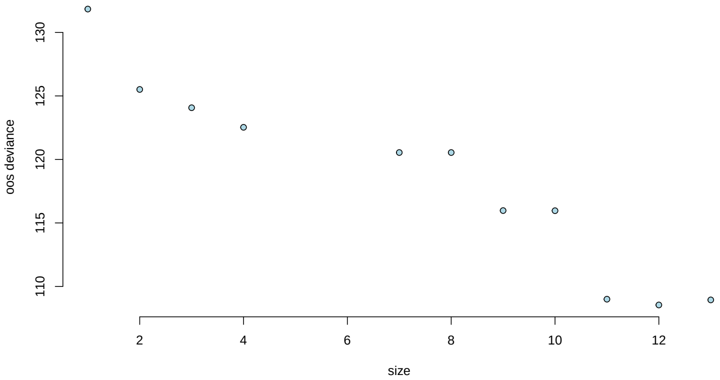
Trees: Tuning and pruning

- Key tuning parameter: Total number of splits M.
- We can optimize this via cross-validation.
- CART can furthermore be improved using “pruning.”
- Idea:
  - Fit a flexible tree (with large M) using CART.
  - Then iteratively remove (collapse) nodes.
  - To minimize the sum of squared errors,
  - plus a penalty for the number of elements in the partition.
- This improves upon greedy search. It yields smaller trees for the same mean squared error.

CART: Recap w. Example: Beauty in the classroom

Trees

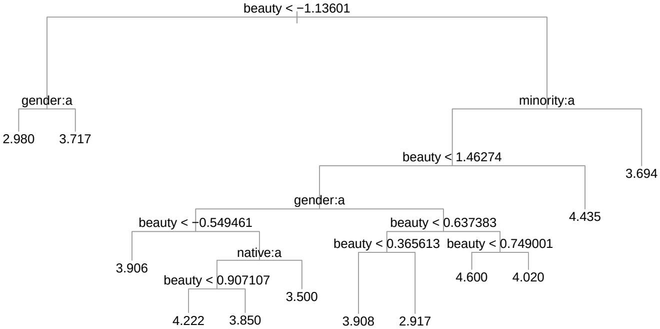
```
## Use cross-validation to prune the tree
cvpst <- cv.tree(pstree, K=10)
plot(cvpst$size, cvpst$dev, xlab="size", ylab="oos deviance", pch=21, bg="lightblue", bty="n")
```



CART: Recap w. Example: Beauty in the classroom

Trees

```r
pstcut <- prune.tree(pstree, best=12)
plot(pstcut, col=8)
text(pstcut)
```



CARTs

- Smart way to represent nonlinearities. Most relevant variables on top.
- Very easy to communicate.
- Reproduces human decision-making process.
- Trees are intuitive and do OK, but
    - They are not very good at prediction
    - If the structure is linear, CART does not work well.
    - Not very robust

Bagging

- Problem with CART: high variance.
- We can improve performance a lot using either bootstrap aggregation (bagging), random forests, or boosting.
- Bagging:
  - Repeatedly draw bootstrap samples $(X_i^b, Y_i^b)_{i=1}^N$ from the observed sample.
  - For each bootstrap sample, fit a regression tree $\hat{f}^b(x)$
  - Average across bootstrap samples to get the predictor

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

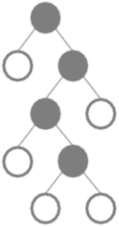
  - Basically we are smoothing predictions.
  - Idea: the variance of the average is less than that of a single prediction.

Random Forests

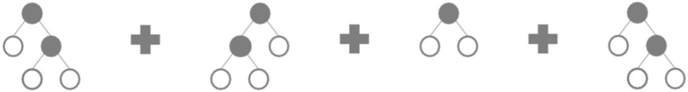
- Problem with bagging: if there is a strong predictor, different trees are very similar to each other. High correlation. Is the variance really reduced?
- Forests: lower the correlation between the trees in the boostrap.
- If there are p predictors, in each partition use only $m < p$ predictors, chosen randomly
- Bagging is forest with $m = p$ (use all predictors in each partition).
- Typically $m = \sqrt{(}p)$

Random Forests

Trees:



Random Forests:



Random Forests

```
require("ranger")

rf_tree <- ranger(eval ~beauty + gender + minority + native +
         tenure + division, data=tr,
        write.forest=TRUE,num.tree=200,min.node.size=25,
        importance="impurity")
sort(rf_tree$variable.importance, decreasing = TRUE)
```

```
##    beauty  minority    gender    native    tenure  division
## 22.881176  3.089366  2.608295  2.104095  2.062075  1.627261
```

Boosting

- Problem with CART: high variance. Instability
- Weak classifier: marginally better classifier than flipping a coin (error rate slightly better than .5)
- E.g.: CART with few branches ('stump', two branches)
- Boosting: weighted average of a succession of weak classifiers.
- Vocab
  - $y \in -1, 1$ (for simplicity), X vector of predictors.
  - $y = G(X)$ (classifier)
  - $err = \frac{1}{N} \sum_i^N I(y_i \neq G(x_i))$

AdaBoost

1. Start with weights $w_i = 1/N$
2. For m = 1 through M:
    1. Adjust $G_m(x)$ using weights w_i .
    2. Compute prediction

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

    3. Compute $\alpha_m = ln\left[\frac{(1-errm)}{errm}\right]$
    4. Update weights: $w_i \leftarrow w_i c_i$

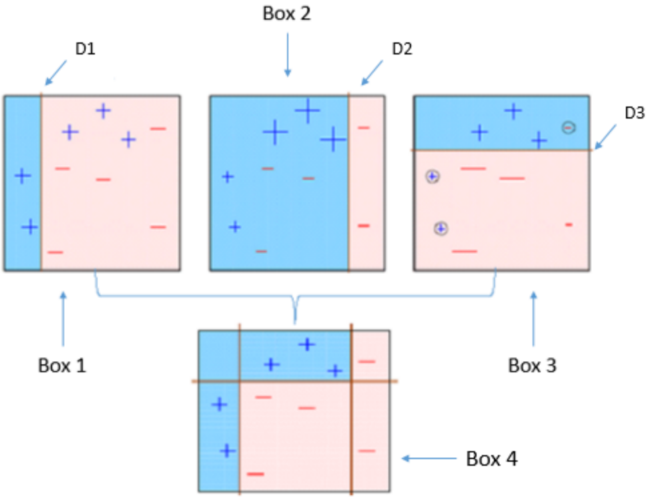
$$c_i = exp\left[\alpha_m I(yi \neq G_m(x_i))\right] \tag{3}$$

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

AdaBoost

- $c_i = exp\left[\alpha_m I(y_i \neq G_m(x_i))\right]$
- If it was correctly predicted, $c_i = 1$. No issue.
- Otherwise, $c_i = exp(\alpha_m) = \frac{(1-err_m)}{err_m} > 1$
- At each step the method gives more relative importance to the predictions that where wrong.
- Final step: weighted average of predictions at each step.

AdaBoost



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

Review & Next Steps

- Recap w. Example: Beauty in the classroom
- Bagging and Random Forests
- Boosting AdaBoost
- Next class: More on trees, and causal inference
- Questions? Questions about software?

Further Readings

- Breiman, L. (2001). “Random Forests”. In: Machine Learning. ISSN: 1098-6596. DOI: 10.1017/CBO9781107415324.004. eprint: arXiv:1011.1669v3.
- Friedman, J., Hastie, T., & Tibshirani, R. (2001). The elements of statistical learning (Vol. 1, No. 10). New York: Springer series in statistics.
- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (Vol. 112, p. 18). New York: springer.
- Kasy M. (2019). Trees, forests, and causal trees. Mimeo.
- Taddy, M. (2019). Business data science: Combining machine learning and economics to optimize, automate, and accelerate business decisions. McGraw Hill Professional.