THE UNIVERSITY OF CHICAGO

STAT 37710 / CMSC 35400 / CAAM 37710
Machine Learning

Bagging & Random Forests

Cong Ma

Recall: decision trees

- Decision Trees are
  - low bias, high variance models
    - Unless you regularize a lot…
    - …but then often worse than Linear Models
  - highly non-linear
    - Can easily overfit
    - Different training samples can lead to very different trees

$$\underbrace{\mathbb{E}_D\left[\left(y-\hat{h}_D(\mathbf{x})\right)^2\right]}_{\text{expected error}} = \underbrace{\mathbb{E}_D\left[\hat{h}_D(\mathbf{x})-y\right]^2}_{\text{bias}} + \underbrace{\mathbb{E}_D\left[\left(\hat{h}_D(\mathbf{x})-\mathbb{E}_{D'}\hat{h}_{D'}(\mathbf{x})\right)^2\right]}_{\text{variance}}$$

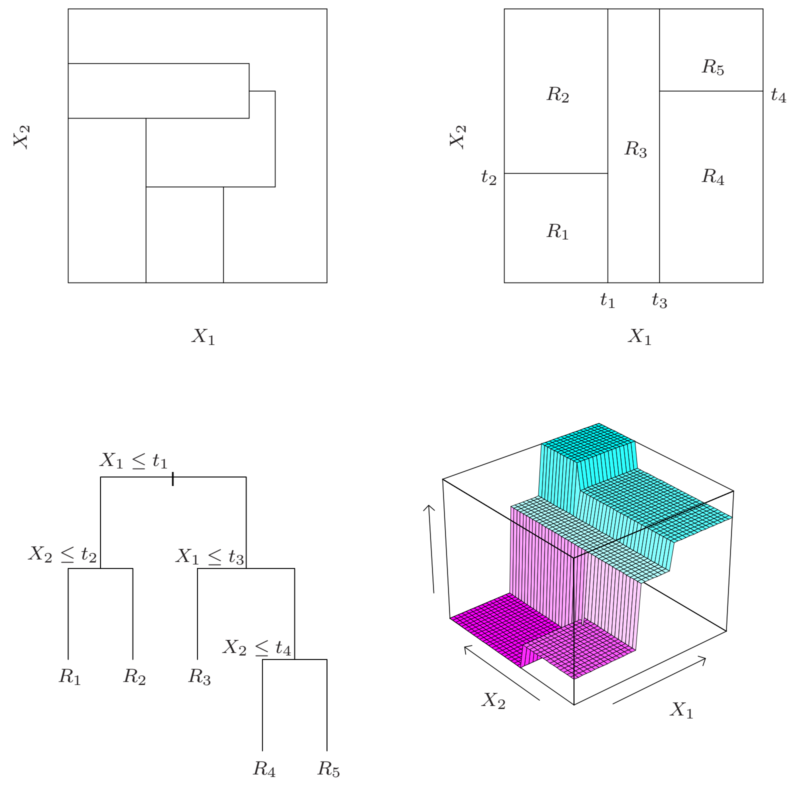


FIGURE 9.2. *Partitions and CART. Top right panel shows a partition of a two-dimensional feature space by recursive binary splitting, as used in CART, applied to some fake data. Top left panel shows a general partition that cannot be obtained from recursive binary splitting. Bottom left panel shows the tree corresponding to the partition in the top right panel, and a perspective plot of the prediction surface appears in the bottom right panel.*

How to improve decision trees?

- What’s the problem of decision tree?
  - Low bias but high variance
- We’d like to keep the low bias, but decrease the variance
  - Key idea: build multiple trees and take the average
  - We know averaging reduces variance (Caveat!)

Average over multiple different datasets

- Goal: reduces variance
- Ideal setting:
  - many training sets D’
    - sample independently
  - train model using each D’
  - average predictions

P(x,y)

Person	Age	Male?	Height > 55”
James	11	1	1
Jessica	14	0	1
Alice	14	0	1
Amy	12	0	1
Bob	10	1	1
Xavier	9	1	0
Cathy	9	0	1
Carol	13	0	1
Eugene	13	1	0
Rafael	12	1	1
Dave	8	1	0
Peter	9	1	0
Henry	13	1	0
Erin	11	0	0
Rose	7	0	0
Iain	8	1	1
Paulo	12	1	0
Frank	9	1	1
Jill	13	0	0
Leon	10	1	0
Sarah	12	0	0
Gena	8	0	0
Patrick	5	1	1

D’

Person	Age	Male?	Height > 55”
Alice	14	0	1
Bob	10	1	1
Carol	13	0	1
Dave	8	1	0
Erin	11	0	0
Frank	9	1	1
Gena	8	0	0

“Bagging Predictors” [Leo Breiman, 1994]

http://statistics.berkeley.edu/sites/default/files/tech-reports/421.pdf

Bagging

- Goal: reduces variance
- In practice:
  - fixed training set D
    - Resample D’ with replacement from D
  - train model using each D’
  - average predictions

D

Person	Age	Male?	Height > 55”
James	11	1	1
Jessica	14	0	1
Alice	14	0	1
Amy	12	0	1
Bob	10	1	1
Xavier	9	1	0
Cathy	9	0	1
Carol	13	0	1
Eugene	13	1	0
Rafael	12	1	1
Dave	8	1	0
Peter	9	1	0
Henry	13	1	0
Erin	11	0	0
Rose	7	0	0
Iain	8	1	1
Paulo	12	1	0
Frank	9	1	1
Jill	13	0	0
Leon	10	1	0
Sarah	12	0	0
Gena	8	0	0
Patrick	5	1	1

D’

Person	Age	Male?	Height > 55”
Alice	14	0	1
Bob	10	1	1
Carol	13	0	1
Dave	8	1	0
Erin	11	0	0
Frank	9	1	1
Gena	8	0	0

“Bagging Predictors” [Leo Breiman, 1994]
http://statistics.berkeley.edu/sites/default/files/tech-reports/421.pdf

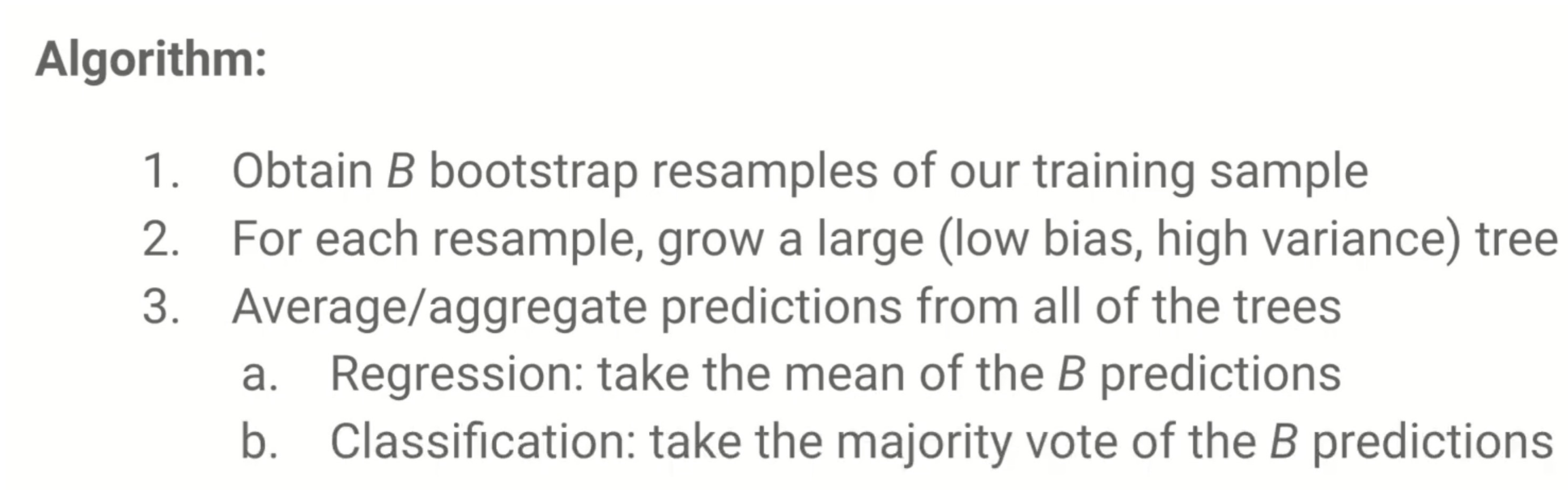
Bagging = Bootstrap Aggregating

- Learns a predictor by aggregating the predictors learned over multiple random draws (bootstrap samples) from the training data
  - A bootstrap sample of size m from $D : \{(\mathbf{x}_i, y_i), i = 1, \ldots, n\}$ is

$$\{(\mathbf{x}'_i, y'_i), i = 1, \ldots, m\}$$

where each (x_i', y_i') is drawn uniformly at random from D (with replacement)

Bagged trees



Algorithm:

1. Obtain *B* bootstrap resamples of our training sample
2. For each resample, grow a large (low bias, high variance) tree
3. Average/aggregate predictions from all of the trees
    a. Regression: take the mean of the *B* predictions
    b. Classification: take the majority vote of the *B* predictions

Aggregating weak predictors

- Imagine we have a model we can fit to the training data to produce a predictor that we use to predict $E(Y|X=x)$
  - E.g. a decision tree or logistic regression

- With bagging, we
  - compute B different bootstrap samples
  - learn a predictor for each one
  - aggregate the predictors to form the target predictor

Bootstrap

- Assume you have a sample $X_1, ..., X_n$ of points and, say, an estimate $\hat{\Theta}$ of a true parameter Θ of this population. You would like to know the distribution of the estimate $\hat{\Theta}$ (for example, because you want to construct confidence sets).
- You now draw a subsample of m points of the original sample (with our without replacement), and on this subsample you compute an estimate of the parameter you are interested in.
- You repeat this procedure B times, resulting in B bootstrap estimates $\hat{\Theta}_1, ..., \hat{\Theta}_B$.
- This set now gives an "indication" about how your estimate is distributed, and you can compute its mean, its variance, confidence sets, etc.

Bagging

- As in bootstrap, you generate B bootstrap samples of your original sample, and on each of them compute the estimate you are interested in: $\hat{\Theta}_1, ..., \hat{\Theta}_B$
- As your final estimate, you then take the average: $\hat{\Theta}_{bag} = mean(\hat{\Theta}_1, ..., \hat{\Theta}_B)$.
- The advantage of this procedure is that the estimate Θ_{bag} can have a much smaller variance than each of the individual estimates $\hat{\Theta}_b$:
  - If the estimates $\hat{\Theta}_b$ were i.i.d. with variance σ^2, then the variance of $\hat{\Theta}_{bag}$ would be σ^2/B .
  - If the estimates are identically distributed but have a (hopefully small) positive pairwise correlation ρ, then the variance of $\hat{\Theta}_{bag}$ is $\rho\sigma^2 + (1-\rho)\frac{\sigma^2}{B}$. If ρ is small and B is large, this is good.

Decorrelate the trees

- Key: we’d like “diversity” in the trees we build, or further decorrelate the trees we build
- Use random features in splitting the nodes!

Random Forests

- Goal: reduce variance
  - Bagging can only do so much
  - Resampling training data

- Random Forests: sample data & features!
  - Sample S’
  - Train DT
    - At each node, sample features
  - Average predictions

Random Forests

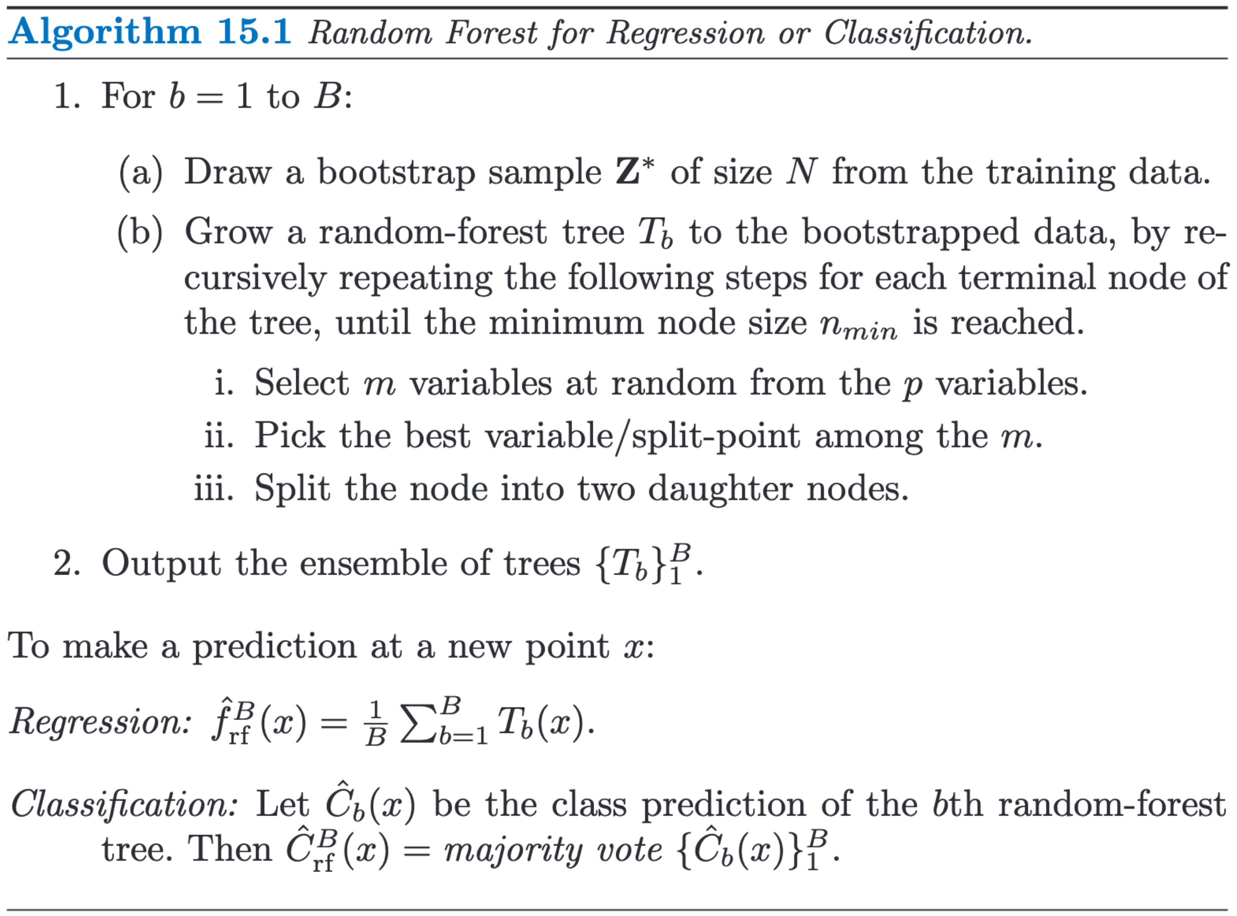
- Extension of bagging to sampling features

- Generate bootstrap D’ from D
  - Train DT top-down on D’
  - Each node, sample subset of features for splitting
    - Can also sample a subset of splits as well

- Average predictions of all DTs

“Random Forests – Random Features” [Leo Breiman, 1997]
http://oz.berkeley.edu/~breiman/random-forests.pdf

Algorithm for random forest



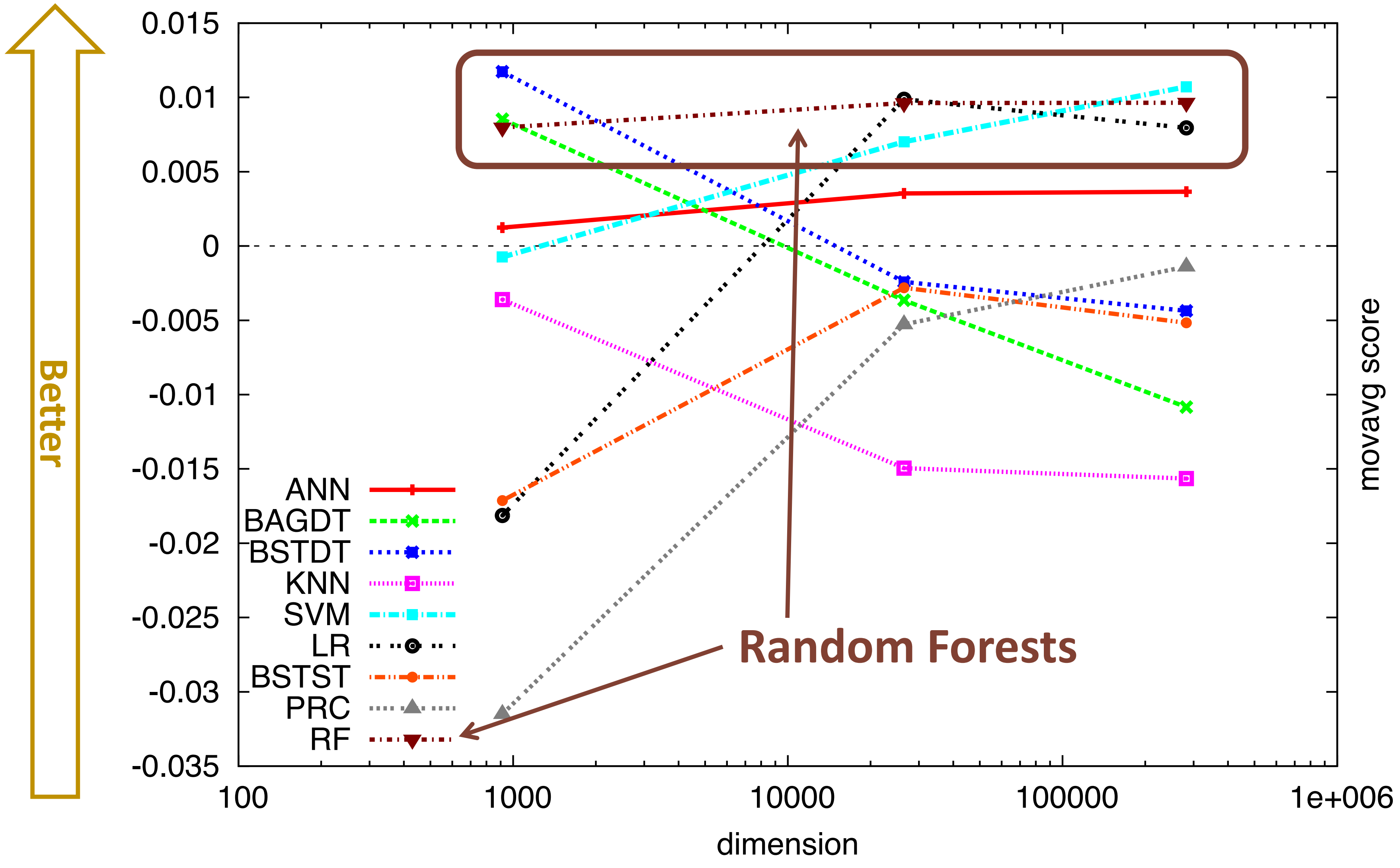
Algorithm 15.1 *Random Forest for Regression or Classification.*

```
1. For $b = 1$ to $B$:
    (a) Draw a bootstrap sample $\mathbf{Z}^*$ of size $N$ from the training data.
    (b) Grow a random-forest tree $T_b$ to the bootstrapped data, by re-
        cursively repeating the following steps for each terminal node of
        the tree, until the minimum node size $n_{min}$ is reached.
          i. Select $m$ variables at random from the $p$ variables.
         ii. Pick the best variable/split-point among the $m$.
        iii. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^B$.
```

To make a prediction at a new point x:

Regression: $\hat{f}_{\text{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^B T_b(x)$.

Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree. Then $\hat{C}_{\text{rf}}^B(x) = \textit{majority vote}\ \{\hat{C}_b(x)\}_1^B$.



Average performance over many datasets
Random Forests perform the best

"An Empirical Evaluation of Supervised Learning in High Dimensions"
Caruana, Karampatziakis & Yessenalina, ICML 2008

References & acknowledgement

- Hastie et al. (2009). “The Elements of Statistical Learning”
  - Ch 8.7 , “Bagging”
- Willett & Chen (2020). “CMSC 35400: Machine Learning”
- Yue (2018). “Machine Learning & Data Mining”
  - Lecture 5, “Decision Trees, Bagging & Random Forests”
- Breiman (1994). “Bagging Predictors”
- Breiman (1997). “Random Forests – Random Features”