LECTURE 19:

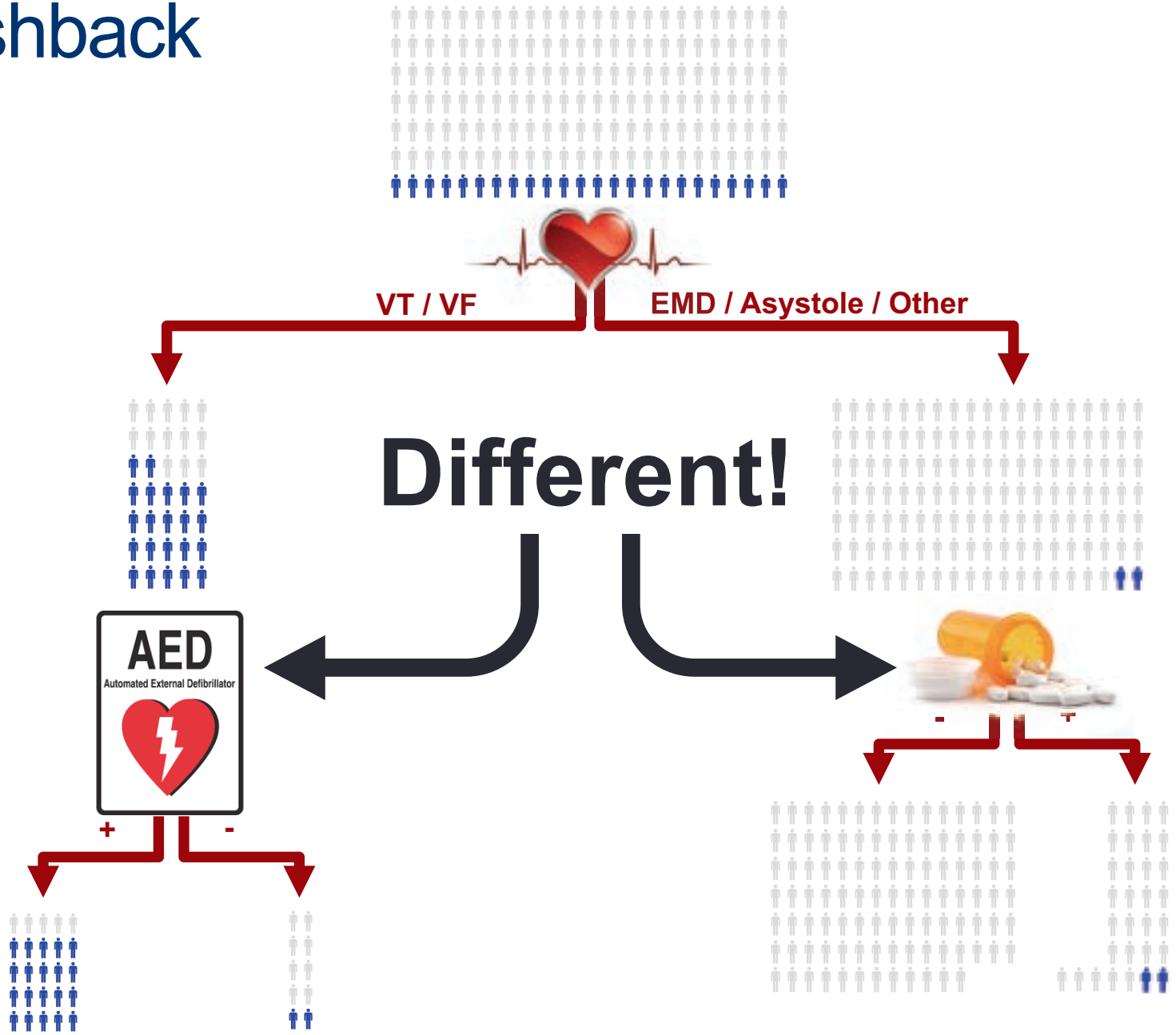
TREE-BASED METHODS PT. 2

November 15, 2017
SDS 293: Machine Learning

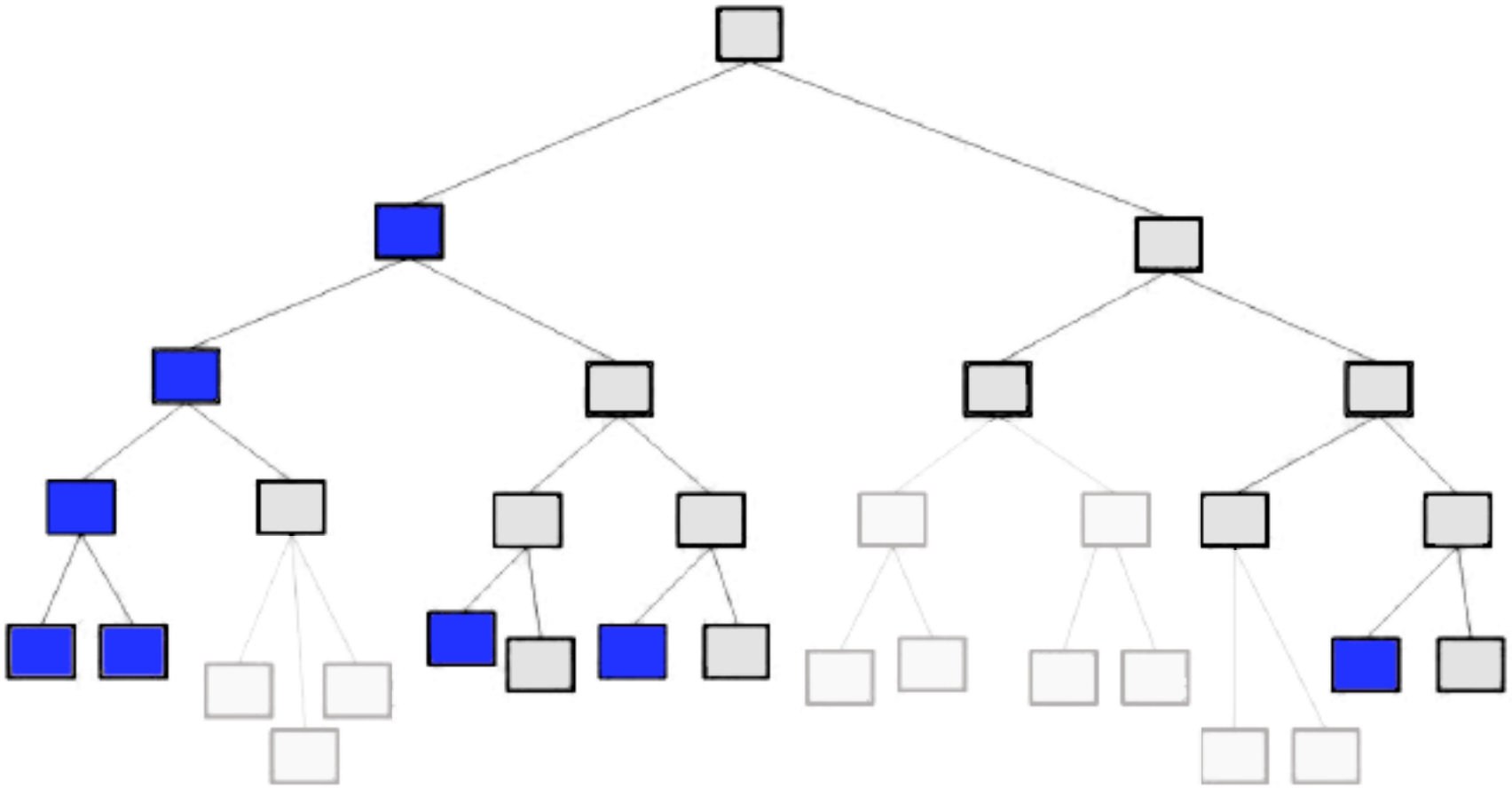
Outline

- ✓ Basic mechanics of tree-based methods
  - ✓ Classification example
  - ✓ Choosing good splits
  - ✓ Pruning
- How to avoid over-fitting
  - Bootstrap aggregation (“bagging”)
  - Random forests
  - Boosting
- Lab

Flashback


VT / VF
EMD / Asystole / Other
Different!
AED
Automated External Defibrillator
+
-
-
+

Flashback



Problem: high variance

Discussion

- **Question:** what can we do to combat high variance?
- **Answer:** bootstrap and aggregate!

Bagging

- **Big idea:** use regular old bootstrapping to generate a bunch of sample training sets, build trees for each one, and **average** their resulting predictions

Image courtesy Rachopin77 on DeviantArt

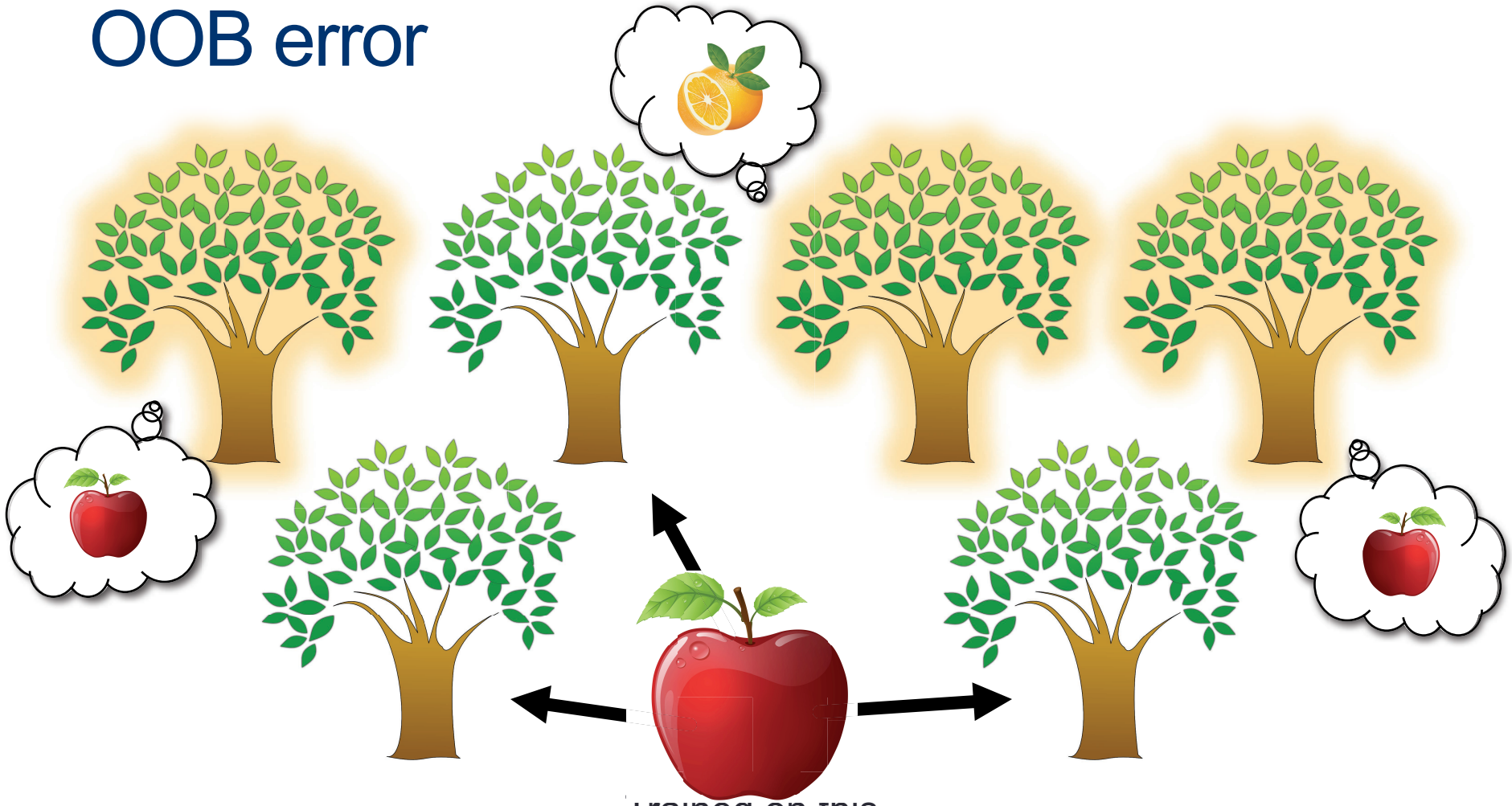
Estimating test error

- **Fun fact**: there is a straightforward way to estimate the test error of a bagged model, without needing to use a test set or cross-validate!

- **Key**: trees are repeatedly fit to bootstrapped subsets
  - Each bagged tree only trains on ~2/3 of the observations
  - The remaining ~1/3 are called the **out-of-bag (OOB) observations**

- **Question**: how does this help?

OOB error



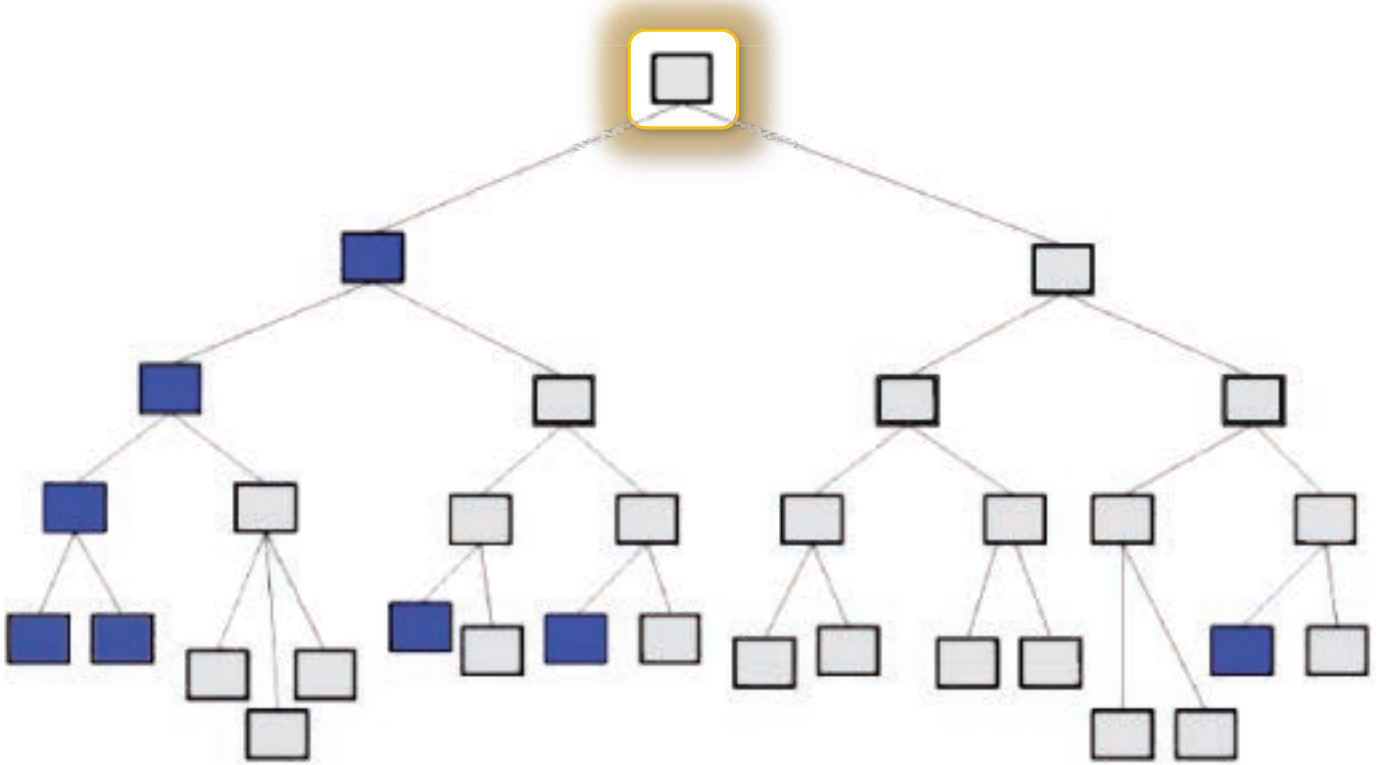
Trained on this observation?

Repeat for **every observation**, average to get MSE or classification error

*with enough trees, this is essentially equivalent to LOOCV error

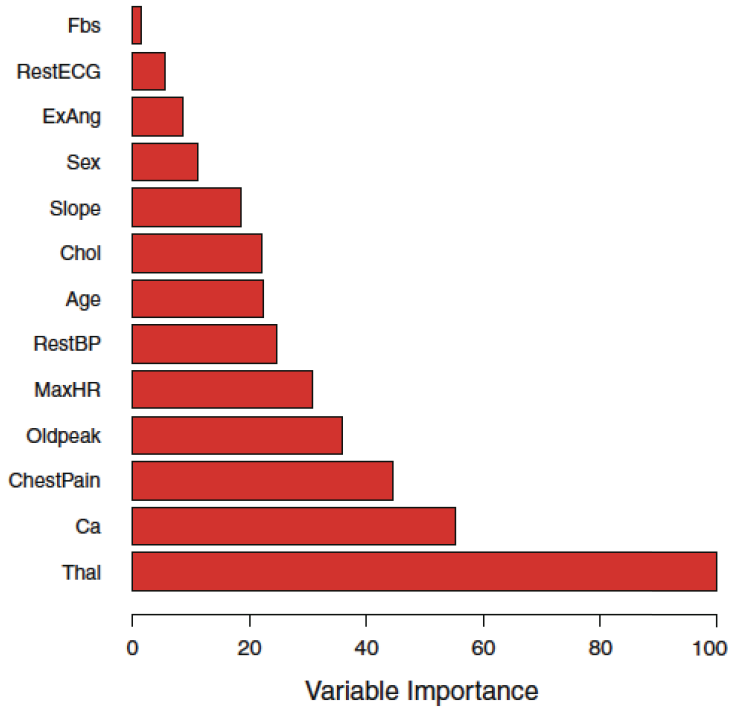
Measuring predictor importance

- With just one tree, it was easy to pick out the most important predictor (which one was it?)



Measuring predictor importance

- With lots of trees, we can't just "read from the top"
- Instead, we can look at average reduction in RSS or Gini due to splits on a given predictor (we'll see this in lab)



Just one problem…

- One issue with bagging is that it sometimes gives us trees that are pretty **highly correlated** (why?)

If we have one **very strong predictor** in the data set, most or all of the trees will use this predictor in the **top split**

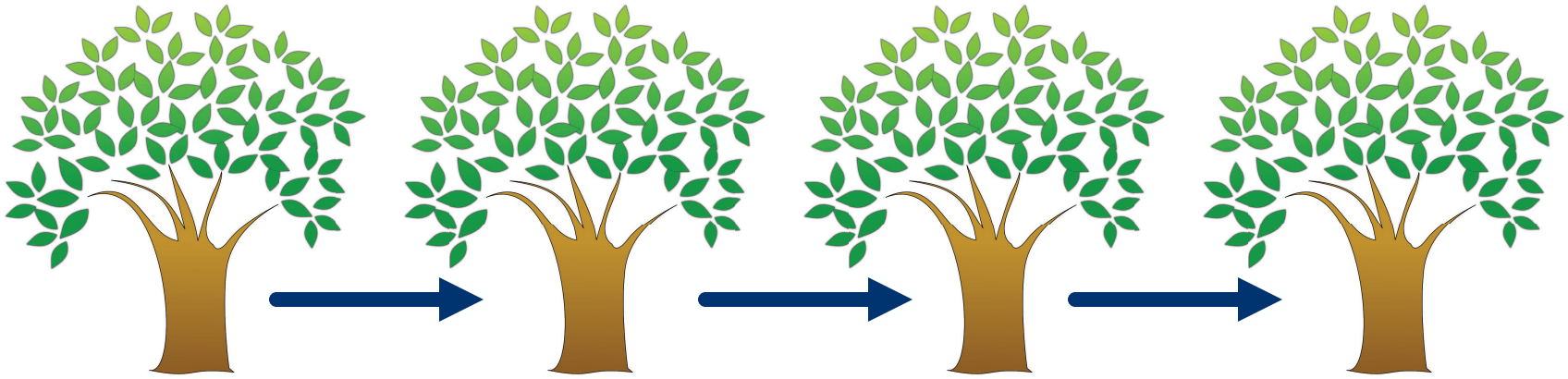
- Averaging highly correlated values **doesn't reduce variance** as much as averaging uncorrelated quantities

Random forests

- **Strange idea**: what if each time we go to make a split, we randomly limit the choice to some *subset* of predictors?
  - Only consider m **out of** p predictors each time we decide on a split
  - Roughly $(p\text{-}m)/p$ splits **won't even consider** that bully predictor, so other predictors will have a chance
  - **Note:** when $m = p$, this is just bagging!

Boosting

- **Previous methods:** generate a bunch of training sets, fit a tree on each one independently, and aggregate results



- **Boosting** works in a similar way, except that each tree is grown using information from **previous trees**

Boosting

- **Big idea:** fit each new tree using the **residuals** from the previous tree as the response

- A shrinkage parameter λ slows the process even more, allowing different-shaped trees to try to deal w/ residuals

- By fitting small trees to the residuals (i.e. variance we haven't yet explained), we **slowly*** improve the model in areas where it makes mistakes

* in general, statistical learning approaches that learn **slowly** tend to perform **well**

Boosting: algorithm

1. Initialize $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set
2. For each $b = 1, 2, \ldots, B$:
   a) Fit a tree $\hat{f}^b$ with d splits ($d + 1$ terminal nodes) to (X, r)
   b) Update $\hat{f}$ by adding in a shrunken version of the new tree:
   $$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x)$$
   c) Update the residuals: $r_i \leftarrow r_i - \lambda \hat{f}^b(x_i)$
3. Output the boosted model:
   $$\hat{f}(x) = \sum_{b=1}^{B} \hat{f}^b(x)$$

Takeaways

- Tree-based methods **partition the predictors** into a number of simple regions, and use the average value of each region to make predictions
- While easy to interpret, trees **typically won't outperform** other methods we've seen in terms of prediction accuracy
- Bagging, random forests, and boosting all try to fix this by **growing multiple trees** and using "consensus prediction"
- In lab, we will see that **combining a large number of trees** can result in dramatic improvements in prediction accuracy (at the expense of some *loss in interpretability*)

Lab: bagging, random forests, & boosting

- To do today's lab in R: **`tree`**, **`randomForest`**, **`gbm`**
- To do today's lab in python: **`graphviz`**
- Instructions and code:

[course website]/labs/lab14-r.html

[course website]/labs/lab14-py.html

- Full version can be found beginning on p. 324 of ISLR

Coming up

- A7 due tonight by 11:59pm
- Next week: **final project workshop**
- After break: **support vector machines**