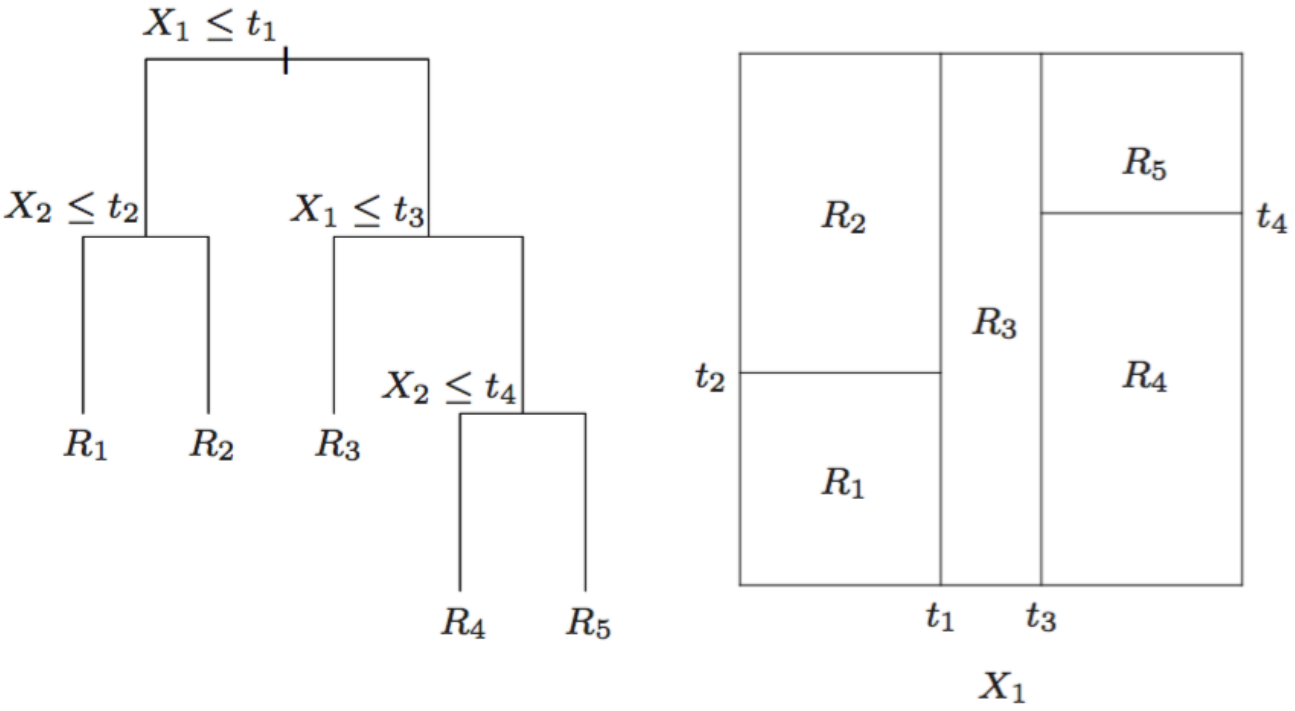
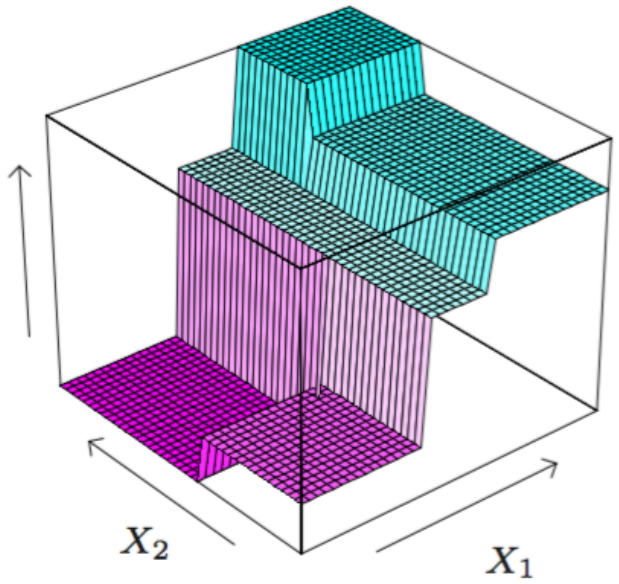
Trees

Trees

Example: binary tree with splits along axes

$X \in \mathbb{R}^d$





$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m).$$

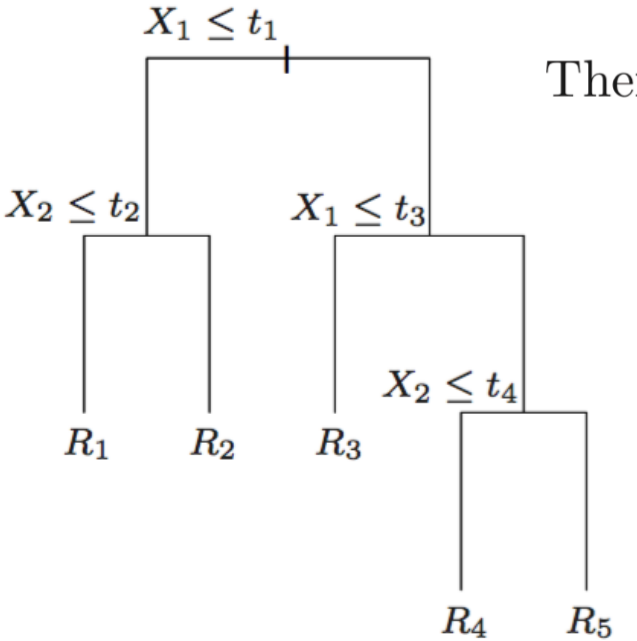
Regression Trees

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m).$$

$$\hat{c}_m = \text{ave}(y_i | x_i \in R_m).$$

Binary tree with splits along axes.

How do you build the tree / find the splits?



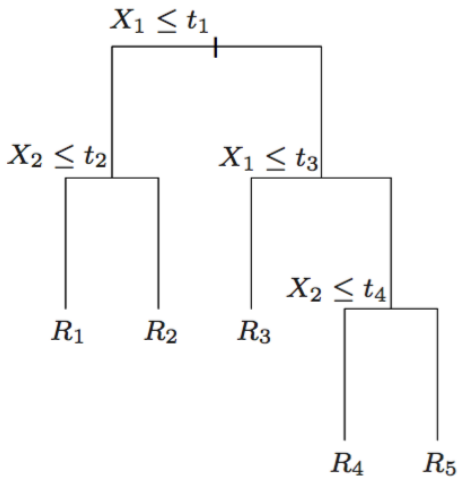
$$R_1(j,s) = \{X | X_j \le s\} \text{ and } R_2(j,s) = \{X | X_j > s\}.$$

Then we seek the splitting variable j and split point s that solve

$$\min_{j,\ s}\left[\min_{c_1}\sum_{x_i \in R_1(j,s)}(y_i - c_1)^2 + \min_{c_2}\sum_{x_i \in R_2(j,s)}(y_i - c_2)^2\right].$$

Learning decision trees

- Start from empty decision tree
- Split on next best attribute (feature)
  - Use, for example, information gain to select attribute
  - Split on $\arg\max_i IG(X_i) = \arg\max_i H(Y) - H(Y \mid X_i)$
- Recurse
- Prune

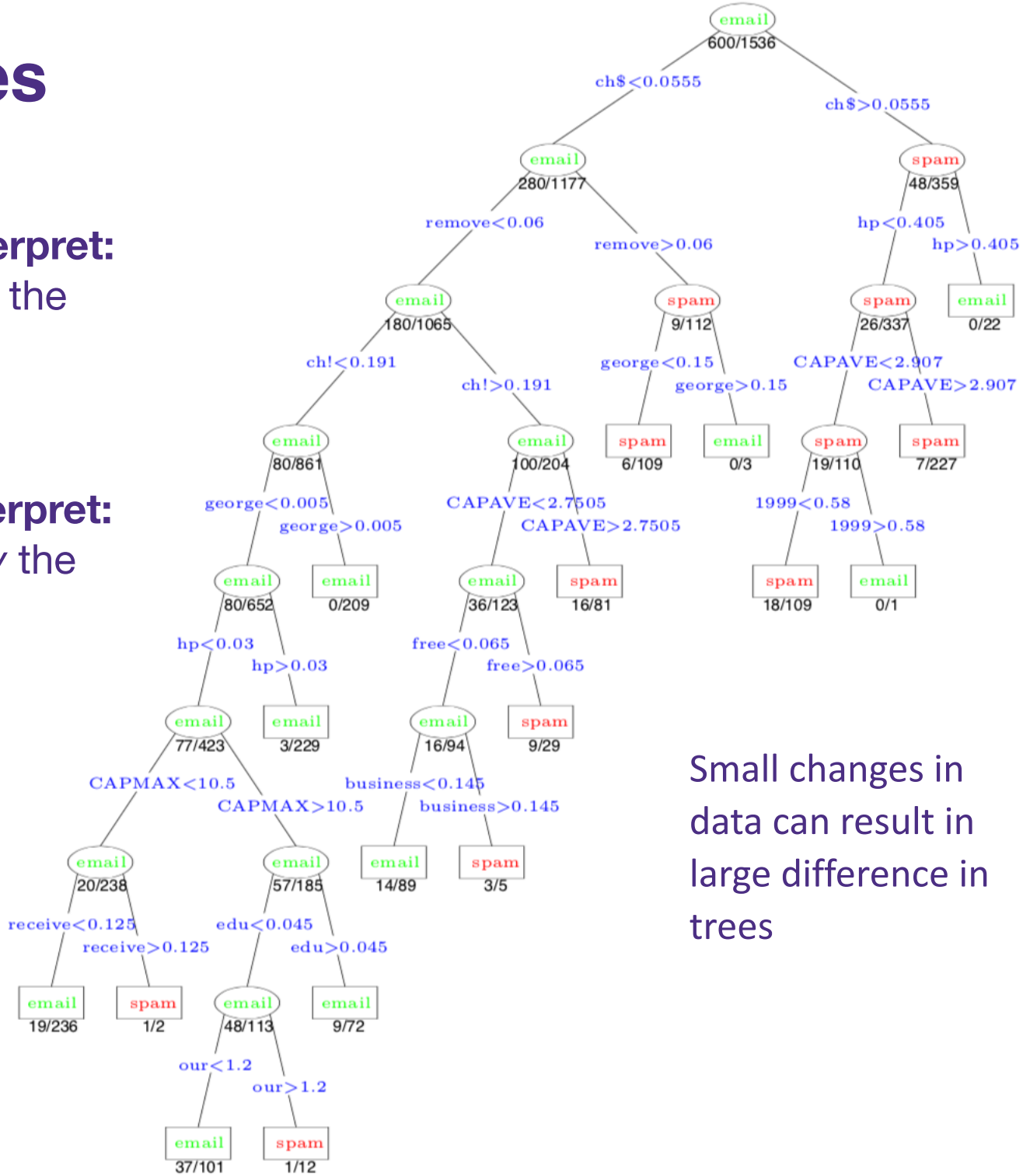


$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m).$$

Decision Trees

Trees are easy to interpret:
- You can explain *how* the classifier came to the conclusion it did

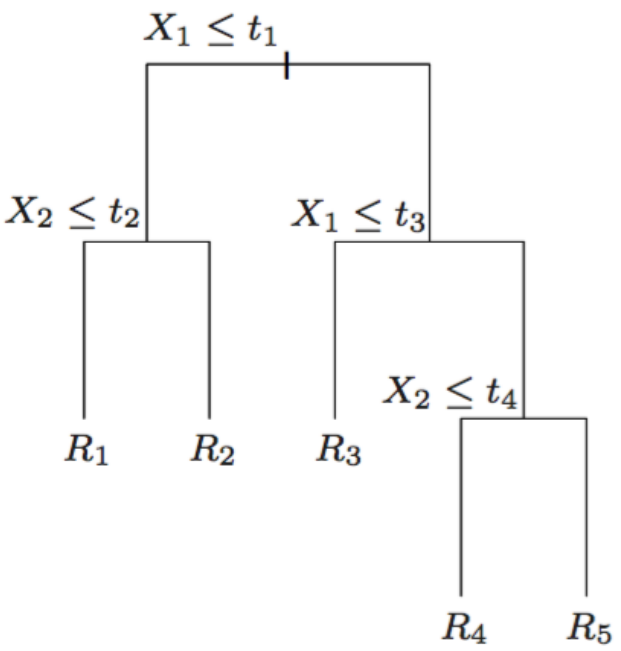
Trees are hard to interpret:
- Tough to explain *why* the classifier came to the conclusion it did



Small changes in data can result in large difference in trees

Trees

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m).$$



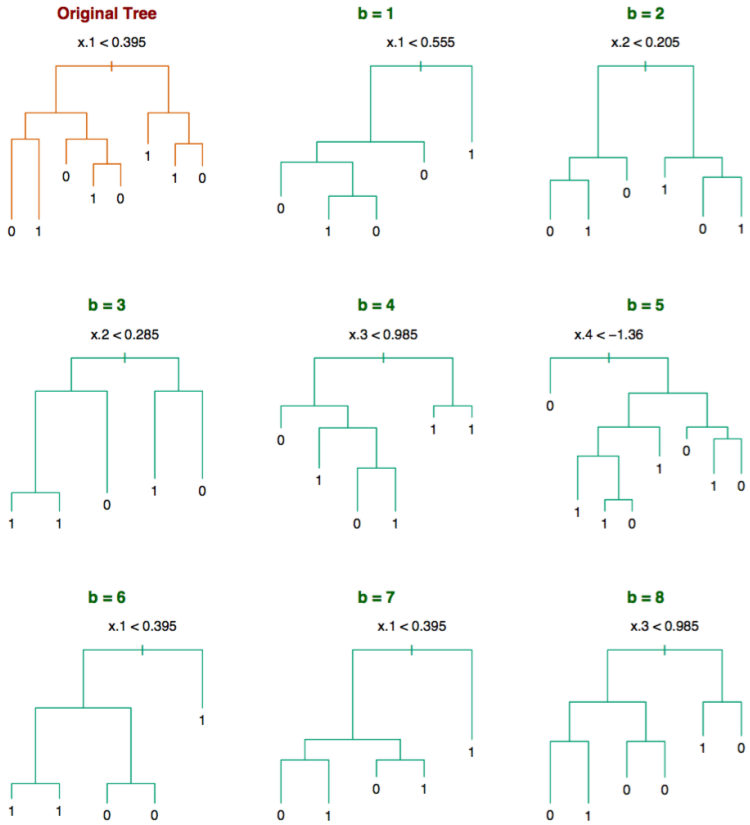
- Trees
  - **have low bias, high variance**
  - deal with categorial variables well
  - intuitive, interpretable
  - good software exists
  - Some theoretical guarantees

Random Forests

Random Forests

Tree methods have **low bias** but **high variance**.

One way to reduce variance is to construct a lot of “lightly correlated” trees and average them:



“Bagging:” Bootstrap aggregating

Random Forests

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 recursively 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.

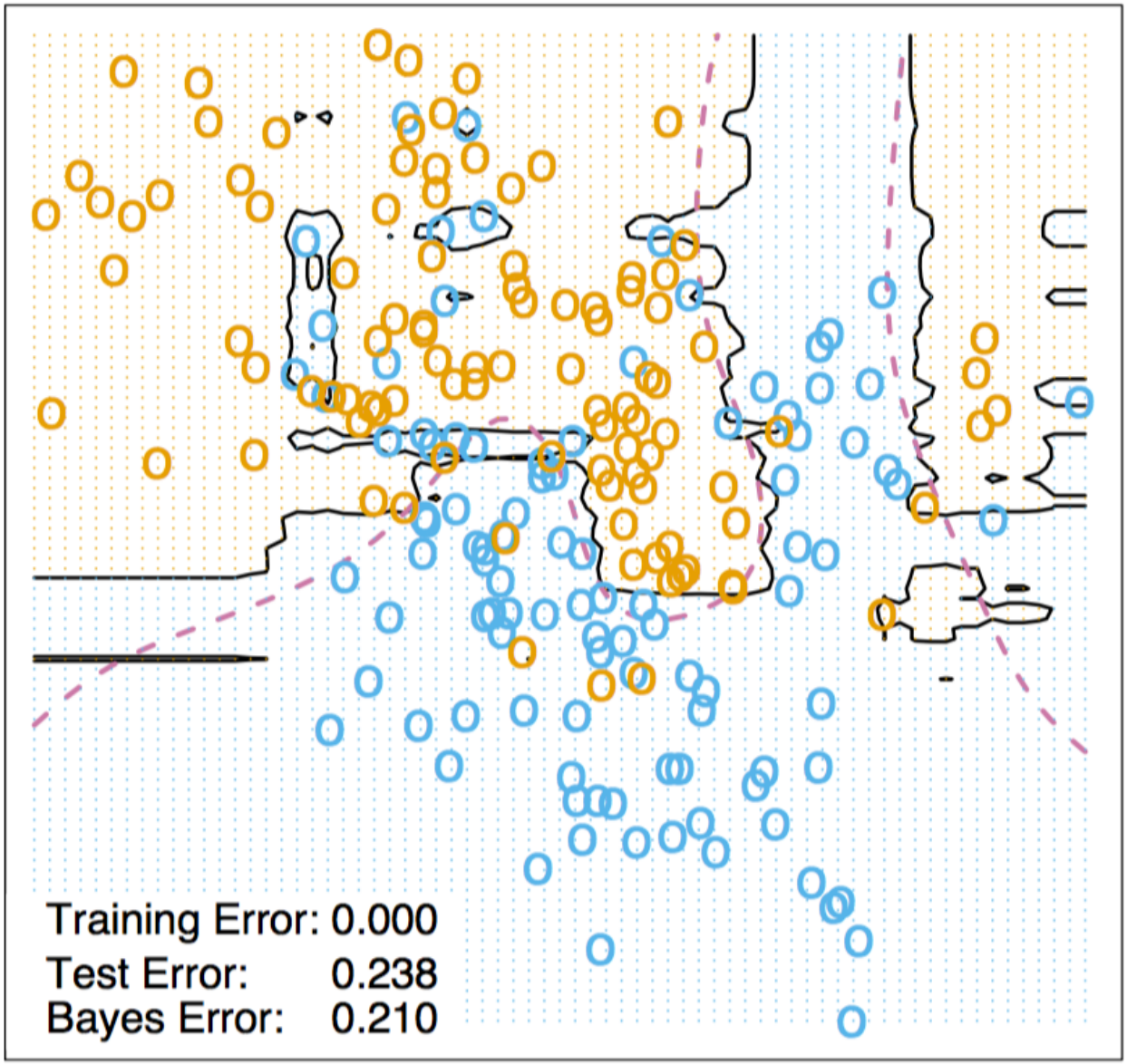
2. Output the ensemble of trees $\{T_b\}_1^B$.

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 $\hat{C}_{\text{rf}}^B(x) = \textit{majority vote}\ \{\hat{C}_b(x)\}_1^B$.

Random Forest - Decision Boundary Example



Random Forest

Given random variables $Y_1, Y_2, \ldots, Y_B$ with
$\mathbb{E}[Y_i] = y$, $\mathbb{E}[(Y_i - y)^2] = \sigma^2$, $\mathbb{E}[(Y_i - y)(Y_j - y)] = \rho\sigma^2$

σ^2 **Variance of individual predictor**

Assume bias = 0

$\rho\sigma^2$ **Correlation between predictors**

The Yi's are identically distributed but **not** independent

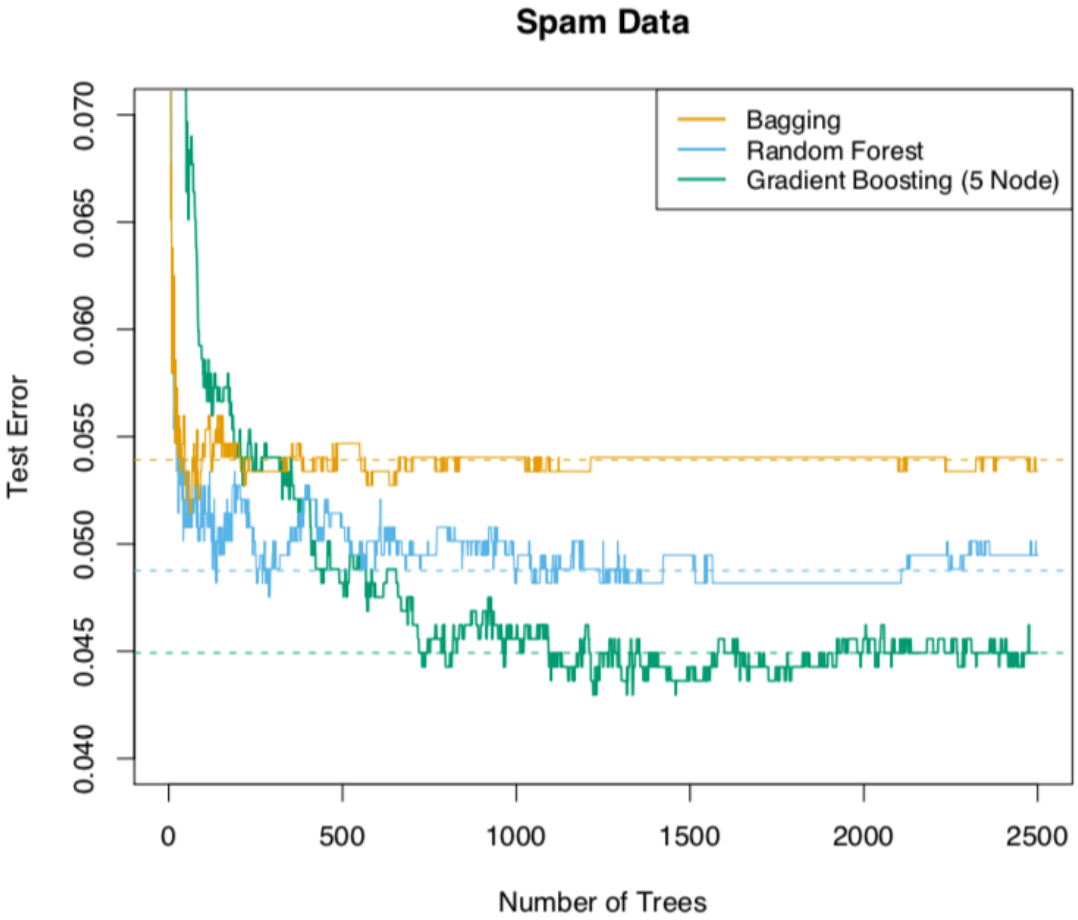
$$\mathbb{E}[(\frac{1}{B}\sum_{i=1}^{B} Y_i - y)^2] = \mathbb{E}\left[\left(\frac{1}{B}\sum_i (Y_i - y)\right)^2\right]$$

$$= \frac{1}{B^2}\mathbb{E}\sum_i (Y_i - y)^2 + \frac{1}{B^2}\sum_{i \neq j} \mathbb{E}[(Y_i - y)(Y_j - y)]$$

$$= \frac{1}{B}\sigma^2 + \frac{1}{B^2}B(B-1)\rho\sigma^2 \xrightarrow{B \to \infty} \rho\sigma^2$$

Random Forest

The power of weakly correlated predictors:



Bagging: Averaged trees trained on bootstrapped datasets that used **all d variables**

Random forest: Averaged trees trained on bootstrapped datasets that used **m < d random variables**

Gradient boosting: ignore for now

Takeaway: reducing correlation improves performance!

Random Forests

- Random Forests
  - **have low bias, low variance**
  - deal with categorial variables well
  - not that intuitive or interpretable
  - Notion of confidence estimates
  - good software exists
  - Some theoretical guarantees
  - **works well with default hyperparameters**