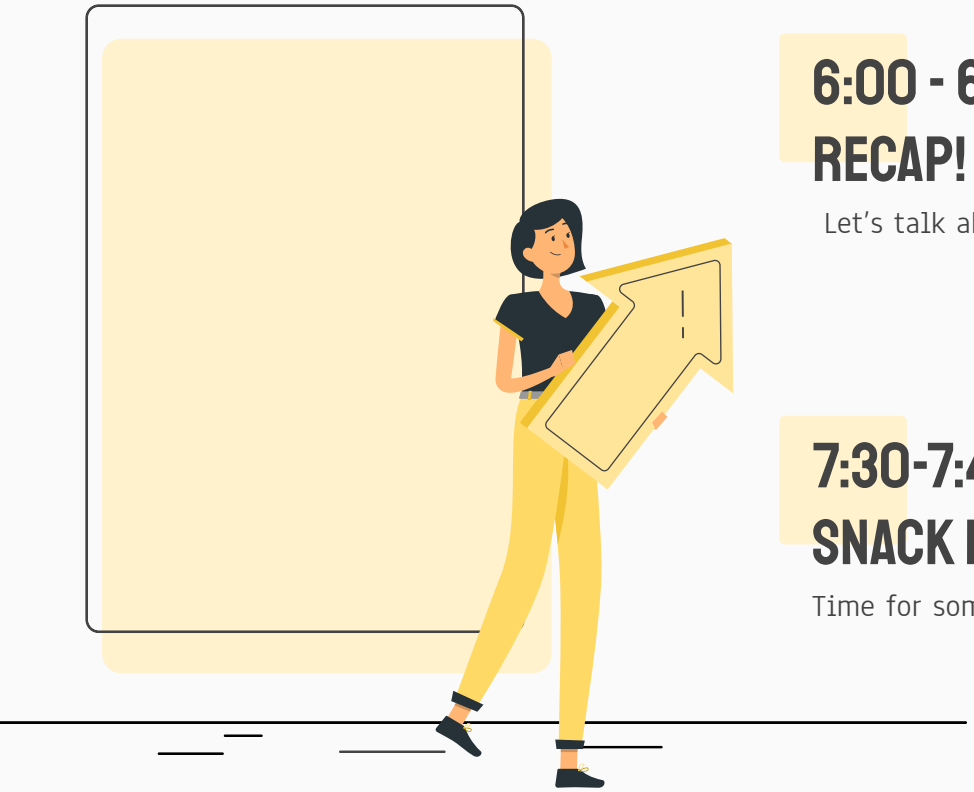
APPLIED DATA SCIENCE II

Week 7: Trees!

Kyle Scot Shank
WI-22



6:00 - 6:30
RECAP!

Let's talk about it!

6:30-7:30
TOPICS + CODE!

Let's pump up our power with some **additive models and other kinds of neat stuff!** !

7:30-7:45
SNACK BREAK!

Time for some munchies

7:45 - 9:00
HANDS-ON CODE LAB

Work through stuff together

RECAP

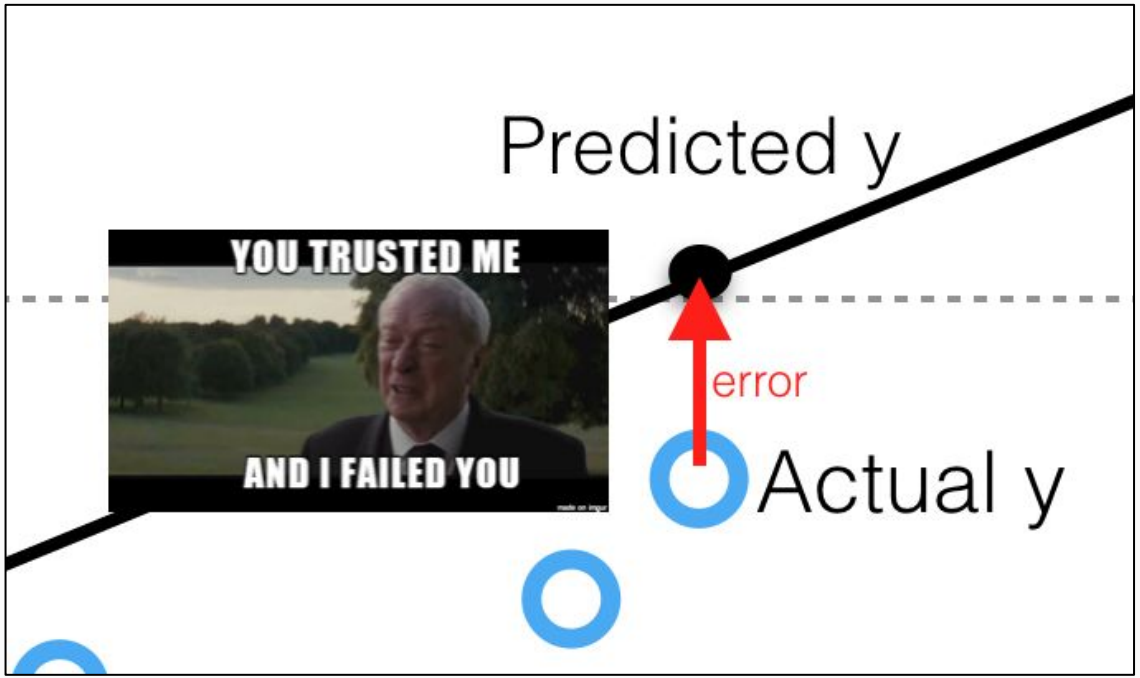
WHY ARE WE HERE?

The purpose of this course:

Sometimes, we build models to help us understand the world.

Other times, we build models to help us predict what might happen.

In this class, we'll do (and have already done!) a little of both!



WE STARTED WITH THE BASICS

We started with linear regression.

Why?

Because it's simple and useful - like a hammer. And, much like the saying goes, "when you think like a hammer, all you see is nails".

*Well, thinking about models through the lens of linear relationships makes it so that all you see are relationships - **not** math.*

AFTER LEARNING ABOUT APPLE PICKIN', WE LEARNED HOW TO PICK APPLES (AND REGULARIZE THEM)

We quickly realized something: we knew how to put a lot of stuff into models, but we didn't know how to figure out what worked.

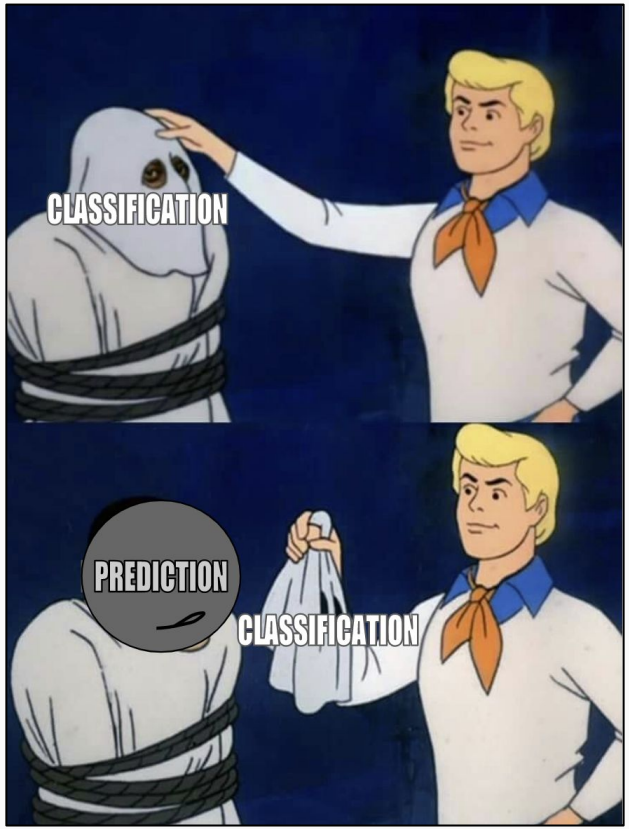
So we introduced selection (best subsets) and regularization (ridge and lasso).

These models let us keep the basics from linear regression, but helped us make smarter models by only keeping the stuff the mattered.

THEN WE LEARNED ALL ABOUT LABELS, LOGISTICS, AND NOT AT ALL ABOUT LINEAR DISCRIMINANTS

Next, we dived into the beautiful sibling of regression: classification! We learned about why you need a different approach when trying to predict a label vs. a number, and some great approaches to doing so (logistic regression, knn).

We also started talking about the idea of predictive accuracy - and why your model training error (i.e., how good a model fits the data you trained it on) and test error (how good it fits on the unseen) are different and important metrics to know.

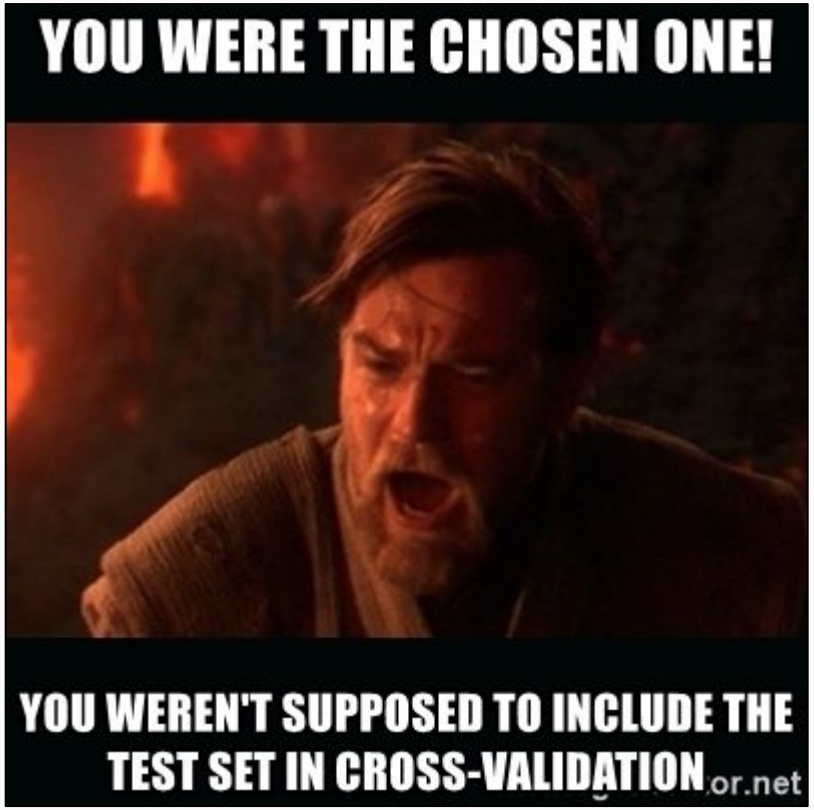


A LITTLE DETOUR INTO THE WHACKY WORLD OF SAMPLING THEORY

Speaking of training and testing errors: we next took a quick break to talk about one of the most uninteresting (but seriously, seriously useful) parts of modern statistical learning: cross-validation!

This technique - which is basically just shuffling the cards of your training data a bunch of times to minimize training error - usually helps our final model to be more robust and accurate in its predictions.

Remember the three rs of data conservation: *reduce*, reuse, ***and*** *reshuffle!*

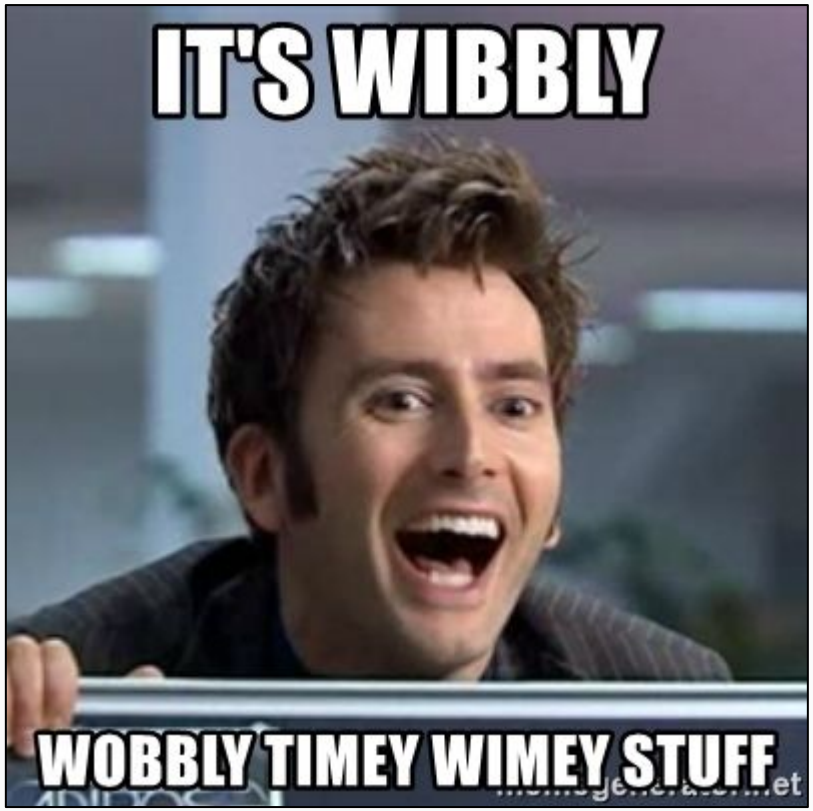


THINGS GOT WEIRD (AND WIGGLY) WITH GAMS AND SPLINES..

Now that we covered all the basics of regression, classification, and cross-validation, we took our first steps into more advanced things: namely, wiggly lines!

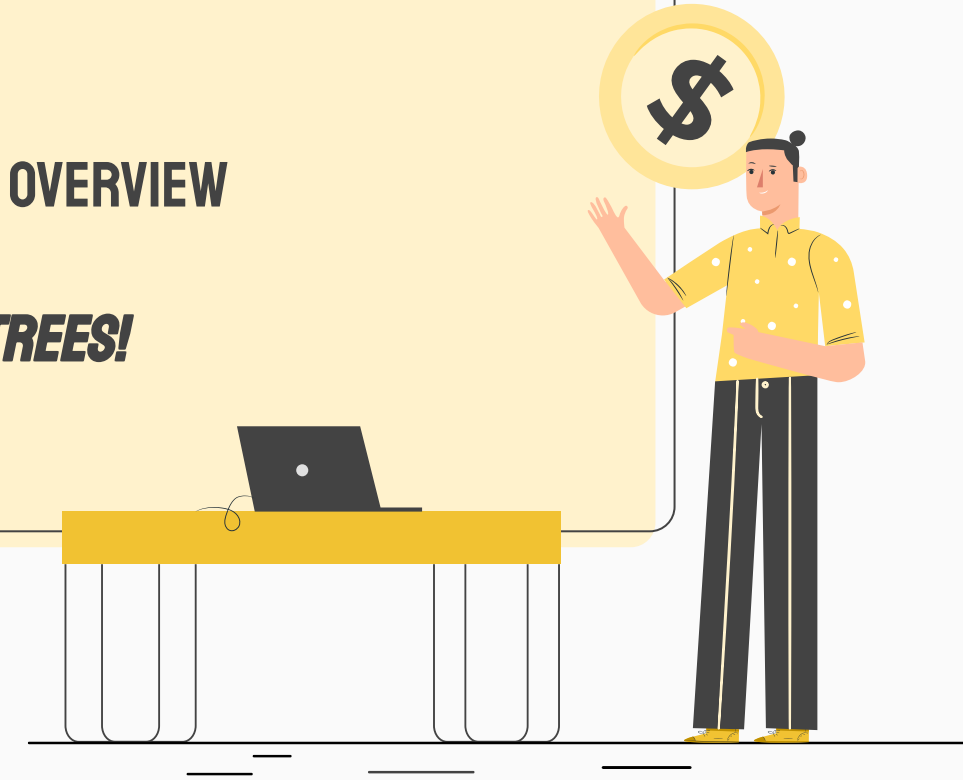
We finally started loosening our grip on things and looked past just linear relationships into those that were nonlinear by using the power of splines and generalized additive models.

We've looked into the abyss and have learned arcane magic. We must now go forward and master the darker arts.



TOPIC OVERVIEW

TREES!

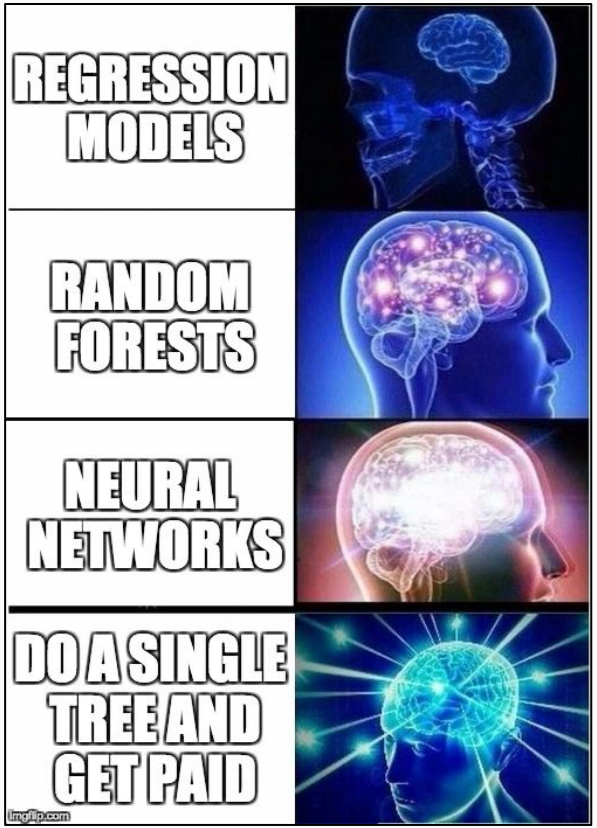


SEEING THE FOREST FOR THE TREES...

WE'RE NOW LEAVING THE WORLD OF "SIMPLE STATISTICS" AND ENTERING THE WORLD OF "MACHINE LEARNING"

YOU ARE HERE ->

I AM HERE AND IT RULES ->



WHAT ARE DECISION TREES?

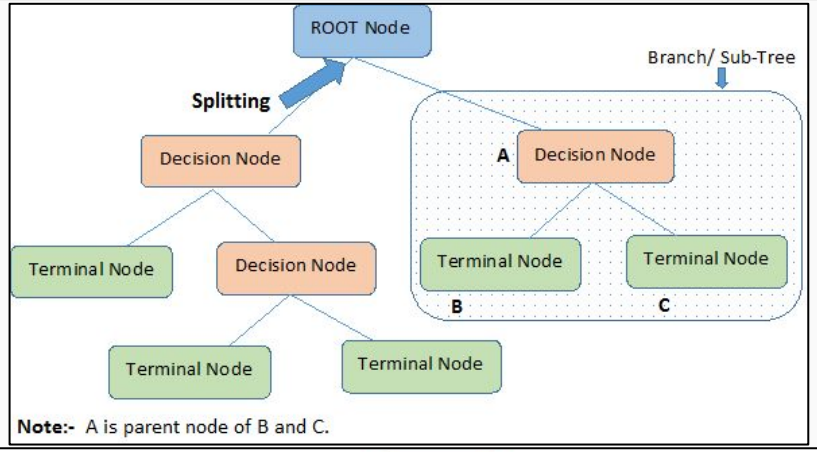
What makes this "machine learning"?

Decision Trees are the foundation for many classical machine learning algorithms like Random Forests, Bagging, and Boosted Decision Trees. They were first proposed by Leo Breiman, a statistician at the University of California, Berkeley. *(These are sometimes called CART models - "Classification and Regression Trees")*

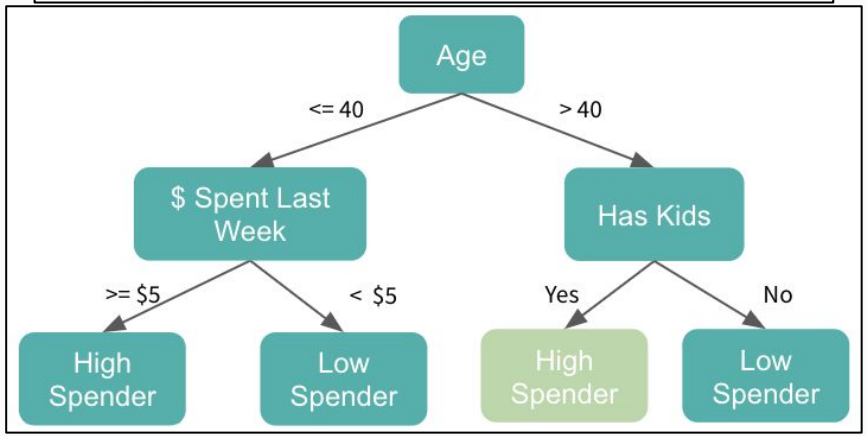
His idea was to represent the data as a tree where each internal split of a branch denotes a test on an attribute, each branch represents an outcome of the test, and each leaf represents some kind of label.

WHAT ARE DECISION TREES?

This is the "abstract" model of a decision tree - showing each branch (i.e., a split on some kind of decision) and each leaf (a terminal node - sometimes a label or a number)



Here it is in a less abstract way: three variables (age, $ spent last week, and having kids) is used to classify someone as either a high or low spender.



WHAT ARE DECISION TREES?

Pros to using decision trees:

- *Decision trees are able to generate understandable rules that mirror human reasoning.*
- *Decision trees are able to handle both continuous and categorical variables.*
- *Decision trees provide a clear indication of which fields are most important for prediction or classification.*
- *Decision trees are easy to visualize!*

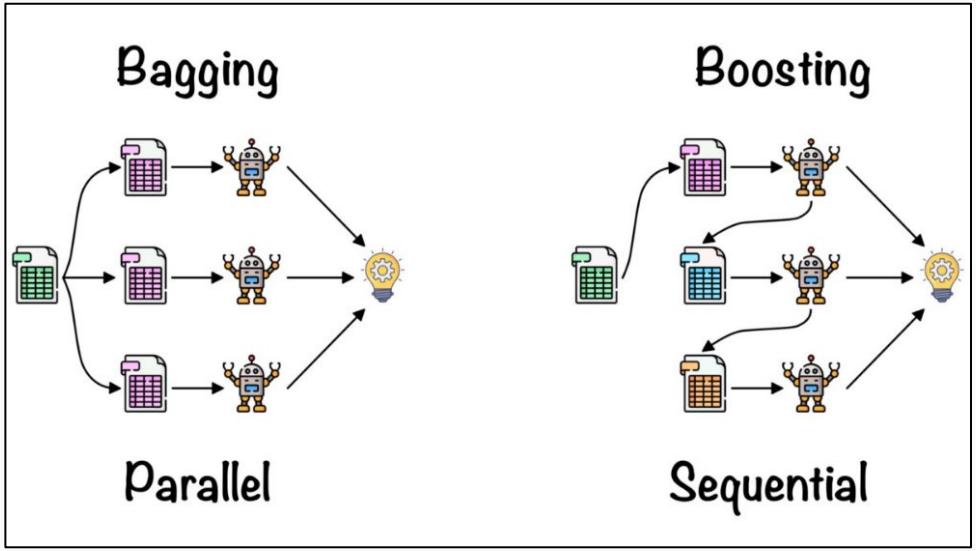
Cons to using decision trees:

- *Decision trees are less appropriate for estimation tasks where the goal is to predict the value of a continuous attribute.*
- *Decision tree can be computationally expensive to train.*
- *Decision trees tend to have less predictive accuracy than other forms of prediction (GAMS, etc.) - but this improves dramatically when we aggregate different trees together!*

ENSEMBLE METHODS

We're going to be talking about two different kinds of ensemble methods when it comes to decision trees: Random Forests and Boosted Trees.

Ensemble methods come in (roughly) two flavors: those that rely on bagging and those that rely on boosting. We won't go too deep into the nuances of this and we're going to cover boosting next week.



ENTER INTO THE RANDOM FOREST

The Random Forest® is a supervised learning algorithm. The "forest" it builds is an ensemble of decision trees, usually trained with the "bagging" method (read more about it in the textbook!). The general idea of the bagging method is that a combination of learning models increases the overall model performance.

Put simply: random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction for your dataset.

ENTER INTO THE RANDOM FOREST

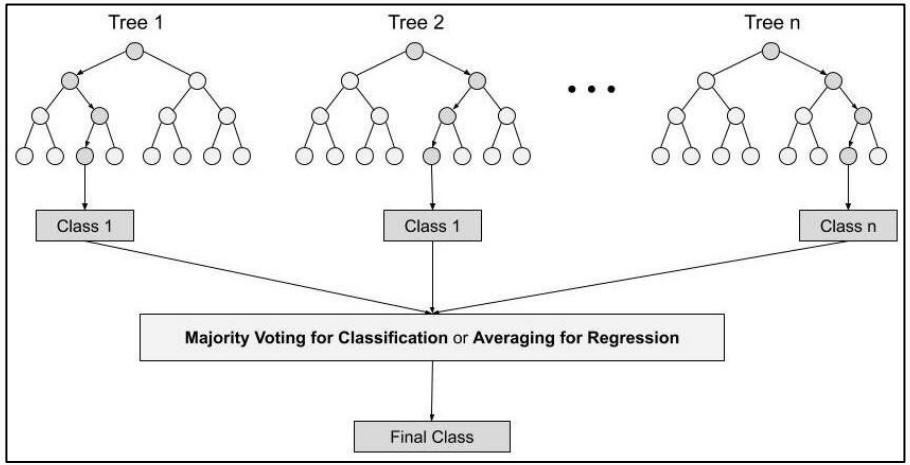
Steps involved in Random Forest:

Step 1: Take n number of random records from the data set and use these as a sample.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression respectively.



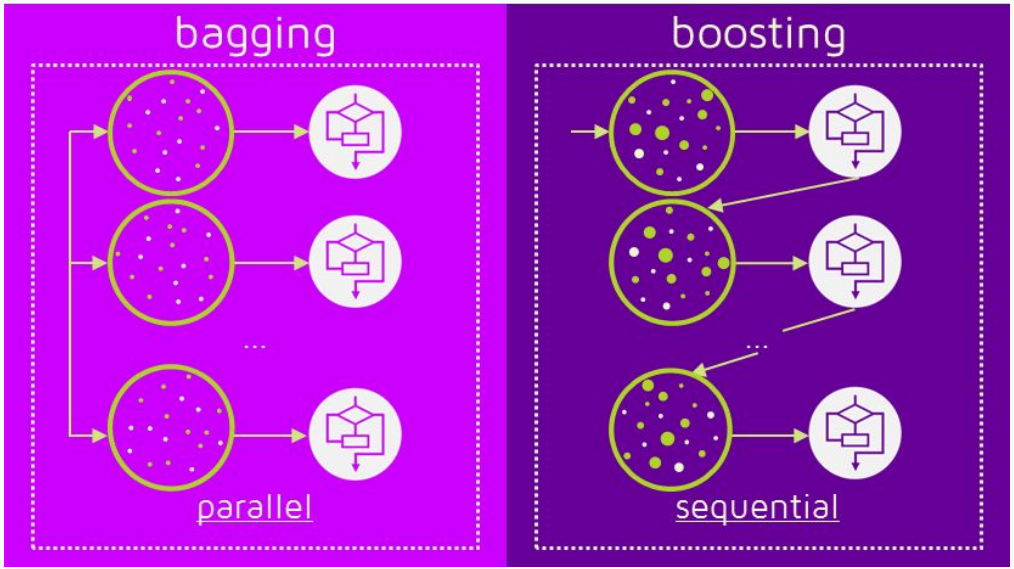
- *Let's do this together!*

Open up R!

LET'S TALK ABOUT BOOSTED TREES...

The math behind boosted trees can get a little intense, but let's build intuition by comparing it to bagged tree models (like random forest).

1. *Bagging models (like RandomForest) run lots of trees in parallel and then average the results*
2. *Boosting models (like XGBoost) use the previous tree to weight the features of the next one, operating in sequence.*



BAGGING VS. BOOSTING

Similarities

Both generate several training data sets by random sampling…

Both make the final decision by averaging the N learners (or taking the majority of them)…

Both are good at reducing variance and provide higher stability…

Differences

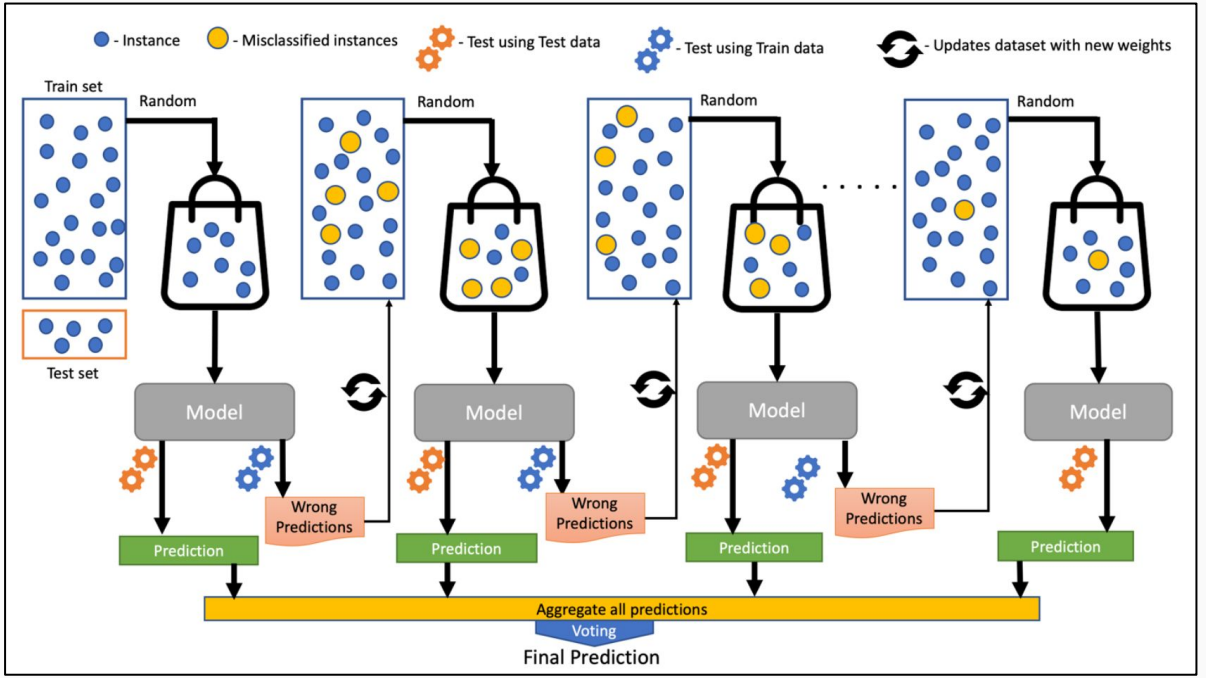
… but only Boosting determines weights for the data to tip the scales in favor of the most difficult cases.

… but it is an equally weighted average for Bagging and a weighted average for Boosting, more weight to those with better performance on training data.

… but only Boosting tries to reduce bias. On the other hand, Bagging may solve the over-fitting problem, while Boosting can increase it.

XGBOOST

Let's talk through this!



- *Let's do this together!*

Open up R!

SNACK BREAK!

COME BACK IN 15!

CODE LAB!

OPEN UP RSTUDIO