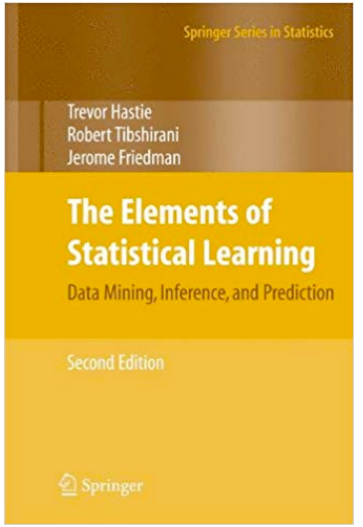
Course material at: https://git.io/fjFga



Model Engineering

Tuesday, Lecture 1

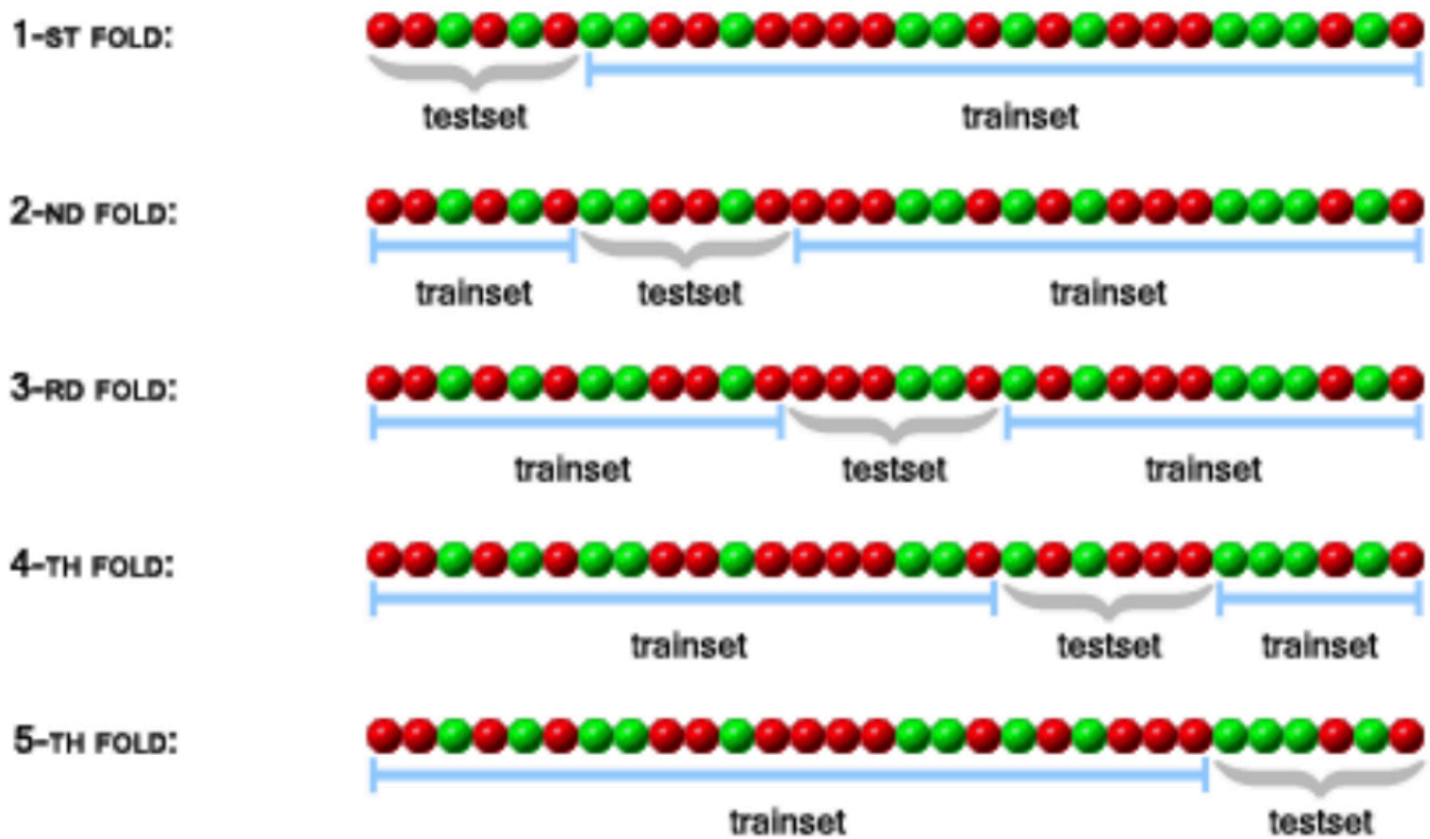
FASE ML Bootcamp

Based on material from:
Elements of Statistical Learning by Hastie, Tibshirani, Friedman,
Theory and Applications of Boosting by Schapire

kaggle

Example: one run of *5-fold* cross validation

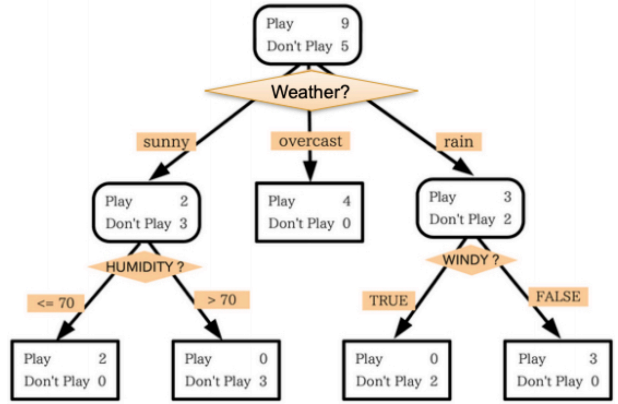
You should do a **few runs** and **compute the average**
(e.g., error rates if that's your evaluation metrics)



Cross validation

1. Divide your data into n parts
2. Hold 1 part as “test set” or “hold out set”
3. Train classifier on remaining n-1 parts “training set”
4. Compute test error on test set
5. Repeat above steps n times, once for each n-th part
6. Compute the average test error over all n folds
   (i.e., cross-validation test error)

Model Combination



Any single model may have **high variance** in its predictions

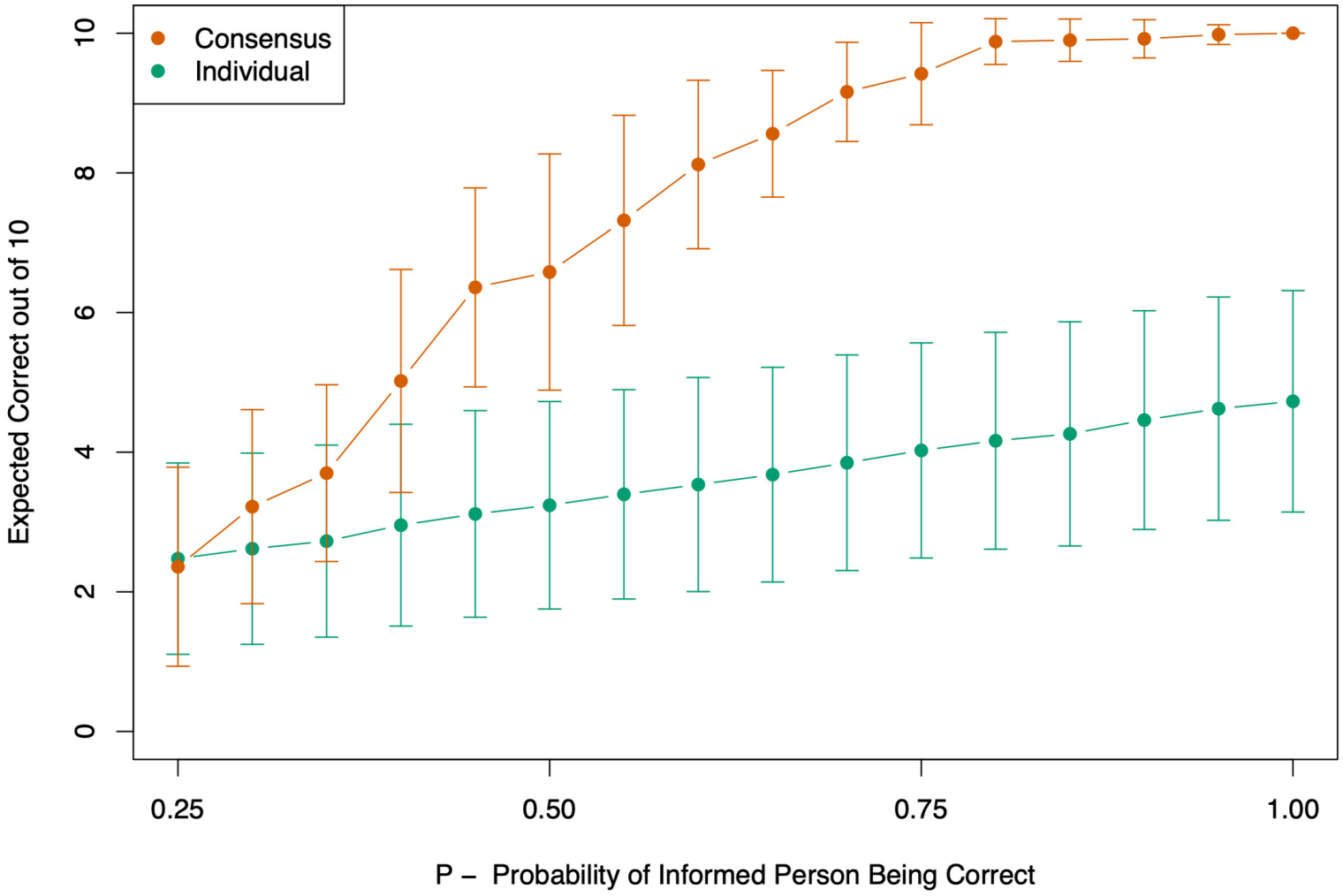
- Decision tree: predictions can be brittle, small perturbation in data can lead to misclassification

Creating more stable models:

- Come up with a completely new model
- **Combine (unstable) models intelligently!**

Simulated academy awards voting

- **50 members** vote in **10 categories**, each with **4 nominations**. For any category, **only 15 voters have some knowledge**, represented by their probability of selecting the “correct” candidate in that category (so P = 0.25 means they have no knowledge).
- For each category, the **15 experts are chosen at random** from the 50.
- Results show the **expected number of correct decisions** (based on 50 simulations) for the consensus, as well as for the individuals. The error bars indicate one standard deviation.
- We see, for example, that **if the 15 informed for a category have a 50% chance of selecting the correct candidate, the consensus doubles the expected performance of an individual.**



Elements of Statistical Learning, Figure 8.11

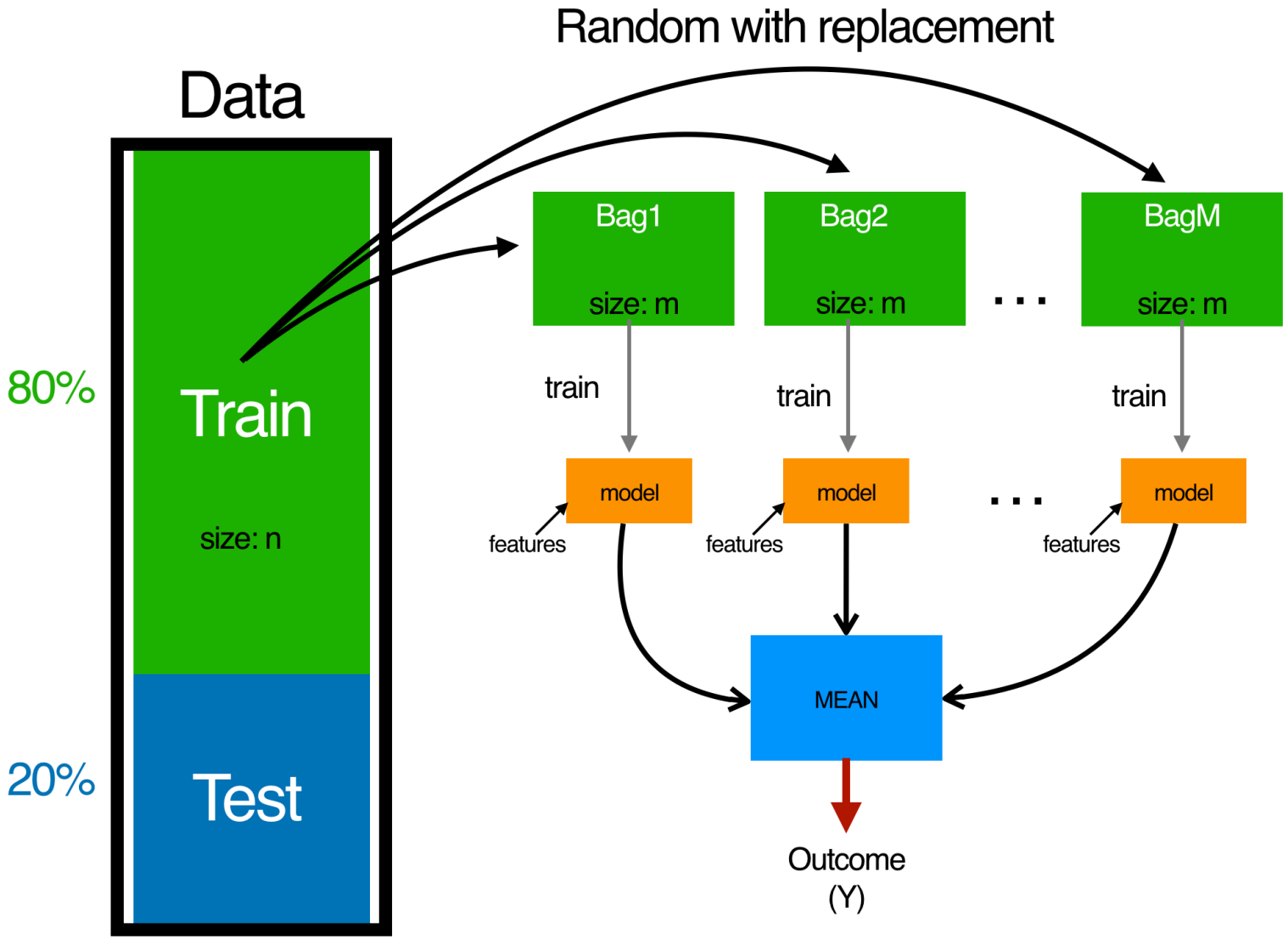
Bootstrap Sampling

Data: $S = \{(x_i, y_i)\}_{i = 1,\ldots,n}$

Bootstrap Sample: subsample of S, drawn **with replacement**

https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205

Bootstrap **Aggregating (Bagging)**

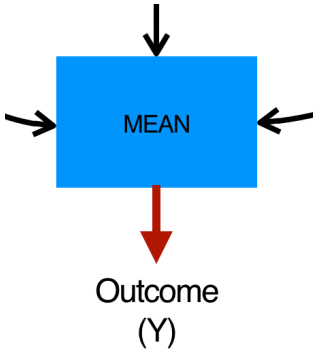


http://www.ub.edu/stat/docencia/EADB/bagging.html

Bagging

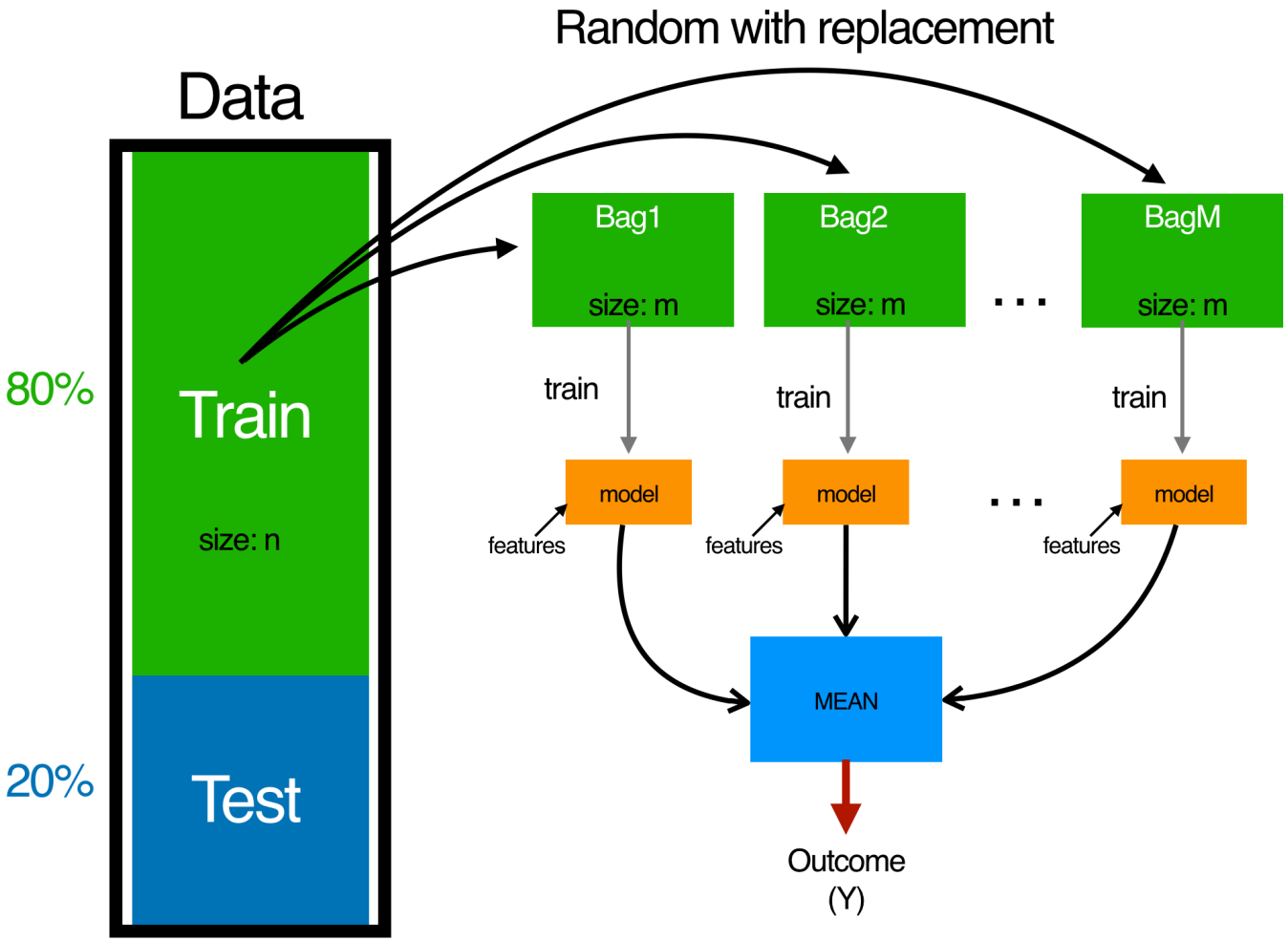
Aggregation:

- **Majority vote!**



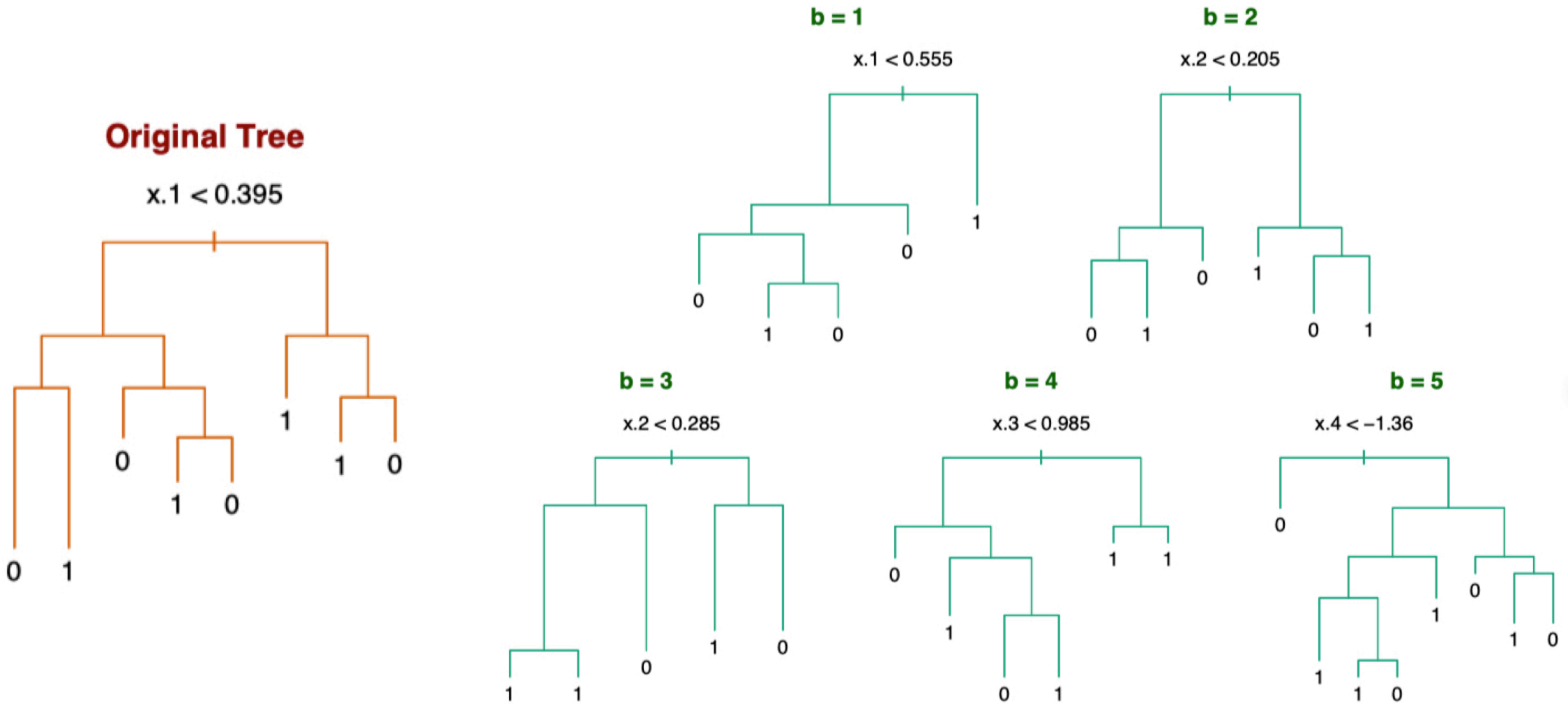
Benefits:

- Even if single model overfits, on average they will not



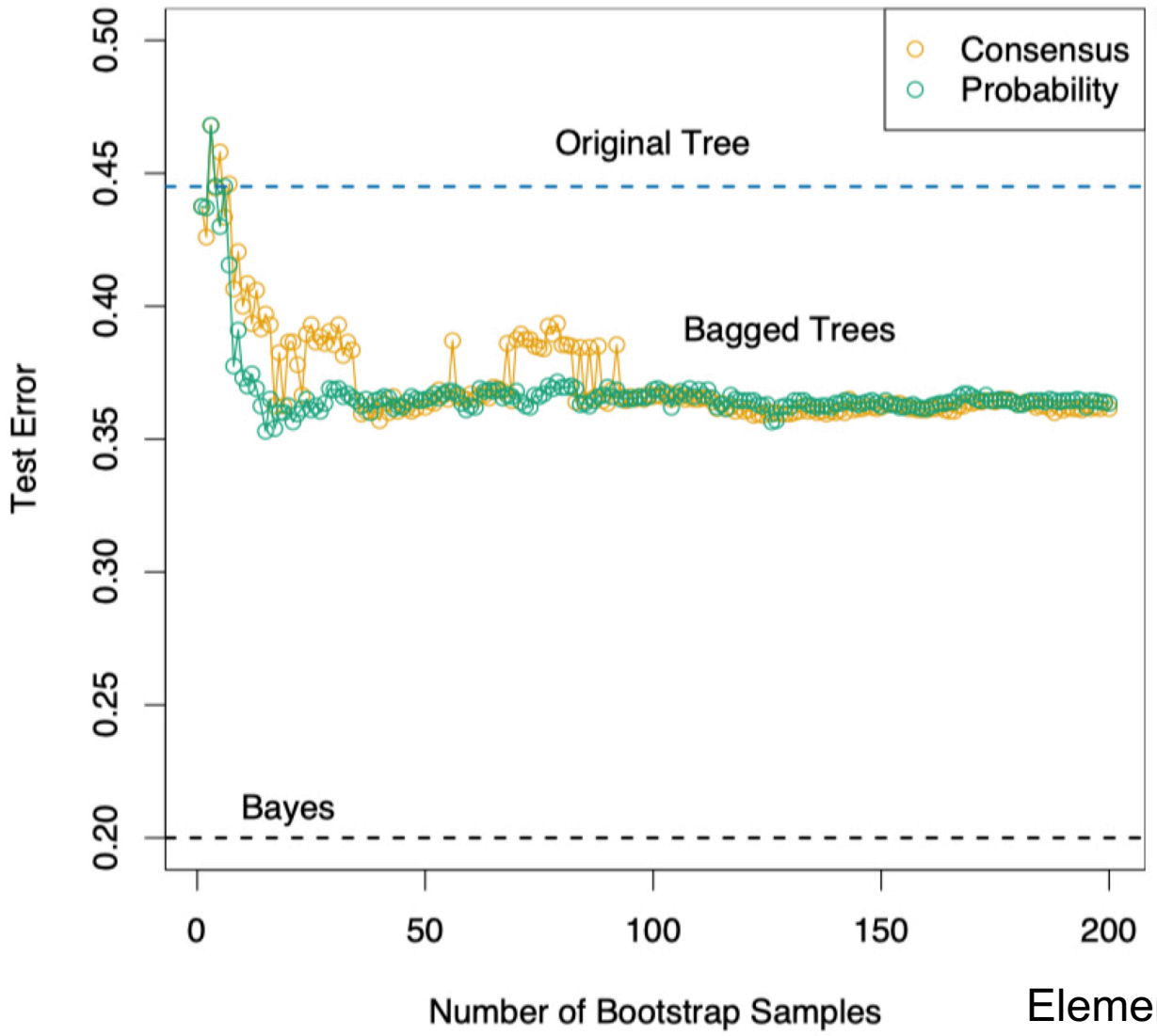
http://www.ub.edu/stat/docencia/EADB/bagging.html

Bagging with Decision Trees



Elements of Statistical Learning, Chapter 8.7

Predicting with bagged decision trees



Consensus: proportion of trees that predict a given class

Probability:

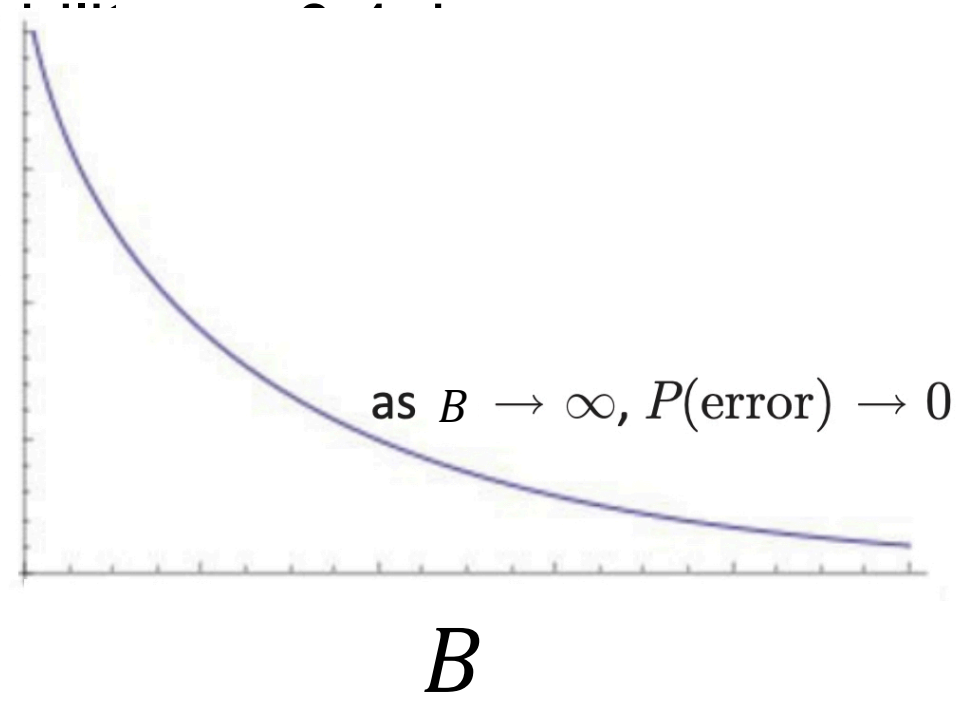
- For b-th tree, look at the leaf corresponding to test sample
- What proportion of the training data had the same label as test sample?
- Average those proportions out

Prediction: in both cases, predict class with highest score

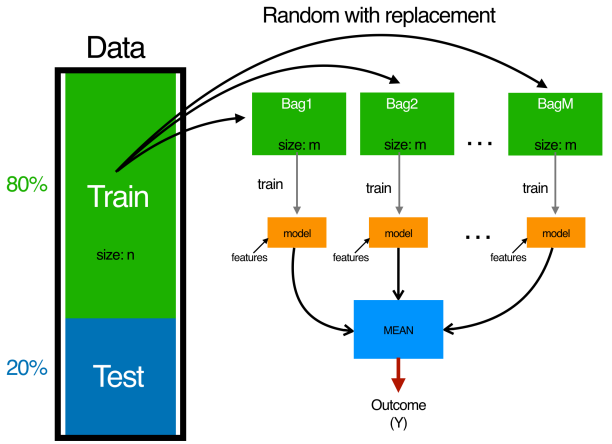
Elements of Statistical Learning, Chapter 8.7

Why does bagging work?

- Single test data point, **x**, with true label 1
- **B independent** classifiers, $f^b(x)$
- Each classifier misclassifies **x** with
$$Prob\left(f^b(x)\right)$$
- The bagged classifier predicts as:
$$f^{bag}(x) = \text{argmax}_{k=0}$$
- $B_0 = \sum_{b=1,\ldots,B} I\{f^b(x) = 0\}$ is the n
- B_0 is **binomial** random variable
- Probability of misclassification:
$$Prob\left(f^{bag}(x) = 0\right) = P(B_0 \geq B/2)$$
- As $B \to \infty$, $Prob\left(f^{bag}(x) = 0\right) \to 0$



When should use Bagging?



High-variance classifiers, e.g., decision trees.

Bagging reduces variance by providing an alternative approach to **regularization**.

Even if each of the learned classifiers are individually overfit, they are likely to **overfit to different things**.

Through voting, we can overcome a significant portion of this overfitting.

In practice, bagging tends to **reduce variance** and **increase bias**.

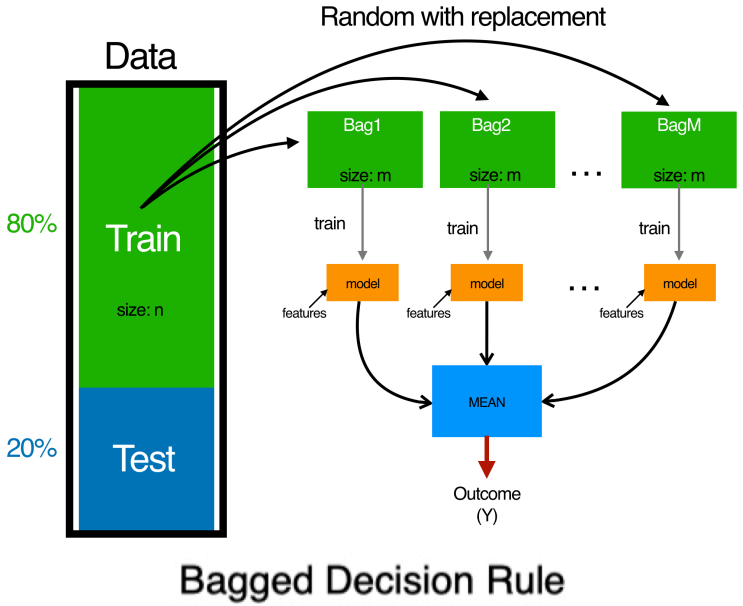
Final words on Bagging

Advantages

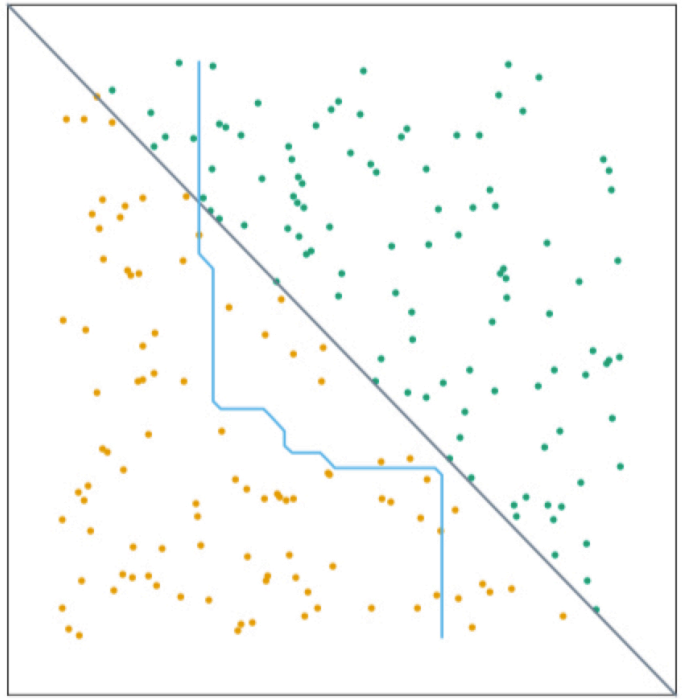
- Reduces model variance
- Trivial to implement and use

Caveats

- **Destroys any interpretability** in the base model
- May not capture simple patterns

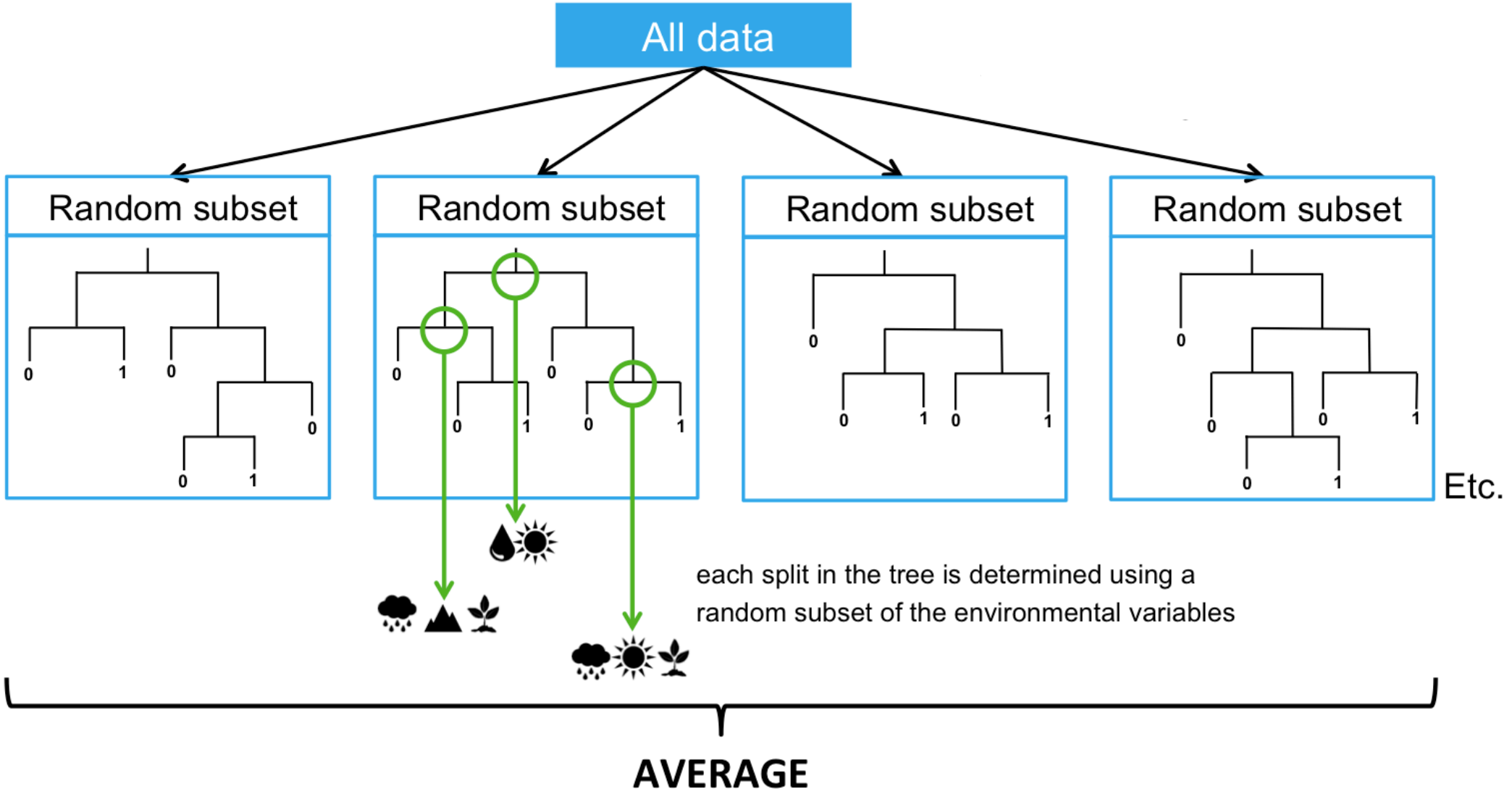


Bagged Decision Rule



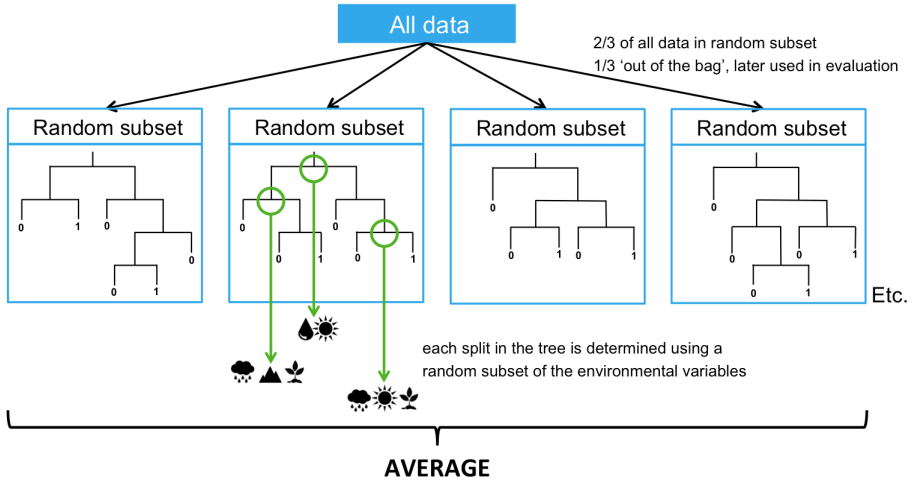
Random Forest

Similar to Bagging Decision Trees **except**:



https://support.bccvl.org.au/support/solutions/articles/6000083217-random-forest

Random Forest



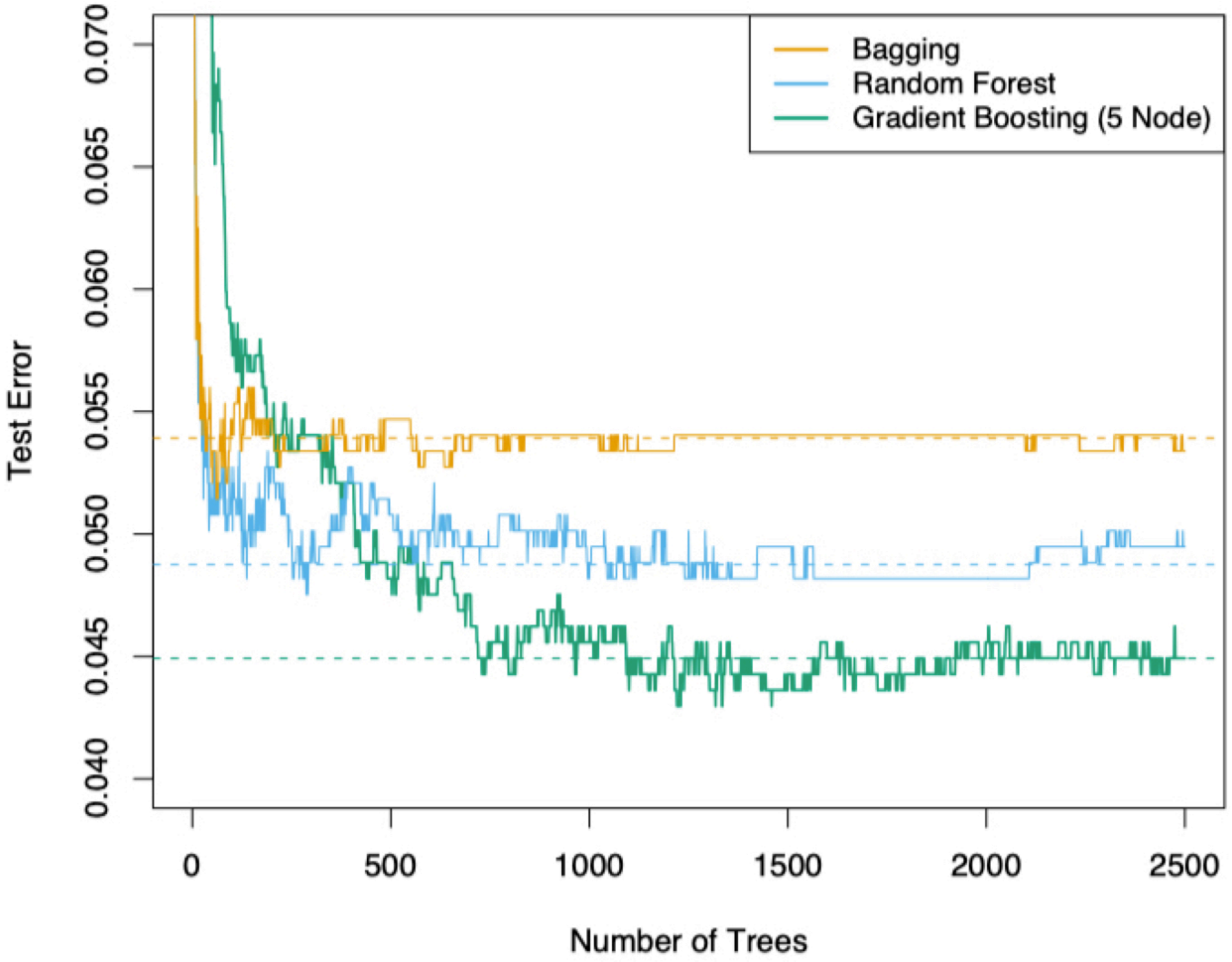
- One of the most **popular** models!
- Usually **more effective** than Bagging Decision Trees
- Typically, use **100s of trees** (hyper-parameter to be tuned!)
- For data with d features, num. random features used for splitting:
  - Classification: $\sqrt{\boldsymbol{d}}$
  - Regression: $\boldsymbol{d/3}$

Example: Spam classification

Dataset of 4000 emails: y = (spam, not spam)

x = % of words equal to 'business', 'address', etc.;

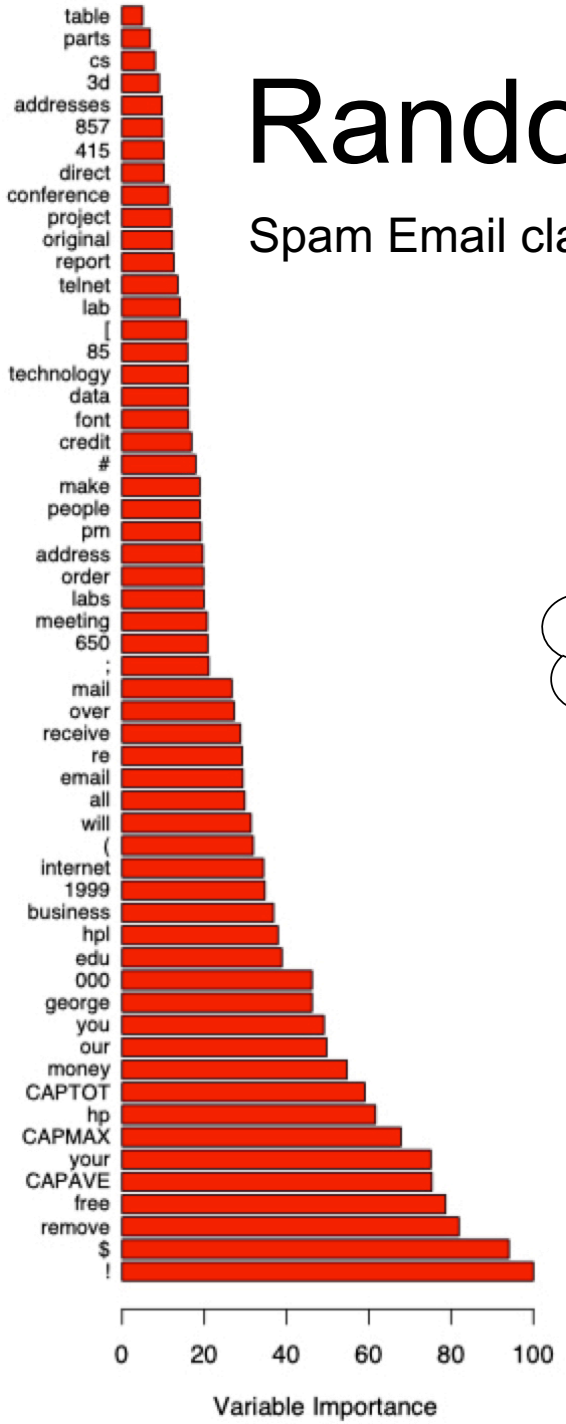
average length of uninterrupted sequences of capital letters, …



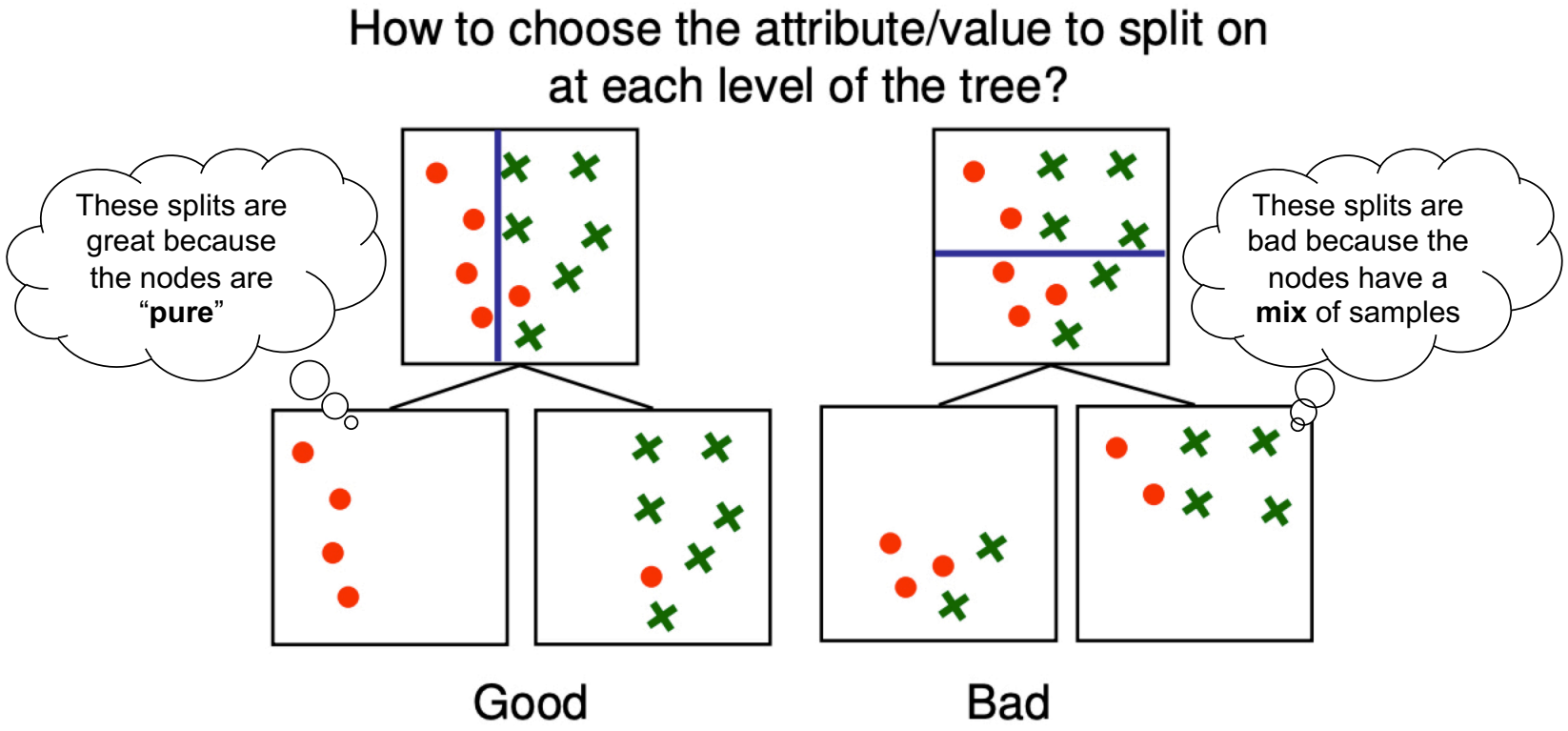
Ignore this for now

Random Forest: Feature Importance

Spam Email classification dataset



words



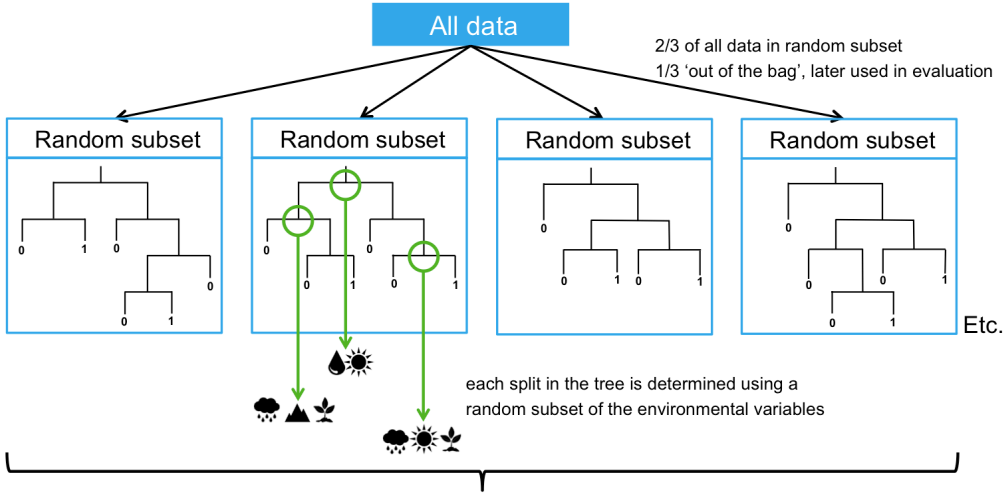
Final words on RF

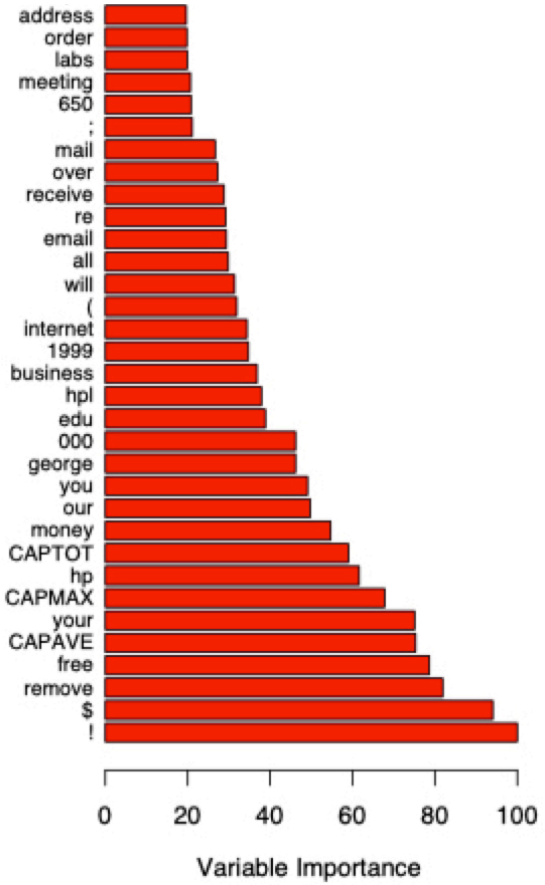
Advantages

- Feed it **data as is** without preprocessing
- **Fast training** because of feature sampling!
- Easy **parallel** training and prediction

Caveats

- **Not suitable for sparse data (why?)**





Boosting: “How May I Help You?”

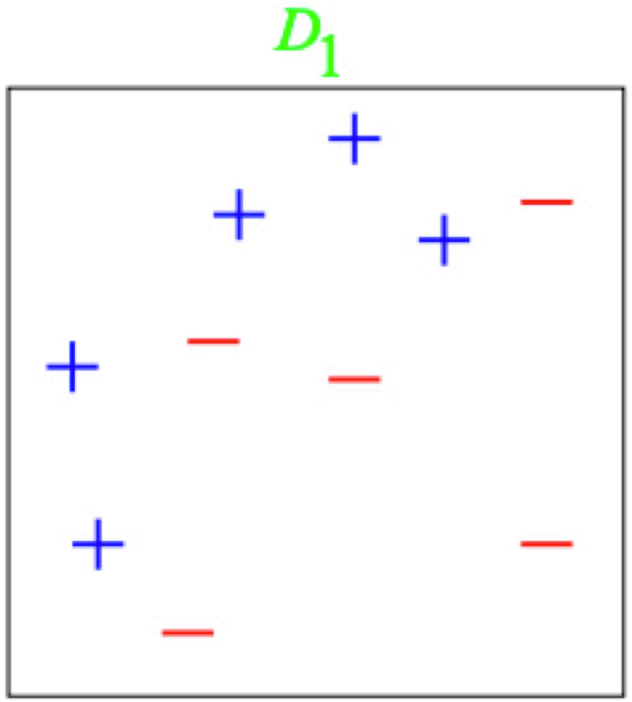
- goal: automatically categorize type of call requested by phone customer (Collect, CallingCard, PersonToPerson, etc.)
  - `yes I'd like to place a collect call long distance please` (Collect)
  - `operator I need to make a call but I need to bill it to my office` (ThirdNumber)
  - `yes I'd like to place a call on my master card please` (CallingCard)
  - `I just called a number in sioux city and I musta rang the wrong number because I got the wrong party and I would like to have that taken off of my bill` (BillingCredit)

Boosting: “How May I Help You?”

- observation:
  - easy to find “rules of thumb” that are “often” correct
    - e.g.: “IF ‘card’ occurs in utterance THEN predict ‘CallingCard’ ”
  - hard to find single highly accurate prediction rule

- goal: automatically categorize type of call requested by phone customer (Collect, CallingCard, PersonToPerson, etc.)
  - `yes I’d like to place a collect call long distance please` (Collect)
  - `operator I need to make a call but I need to bill it to my office` (ThirdNumber)
  - `yes I’d like to place a call on my master card please` (CallingCard)
  - `I just called a number in sioux city and I musta rang the wrong number because I got the wrong party and I would like to have that taken off of my bill` (BillingCredit)

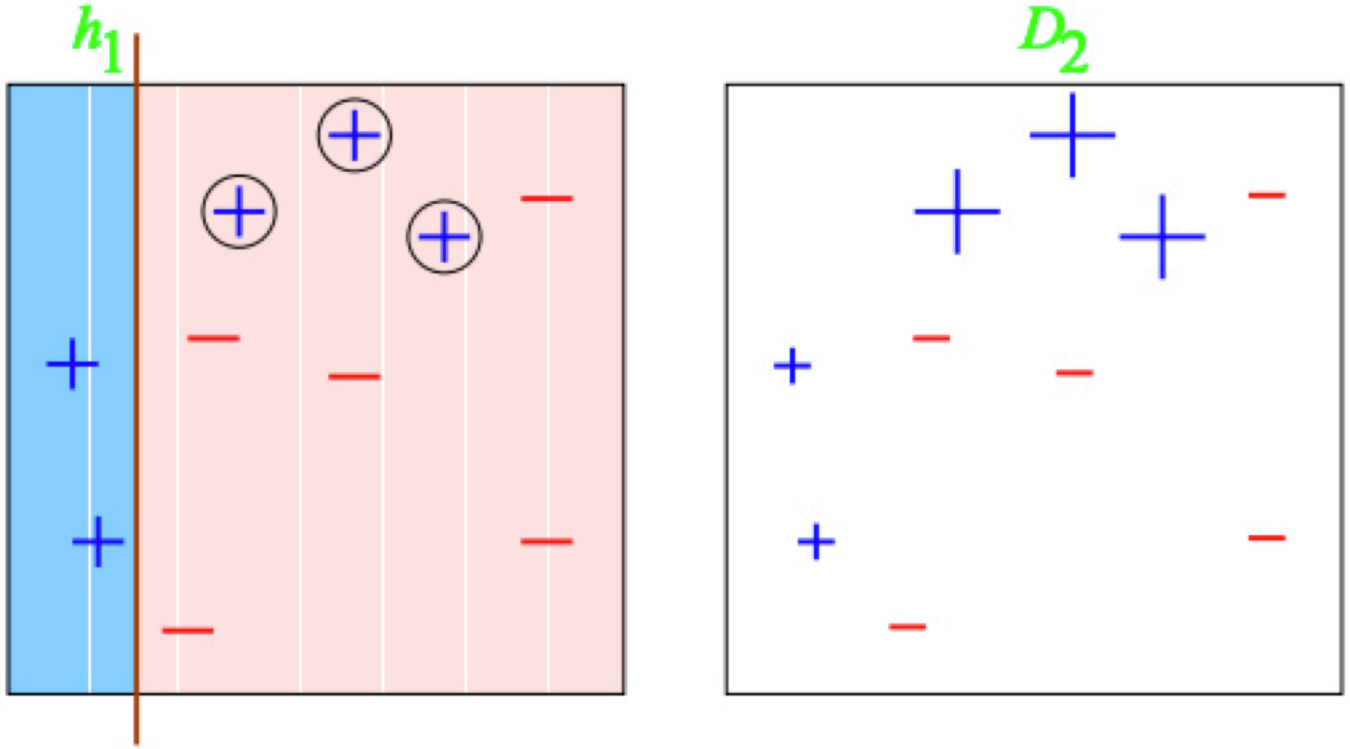
Boosting: A Toy Example



“rule of thumb” classifiers = vertical or horizontal half-planes

Boosting: A Toy Example

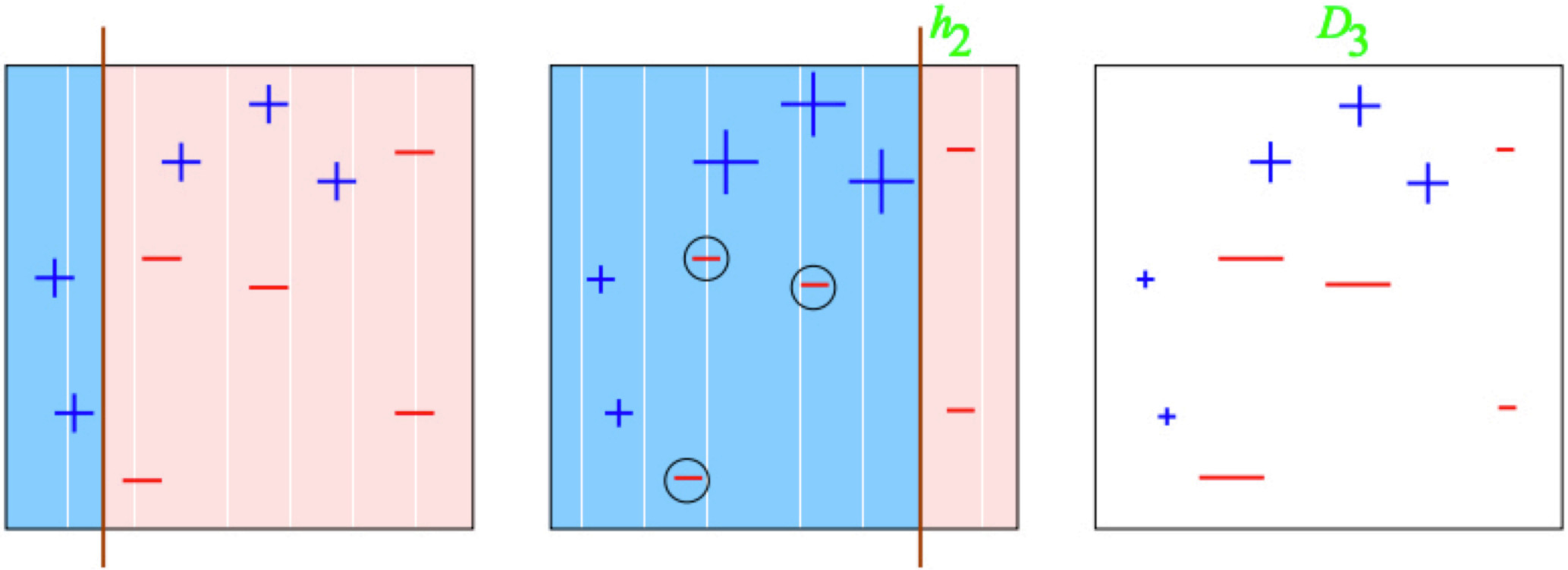
Size of the data point: penalty for misclassifying the point



“rule of thumb” classifiers = vertical or horizontal half-planes

Boosting: A Toy Example

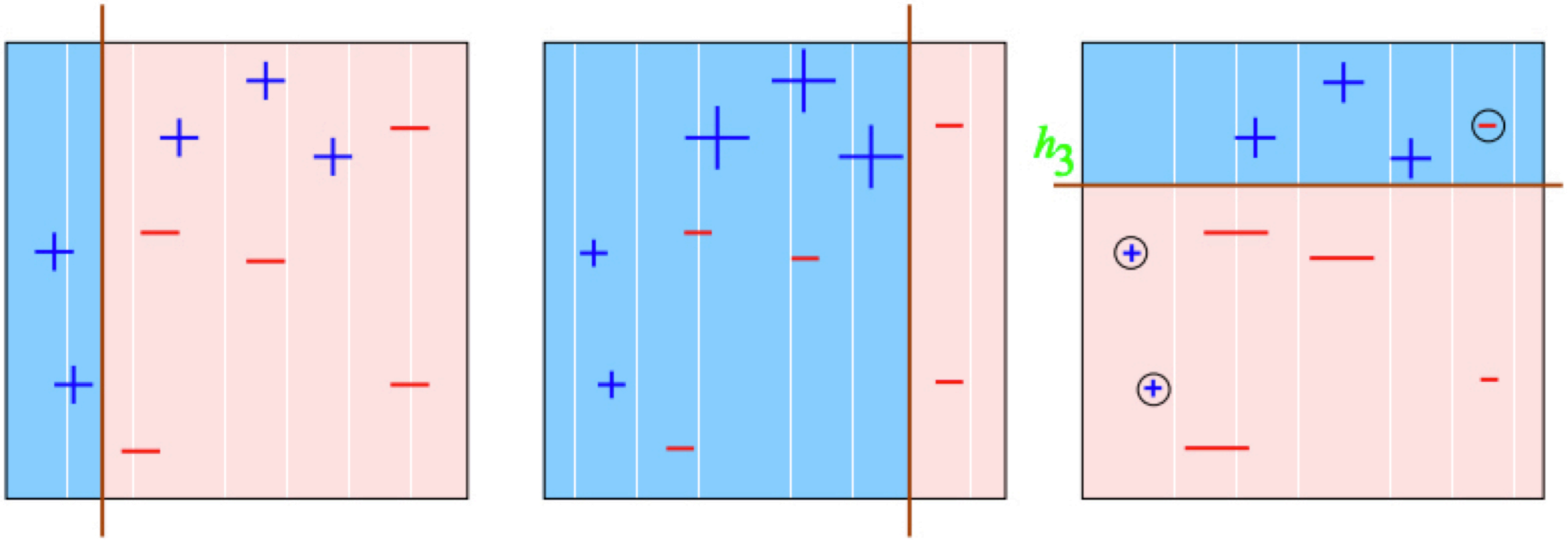
Size of the data point: penalty for misclassifying the point



“rule of thumb” classifiers = vertical or horizontal half-planes

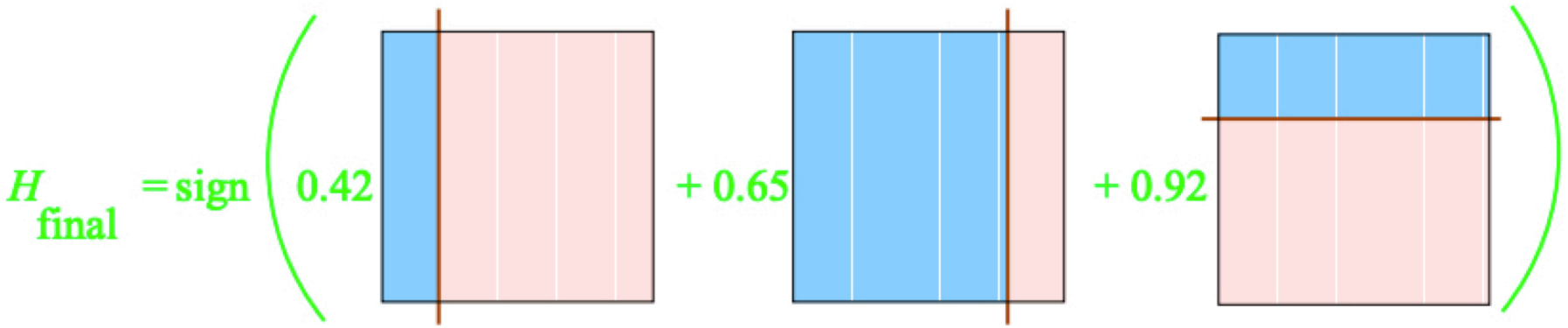
Boosting: A Toy Example

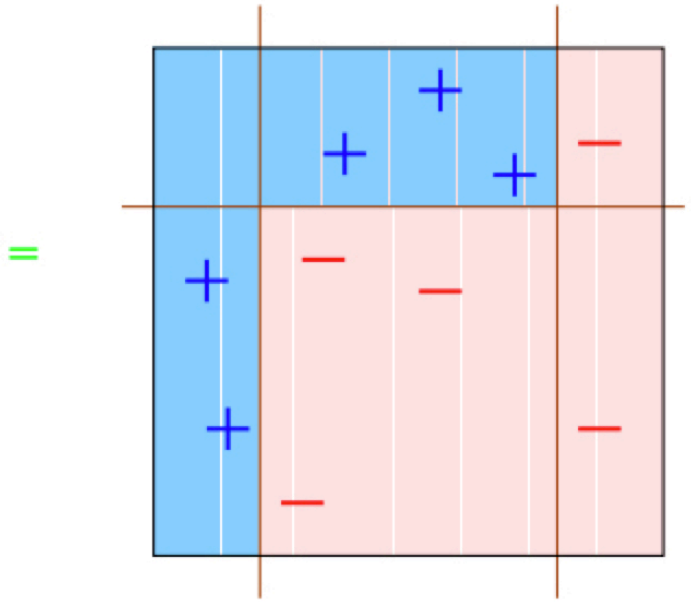
Size of the data point: weight, penalty for misclassifying the point

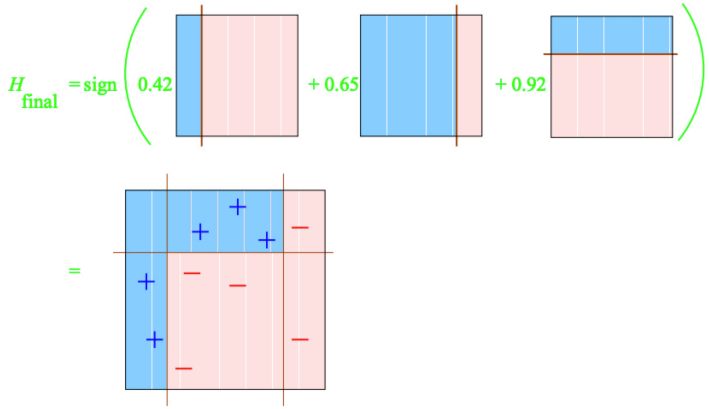


“rule of thumb” classifiers = vertical or horizontal half-planes

Boosting: A Toy Example







Boosting in a Nutshell

A general method of **converting rough rules** of thumb **into highly accurate prediction** rule.

Technically:

- assume given “weak” learning algorithm that can consistently find classifiers (“rules of thumb”) at least **slightly better than random**, say, accuracy ≥ 55%
- given sufficient data, a boosting algorithm can provably **construct single classifier with very high accuracy**

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.

For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : X \to \{-1, +1\}$ with error

$$\epsilon_t = \Pr_{i \sim D_t}\left[h_t(x_i) \neq y_i\right].$$

weighted training error

$$\epsilon_t = \sum_{i:h_t(x_i) \neq y_i} D_t(i)$$

- Choose $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$. **importance of weak learner h_t**

Update:

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t} \end{aligned}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

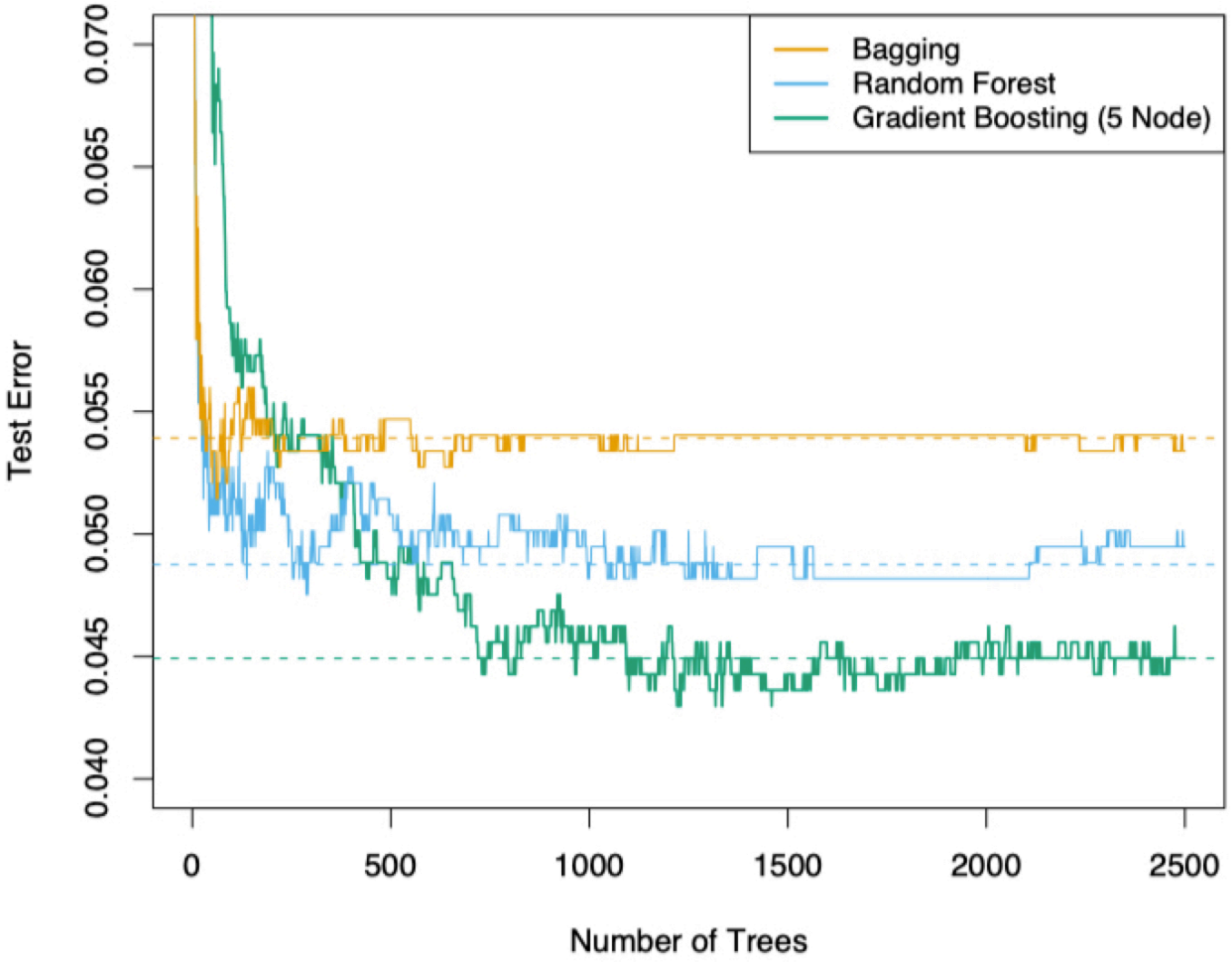
$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

Example: Spam classification

Dataset of 4000 emails: y = (spam, not spam)

x = % of words equal to 'business', 'address', etc.;

average length of uninterrupted sequences of capital letters, …



Boosting outperforms **Random Forests**

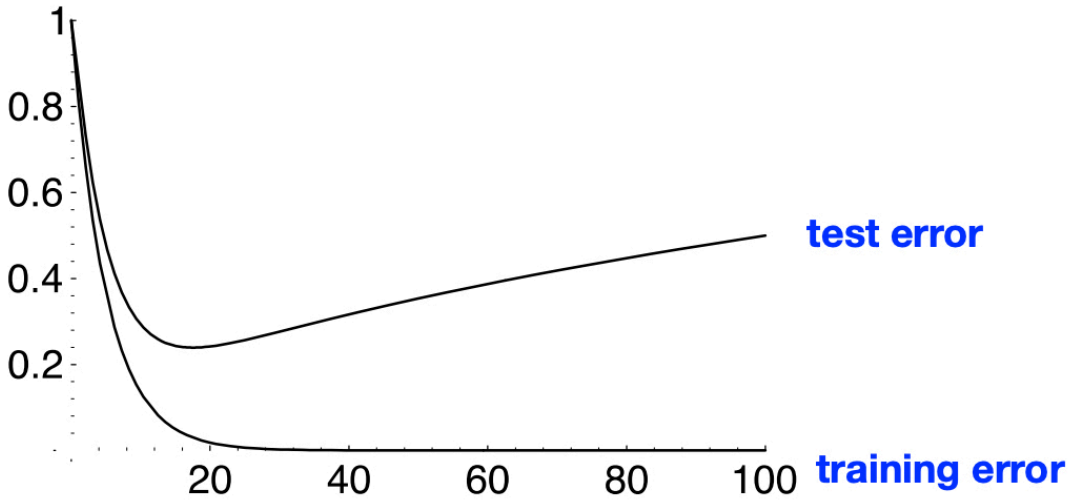
Finding Outliers with Boosting

examples with most weight are often outliers (mislabeled and/or ambiguous)

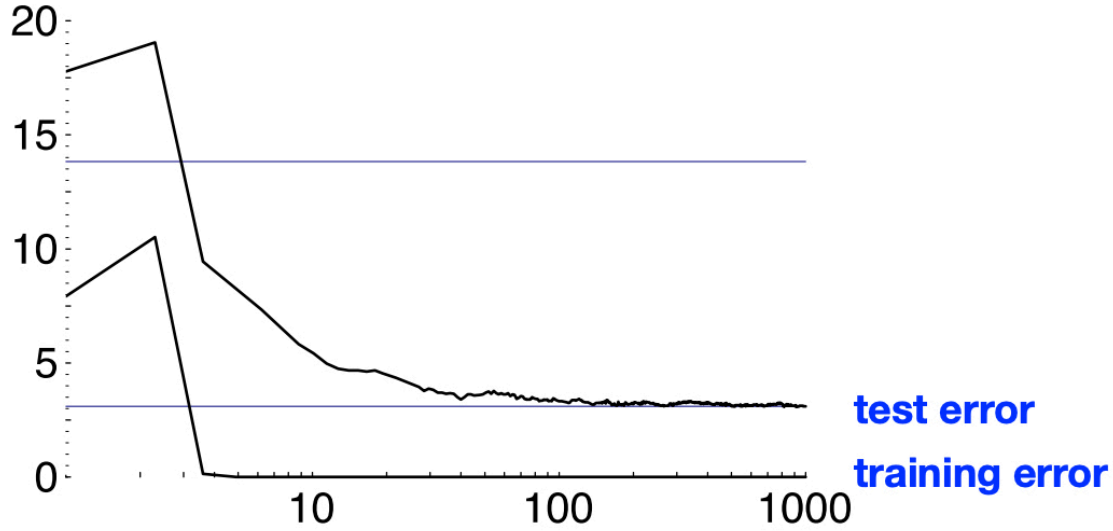
- `I'm trying to make a credit card call` (Collect)
- `hello` (Rate)
- `yes I'd like to make a long distance collect call please` (CallingCard)
- `calling card please` (Collect)
- `yeah I'd like to use my calling card number` (Collect)
- `can I get a collect call` (CallingCard)
- `yes I would like to make a long distant telephone call and have the charges billed to another number` (CallingCard DialForMe)
- `yeah I can not stand it this morning I did oversea call is so bad` (BillingCredit)
- `yeah special offers going on for long distance` (AttService Rate)

Boosting does not overfit in practice

Theory



Practice



Xgboost: A scalable tree boosting system

T Chen, C Guestrin - Proceedings of the 22nd acm sigkdd international …, 2016 - dl.acm.org

Tree boosting is a highly effective and widely used machine learning method. In this paper, we describe a scalable end-to-end tree boosting system called XGBoost, which is used widely by data scientists to achieve state-of-the-art results on many machine learning …

☆ Cited by 3057 Related articles All 14 versions

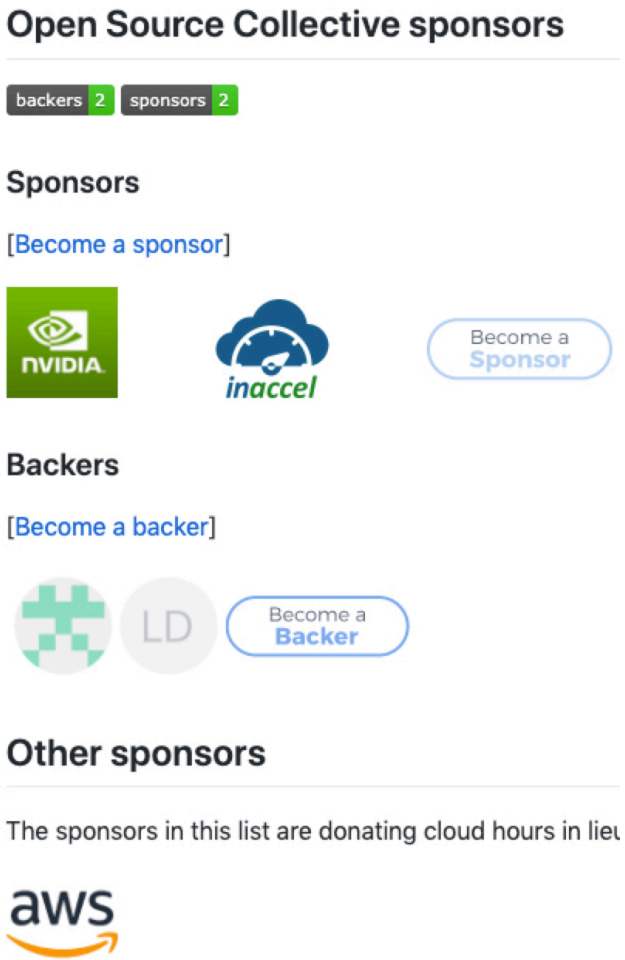
dmlc

XGBoost eXtreme Gradient Boosting

Very efficient, extensive **tree boosting** code by Tianqi Chen (UW)

A standard tool in data science competitions

Easy to parallelize and run on billion-scale problems

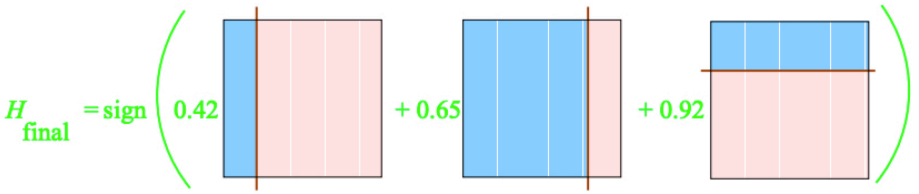


Final words on Boosting

dmlc XGBoost eXtreme Gradient Boosting

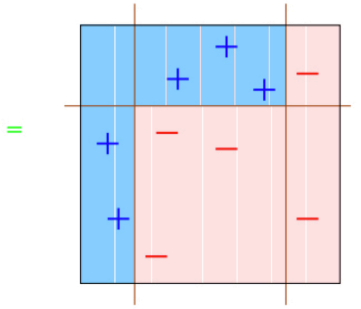
Advantages

- **Fast** and **simple**
- **Flexible**: can work with any weak learner
- Handles **various feature types**
- Strong **theoretical foundations**

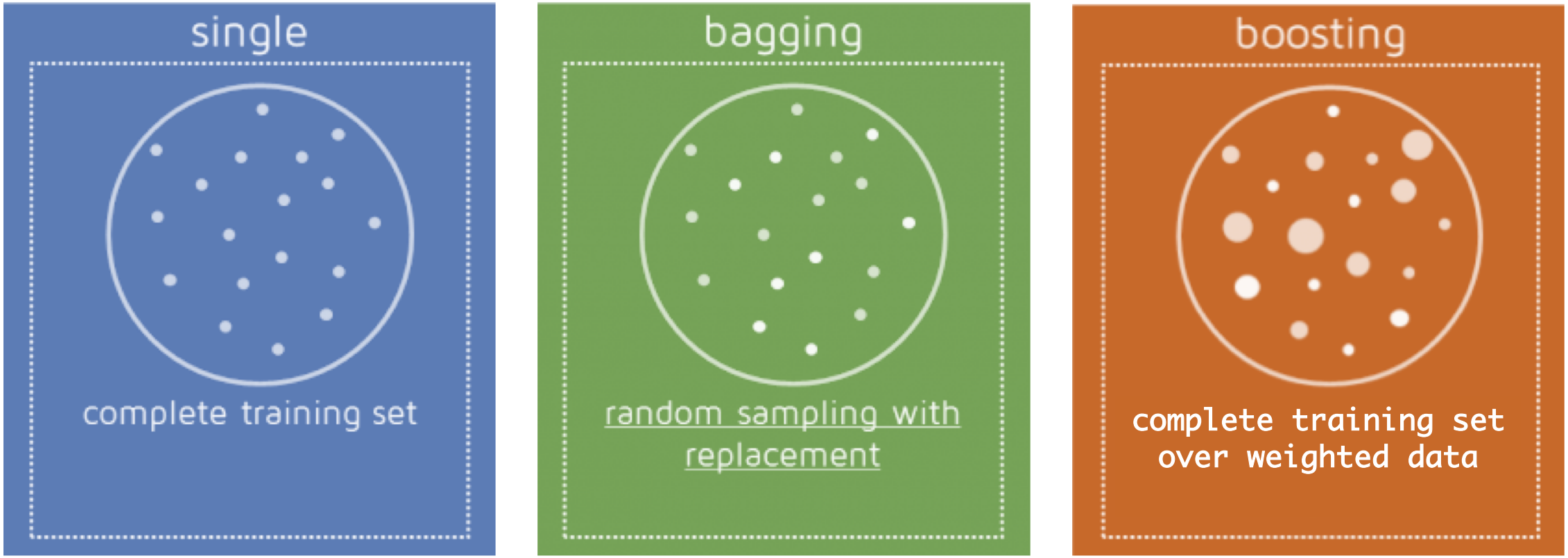


Caveats

- **Overfits** if weak learner is too complex

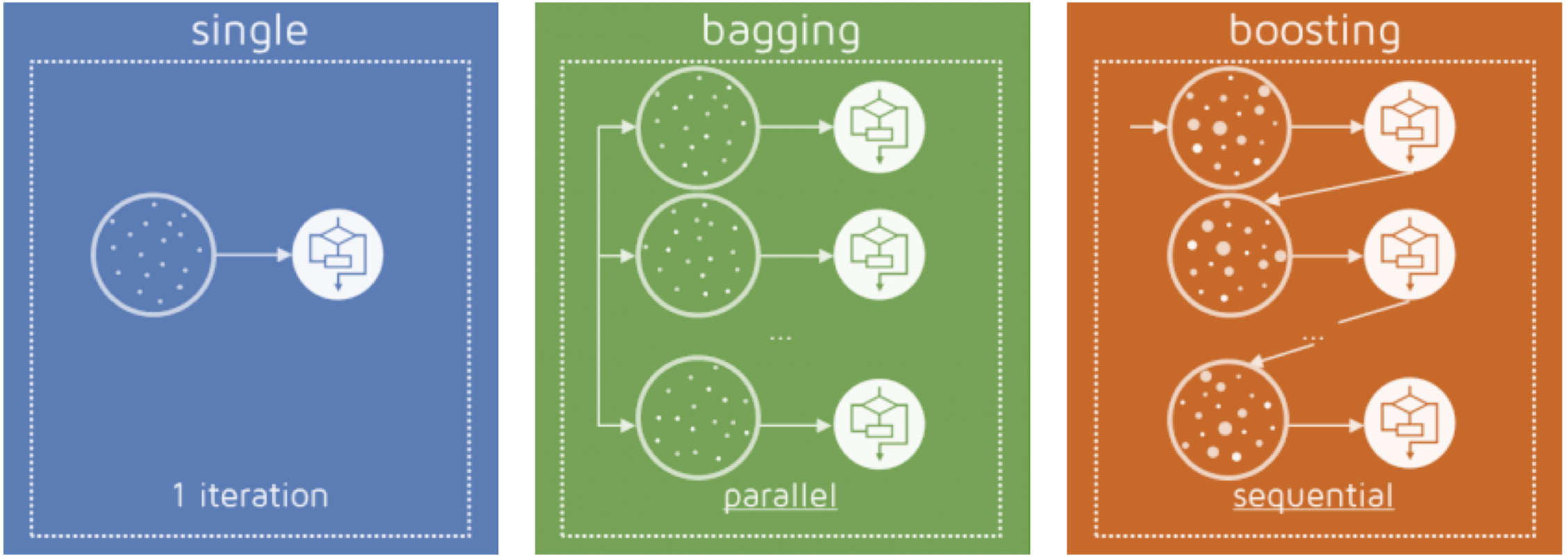


Comparing Ensembles: Data



Based on https://quantdare.com/what-is-the-difference-between-bagging-and-boosting/

Comparing Ensembles: Training



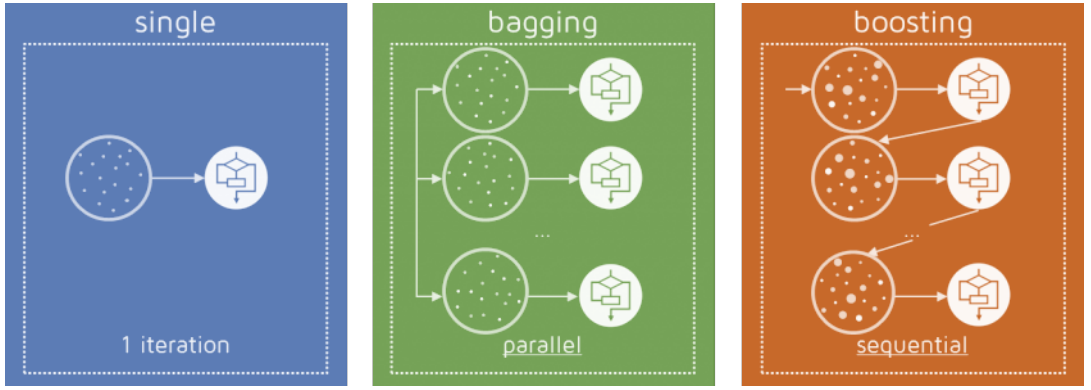
Based on https://quantdare.com/what-is-the-difference-between-bagging-and-boosting/

Boosting vs Bagging

- Bagging is typically faster, but may get a smaller error reduction (not by much).
- Bagging works well with “reasonable” classifiers.
- Boosting works with very simple classifiers.
  - E.g., Boostexter - text classification using decision stumps based on single words.
- Boosting may have a problem if a lot of the data is mislabeled, because it will focus on those examples a lot, leading to overfitting.

Slide from Edith Law at U Waterloo

Recap



- Ensemble methods are a **must-try**

- Bagging: average models trained on bootstrap samples
- Good bagging: **Random Forests**

- Boosting: sequence of models that correct for remaining mistakes
- Great boosting: **xgboost**