Boosting

GENERAL ASSEMBLY

Learning Objectives

- Understand the differences between bagging and boosting
- Understand how boosting is an ensemble method
- Learn the intuition behind AdaBoost

Review: what is bagging?

- **Bootstrapping**: sampling with replacement on the original training data
- **Aggregating**: using predictions made by many models in aggregate

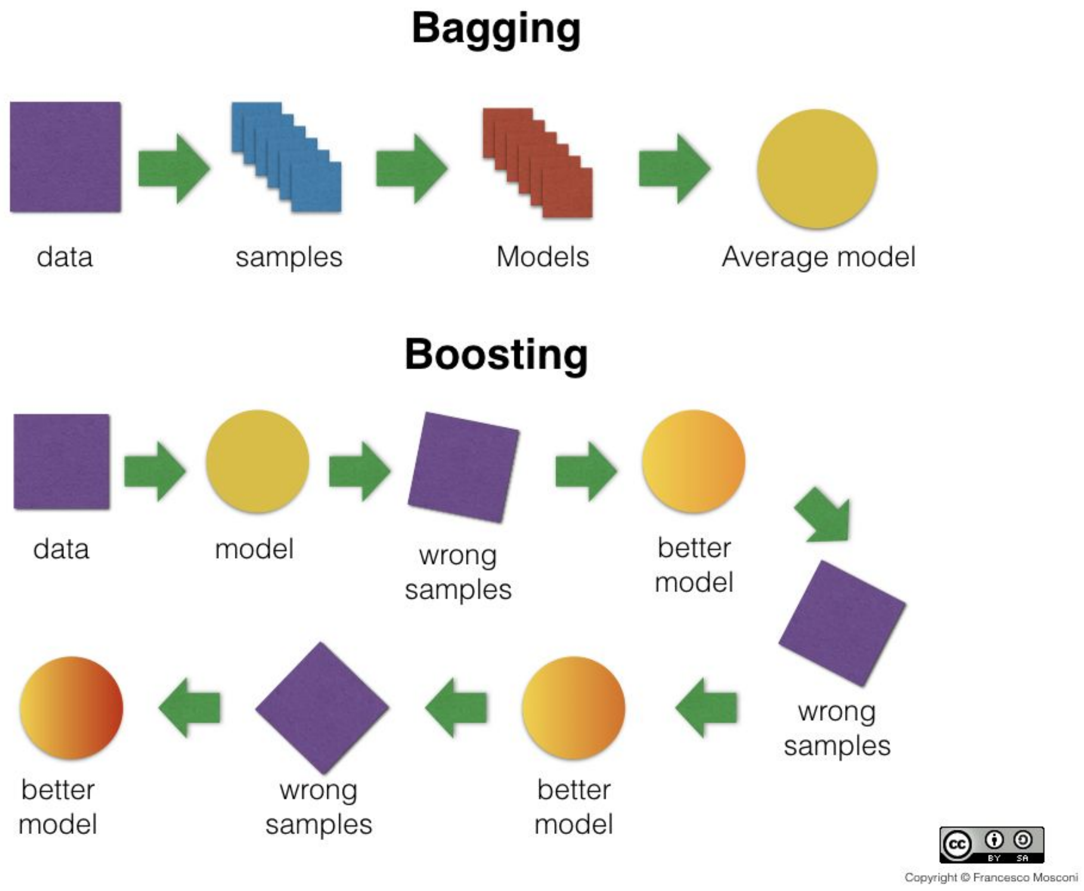
A simple diagram

Some properties of bagged models

- **Fit in parallel**
- Decreased variance
- Individual models may still be correlated with each other
    - Random forests introduce additional randomness by limiting the features each split is allowed to consider

What is boosting?

- **Boosted models** are ensemble models fit sequentially instead of in parallel
- Boosting is much slower
- Can achieve very high performance
- More prone to overfitting


Bagging
data
samples
Models
Average model
Boosting
data
model
wrong samples
better model
wrong samples
better model
wrong samples
better model
Copyright © Francesco Mosconi

What is boosting?

Use many **weak learners**, often **decision stumps**, to develop a single **strong learner**

- Weak learner: a model/learner/estimator that performs poorly, often just better than chance
- Decision stump: a decision tree with just one split
- Strong learner: a model that performs well

AdaBoost

- AdaBoost is the original boosting algorithm
- Fit a sequence of weak learners on repeatedly modified versions of the data
    - Each subsequent weak learner is fit on a boostrapped sample, but training data that was misclassified is *more likely to be sampled*. Over time, difficult-to-classify observations become ‘focused on’
- Each weak learner is weighted to produce the final prediction
    - More useful weak learners will be upweighted, and less useful weak learners will be downweighted

Other boosting algorithms

- Focus on AdaBoost this morning
- Also **gradient boosting**
  - Big on Kaggle
  - Fits subsequent models to the residuals of the last model instead of resampling with misclassified observations upweighted

Let’s implement it in code!