University of Amsterdam
Amsterdam School of Economics

Machine Learning for Econometrics

Lecture 5: Random Forest

Yi He

November 29, 2022

Plan for Today

1. Introduction
2. Bootstrap Aggregating
3. Random Forest
4. Bagging VS Boosting

Introduction

Selling A Property: Thought Experiment

Suppose you have a random sample of house prices P_i in million euros such that

$$X_i = \log P_i \overset{iid}{\sim} N(\mu, 1), \quad i = 1, \ldots, n,$$

with

- a unknown parameter μ
- an average log-price $\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \sim N(\mu, 1/n)$.

Suppose you are deciding whether to sell a house at a known bid price $P_{\text{bid}} = \exp(x) > 0$ in million euros based on

$$f(x;\mu) = \mathbb{P}(\bar{X} \leq x) = \mathbb{P}(\sqrt{n}(\bar{X} - \mu) \leq \underbrace{\sqrt{n}(x - \mu)}_{c(x;\mu)}) = \Phi(c(x;\mu))$$

Unstable Estimator: Toy Example

The unbiased estimator

$$\widetilde{f}(x; \bar{X}) = \mathbb{1}[\bar{X} \leq x].$$

is unstable in the sense a change in $\bar{X}$ may cause large changes in $\widetilde{f}(x; \bar{X})$:

If there is a small change in $\bar{X}$, for instance,

$$\bar{X} = 0.5 \Rightarrow \bar{X} = 0.5 + \delta \text{ for } \delta > 0$$

then there would be a jump of the predicted value at $x = 0.5$ regardless the size of δ:

$$\widetilde{f}(0.5; 0.5) = 1 \Rightarrow \widetilde{f}(0.5; 0.5 + \delta) = 0.$$

A Stable Estimator: First Thought

Recall that

$$f(x;\mu) = \Phi(c(x;\mu)), \quad c(x;\mu) = \sqrt{n}(x-\mu)$$

Consider the estimator

$$f(x;\bar{X}) = \Phi(c(x;\bar{X})) = \Phi(c(x;\mu) - Z) = \Phi(-(Z - c(x;\mu))),$$

where

$$c(x;\bar{X}) = \sqrt{n}(x-\bar{X}) = \underbrace{\sqrt{n}(x-\mu)}_{c(x;\mu)} - \underbrace{\sqrt{n}(\bar{X}-\mu)}_{Z}$$

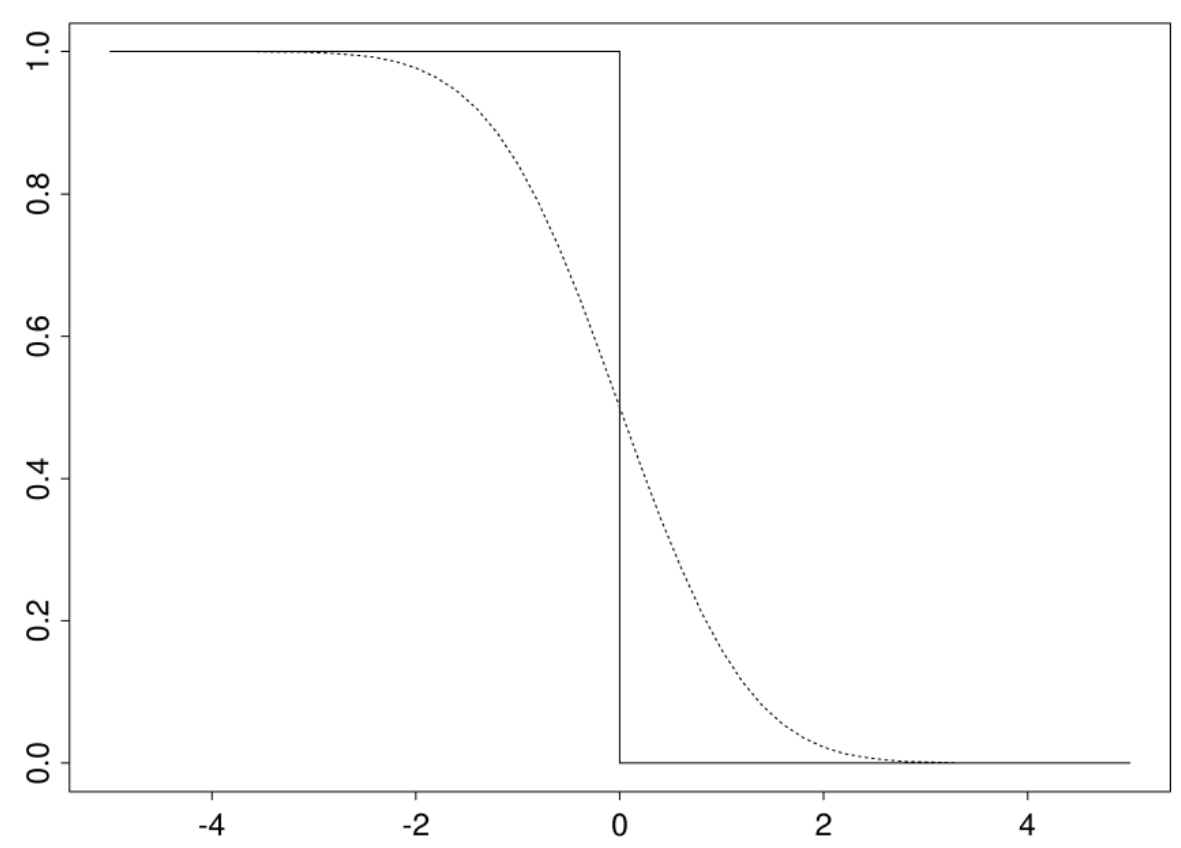
Since $|\Phi'|$ is bounded, a small change in $\bar{X}$ only results in a small change in $f(x;\bar{X})$.

Compare with the unstable estimator

$$\widetilde{f}(x;\bar{X}) = \mathbb{1}[\bar{X} \leq x] = \mathbb{1}[Z \leq c(x;\mu)] = \mathbb{1}[Z - c(x;\mu) \leq 0].$$

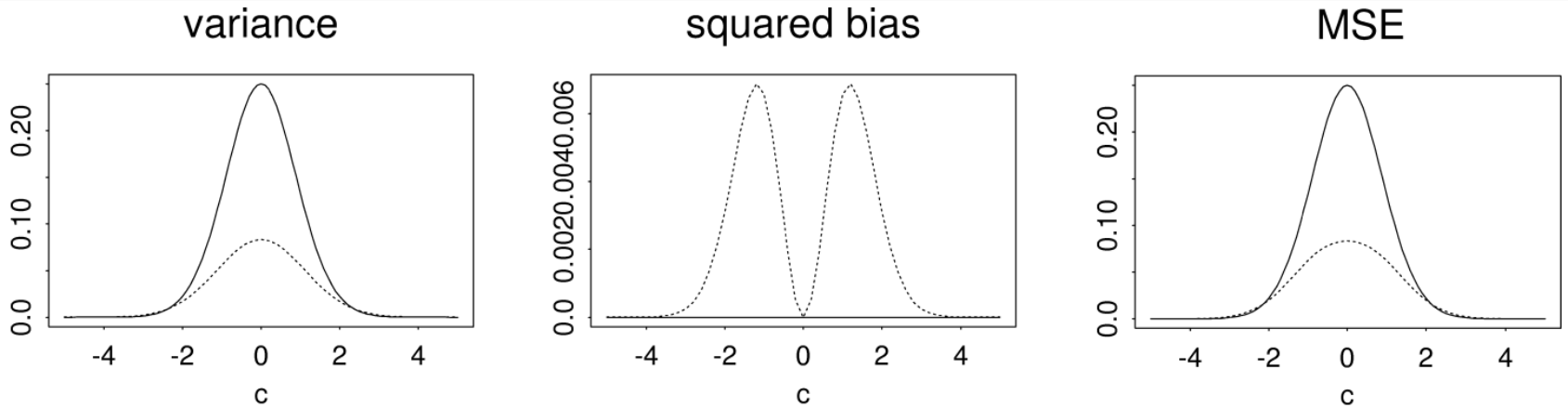
Smoothing the Indicator Function

The stable estimator smooths the indicator function $\mathbb{1}[a \leq 0]$ to be $\Phi(-a)$



Solid $= \mathbb{1}[a \leq 0]$, Dashed $= \Phi(-a)$

Smoothing Reduces Variance



Variance, Squared Bias and MSE as functions of $c = c(x; \mu)$

- Small bias, if any, but substantial variance reduction

How to generate a (more) stable estimator, in general?

Bootstrap Aggregating

Bagging

- Bagging = Bootstrap aggregating
- Draw n^* data points randomly with replacement from the training database

$$S = \{Y_i, X_i : i = 1, \ldots, n\}$$

- Repeat M times to extract the bootstrap data sets

$$S_m^* = \{Y_i^*, X_i^* : i = 1, \ldots n^*\}, \ m = 1, \ldots, M$$

- Fit a base estimator $f_m(x)$ to every bootstrap sample S_m^* and output the ensemble estimator

$$\widehat{f}(x) = \frac{1}{M} \sum_{m=1}^{M} f_m(x).$$

- Bagging uses $n^* = n$ by default.

Bootstrap Principle

- The original data set S is a random sample from some population distribution
- Conditioning on S, each bootstrap sample S_m^* is a random sample from the empirical distribution

$$\mathbb{P}(Y^*, X^* = Y_i, X_i \mid S) = \frac{1}{n}, \quad 1 \le i \le n.$$

when Y_i, X_i are distinct; otherwise sum up the probability masses at the same point.

- $\widehat{\theta} = \frac{1}{n}\sum_{i=1}^{n} h(Y_i, X_i)$ with mean $\theta = \mathbb{E}h(Y_i, X_i)$
- $\theta^* = \frac{1}{n^*}\sum_{i=1}^{n^*} h(Y_i^*, X_i^*)$ with conditional mean $\widehat{\theta}$
- Bootstrap principle: under some regularity conditions

$$\mathbb{P}\left(\sqrt{n^*}\left(\theta^* - \widehat{\theta}\right) \le x | S\right) \approx \mathbb{P}\left(\sqrt{n}\left(\widehat{\theta} - \theta\right) \le x\right)$$

uniformly for x.

Bagging Is Smoothing: Toy Example

- The unstable estimator for the toy example

$$\widetilde{f}(x; \bar{X}) = \mathbb{1}[\bar{X} \leq x]$$

- By LLN, the bagging estimator

$$\widehat{f}(x) = \frac{1}{M} \sum_{m=1}^{M} \mathbb{1}[\bar{X}_m^* \leq x] \approx \mathbb{P}\left(\bar{X}^* \leq x \mid S\right), \text{ as } M \to \infty,$$

  where $\bar{X}_m^*$ is the m-th bootstrap average

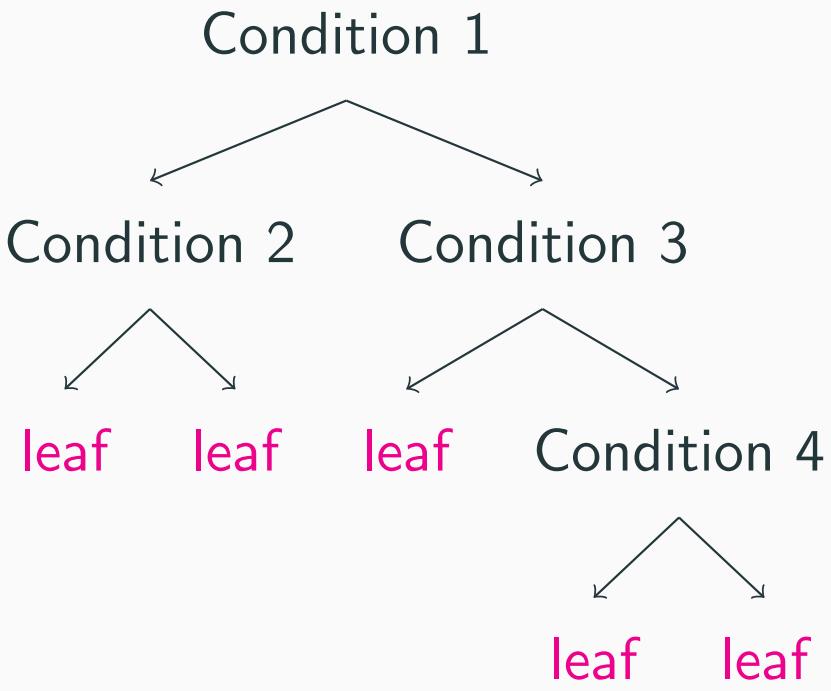
- Bootstrap principle:

$$\begin{aligned} \mathbb{P}\left(\bar{X}^* \leq x \mid S\right) =& \mathbb{P}\left(\sqrt{n}\left(\bar{X}^* - \bar{X}\right) \leq \sqrt{n}\left(x - \bar{X}\right) \mid S\right) \\ \approx& \Phi\left(\sqrt{n}\left(x - \bar{X}\right)\right) = \underbrace{f(x; \bar{X})}_{\text{stable estimator}}, \end{aligned}$$

  where Φ is the distribution function of $\sqrt{n}(\bar{X} - \mu) \sim \mathcal{N}(0, 1)$.

Random Forest

Unstable Model: Decision Tree



- Make predictions at the terminal nodes (leafs)
- Interior node = a condition for a **single** feature $X_j < \theta$
- Condition TRUE $\rightarrow$ left child
- Condition FALSE $\rightarrow$ right child
- Decision stump is a one-split decision tree

Bagging with Decision Trees

- Tree estimators are unstable

$$f_m(x; \{\widehat{\mathcal{R}}_\tau, \widehat{c}_\tau\}) = \sum_{\tau=1}^{\mathrm{T}} \widehat{c}_\tau \mathbb{1}[x \in \widehat{\mathcal{R}}_\tau]$$

  especially around the boundaries of $\widehat{\mathcal{R}}_\tau$
- Grow/fit a decision tree $f_m(x)$ to every bootstrap sample S_m^* and then aggregate
- Regression: predicted value is the average across the trees
- Classification: regression problems with one-hot encoder

CART Algorithm

- Generate a (random) pool of candidate splits with

$$\{(j^{(b)}, \alpha^{(b)}) : b = 1, \ldots, B\},$$

where the $j^{(b)} \in \{1, \ldots, d\}$ is a feature index, $\alpha^{(b)} \in (0, 1)$ is a probability threshold level

- For each split b, divide the input feature space $\mathcal{X} \subset \mathbb{R}^d$ into two sub-regions:

$$\mathcal{X}_L^{(b)} = \{x \in \mathcal{X} : x_{j^{(b)}} < \theta^{(b)}\} \text{ and } \mathcal{X}_R^{(b)} = \{x \in \mathcal{X} : x_{j^{(b)}} \geq \theta^{(b)}\},$$

where the threshold $\theta^{(b)}$ is the sample quantile of the $j^{(b)}$-th feature at level $\alpha^{(b)}$.

- For each sub-region $\mathcal{X}_\tau^{(b)}$, find the subset of target values

$$\mathcal{Y}_\tau^{(b)} = \{Y_i : X_i \in \mathcal{X}_\tau^{(b)}, 1 \leq i \leq n\}, \quad \tau \in \{L, R\}$$

- Choose the best split minimizing the total weighted 'impurity'

$$p_L^{(b)} \psi(\mathcal{Y}_L^{(b)}) + p_R^{(b)} \psi(\mathcal{Y}_R^{(b)}),$$

  where ψ is some impurity function and

$$p_\tau^{(b)} = \frac{|\mathcal{X}_\tau^{(b)}|}{|\mathcal{X}^{(b)}|}, \ \tau \in \{L, R\}.$$

- Repeat the split sequentially for every internal node until a stopping criterion is reached

Choosing the Impurity Function

Regression trees:

- Sample variance

$$\psi_2(\mathcal{Y}_\tau) = \frac{1}{|\mathcal{Y}_\tau|} \sum_{Y_i \in \mathcal{Y}} (Y_i - \bar{Y}_\tau)^2, \;\; \bar{Y}_\tau = \frac{1}{|\mathcal{Y}_\tau|} \sum_{Y_i \in \mathcal{Y}} Y_i.$$

Classification trees:

- Gini index = sum of sample variances over classes

$$\psi(\mathcal{Y}_\tau) = \sum_{k=1}^{K} \psi_2(\mathcal{Y}_\tau^{(k)}) = 1 - \sum_{k=1}^{K} (\bar{Y}_\tau^{(k)})^2$$

where

$$\mathcal{Y}_\tau^{(k)} = \{ \underbrace{Y_i^{(k)} = \mathbb{1}[Y_i = \mathcal{C}_k]}_{\text{One-Hot Encoder}} : Y_i \in \mathcal{Y}_\tau \}$$

$$\bar{Y}_\tau^{(k)} = \frac{1}{|\mathcal{Y}_\tau|} \sum_{Y_i \in \mathcal{Y}} Y_i^{(k)}, \quad k = 1, \ldots, K$$

Note that $\bar{Y}_\tau^{\mathsf{OH}} = (\bar{Y}_\tau^{(1)}, \ldots, \bar{Y}_\tau^{(K)})$ represents a distribution:

$$\bar{Y}_\tau^{(k)} \geq 0, \quad \sum_{k=1}^{K} \bar{Y}_\tau^{(k)} = 1.$$

If $\bar{Y}_\tau^{(k)} > 0$ for all k, we can also use the ~~cross~~ entropy

$$\psi(\mathcal{Y}_\tau) = \ell_{\mathsf{CE}}(\bar{Y}_\tau^{\mathsf{OH}}, \bar{Y}_\tau^{\mathsf{OH}}) = -\sum_{k=1}^{K} \log \bar{Y}_\tau^{(k)} \cdot \bar{Y}_\tau^{(k)}$$

- entropy is maximum at the uniform distribution $(1/K, \ldots, 1/K)$: the most 'impure' case

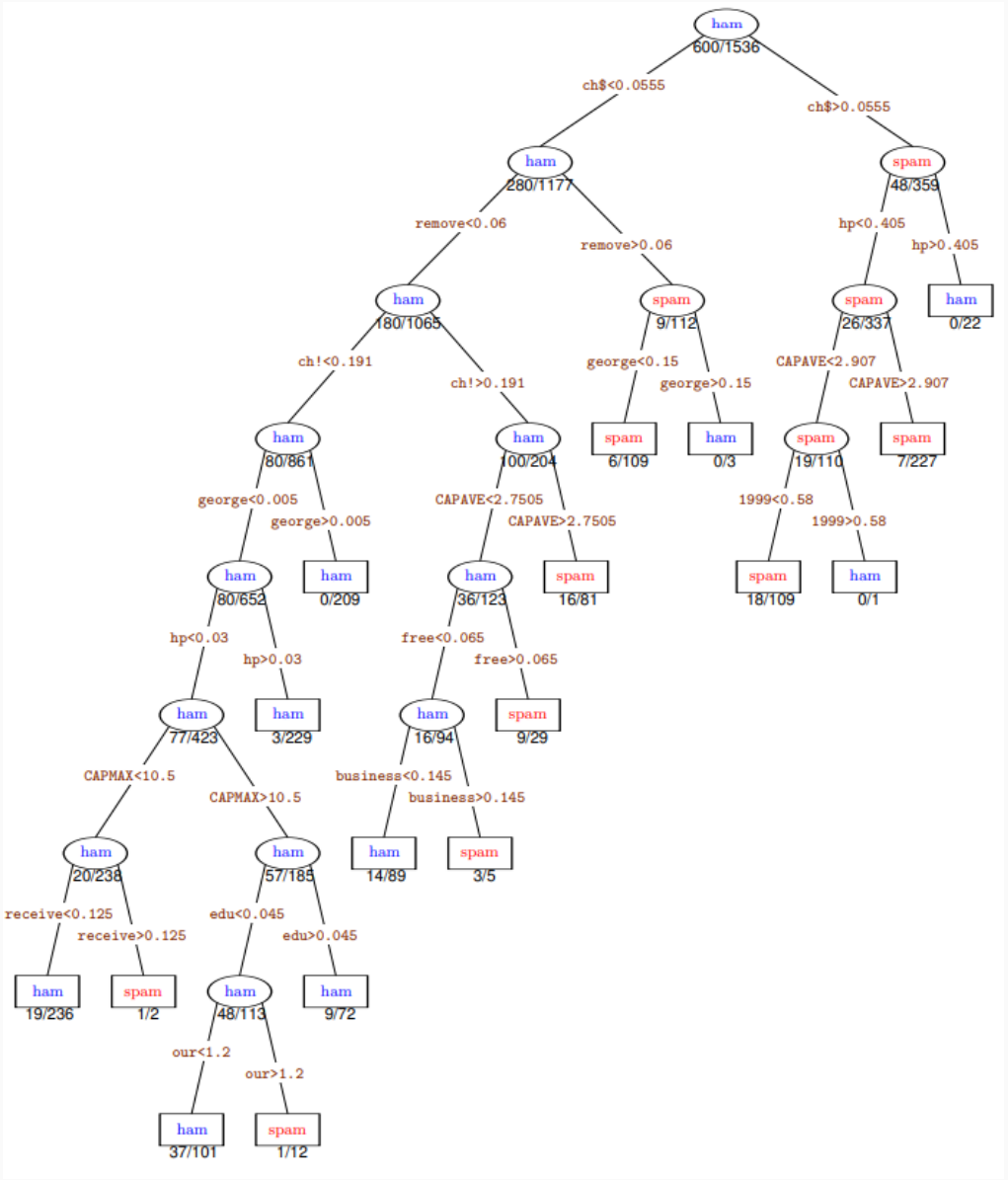


Figure 17.1 in Efron and Hastie (2016): An Example Tree

Forecasting with Decision Trees

- Eventually the terminal nodes partition the entire feature space

$$\mathcal{X} = \cup_{\tau=1}^{\mathrm{T}} \mathcal{R}_\tau, \ \mathcal{R}_\tau \cap \mathcal{R}_{\tau'} = \emptyset, \ \tau \neq \tau'$$

- The predicted target value based on the m-th tree T_m:

$$f_m(x) = \sum_{\tau=1}^{\mathrm{T}_m} c^*_{m,\tau} \mathbb{1}[x \in \mathcal{R}^*_{m,\tau}],$$

$$c^*_{m,\tau} = \frac{1}{|\mathcal{R}^*_{m,\tau}|} \sum_{\substack{X^*_i \in \mathcal{R}^*_{m,\tau} \\ Y^*_i, X^*_i \in S^*_m}} Y^*_i,$$

- Same method applies to the one-hot encoder

Decorrelating the Trees

- The decision trees share a common training data set S and thus f_m are dependent
- High correlations prevent stabilization
- One strategy to 'decorrelate' the trees is to allow only a (small) random subset of features for every split.
- For example, each split may use only a subset of $\sqrt{d}$ features randomly selected from d features.

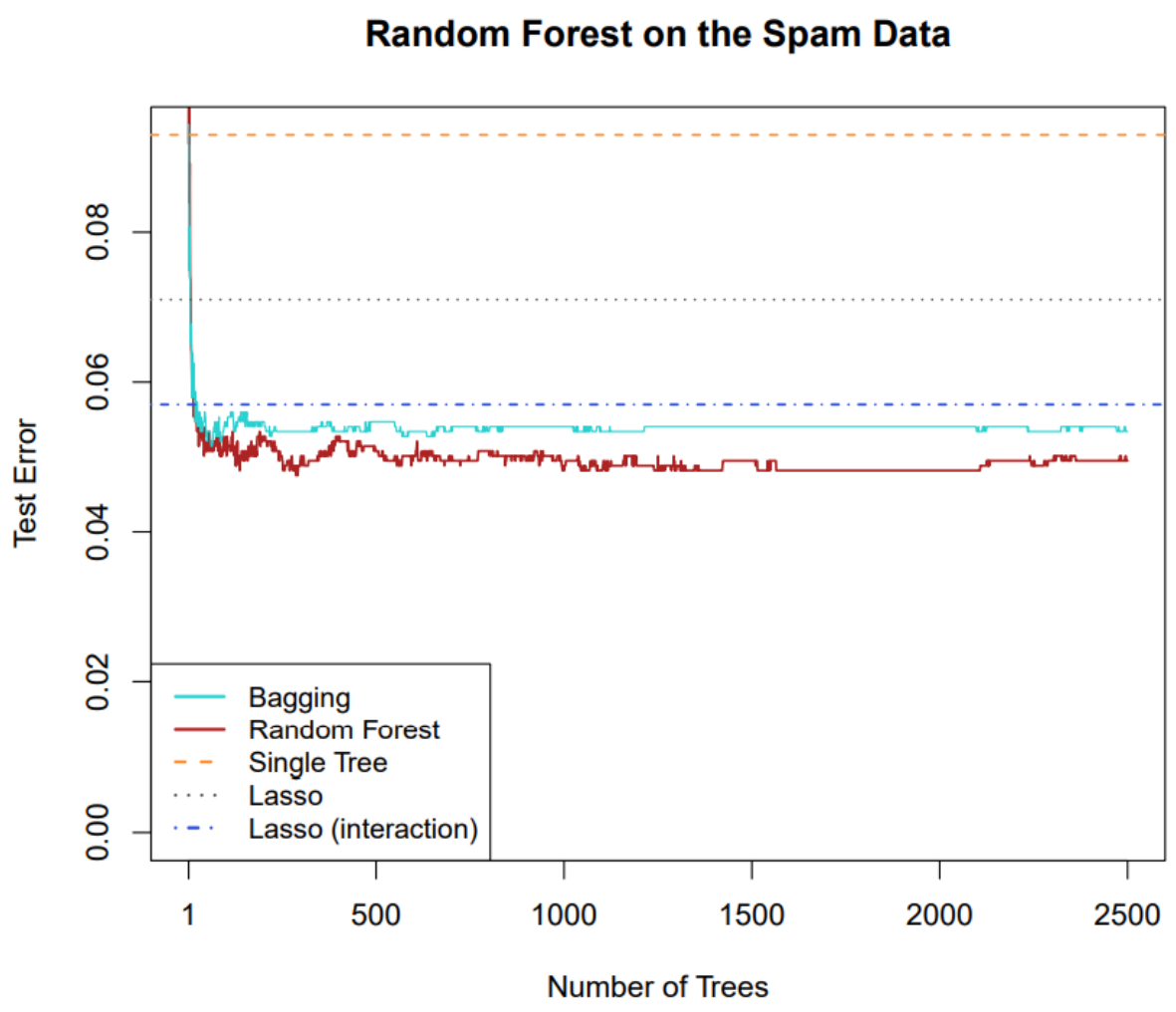


Figure 17.2 in Efron and Hastie (2016): Test misclassification error of random forests as a function of the number of trees. The red curve selects 7 of the 57 features at random as candidates for the split variable.

Bagging VS Boosting

Random VS Deterministic Weights

The empirical risk function on bootstrap sample

$$\frac{1}{n}\sum_{i=1}^{n}\ell(f(X_i^*), Y_i^*) \sim \sum_{i=1}^{n} w_i \ell(f(X_i), Y_i)$$

with exchangeable w_i such that $(nw_1, \ldots, nw_n)$ follows a multinomial distribution [Tute Q3].

The empirical risk function at m-th iteration for AdaBoost

$$\sum_{i=1}^{n} w_i^{(m)} \ell(f(X_i), Y_i),$$

where $w_i^{(m)}$ change over m and focus on the data points misclassified by previous bases.

More Remarks

Parallel VS Sequential Optimization

- Random forest grows trees separately
- Boosting requires knowing the previous bases

Number of Base Learners

- Random forest requires as large M as possible: no bias-variance tradeoff on M
- Boosting forever overfits: early stopping or shrinkage is necessary

Complexity of Trees

- Large trees in RF: bias-variance tradeoff on trees
- Stumps (or small trees) in Boosting: large bias, small variance