Machine Learning for Operations
Lecture 3: Tree-based Learning

Gah-Yi Ban

Columbia GSB
Fall 2016

Outline: Lecture 2

Statistical Learning Theory

- Generalization error
- Vapnik-Chervonenkis Theory
- Stability Theory
- Application: Newsvendor Problem

Outline: Lecture 3

Tree-based Methods

- CART: Classification and Regression Trees
- Bagging: Averaging of Trees
- Random Forest: Cleverer Averaging of Trees I
- Boosting: Cleverer Averaging of Trees II
- Application: Demand forecasting

References

Books (free pdf online!):

- James G., Witten, D., Hastie, T. and Tibshirani, R. An Introduction to Statistical Learning with Applications in R. New York: Springer, 2013. ISLA
- Friedman, J., Hastie, T., and Tibshirani, R. The elements of statistical learning. Vol. 2. Springer, Berlin: Springer series in statistics. 2001. ESLII

Paper:

- Ferreira, K.J., Lee, B.H.A., and Simchi-Levi, D. (2016) Analytics for an Online Retailer: Demand Forecasting and Price Optimization. Manufacturing Service Operations Management 18(1):69-88. [FLSL16]

Resources

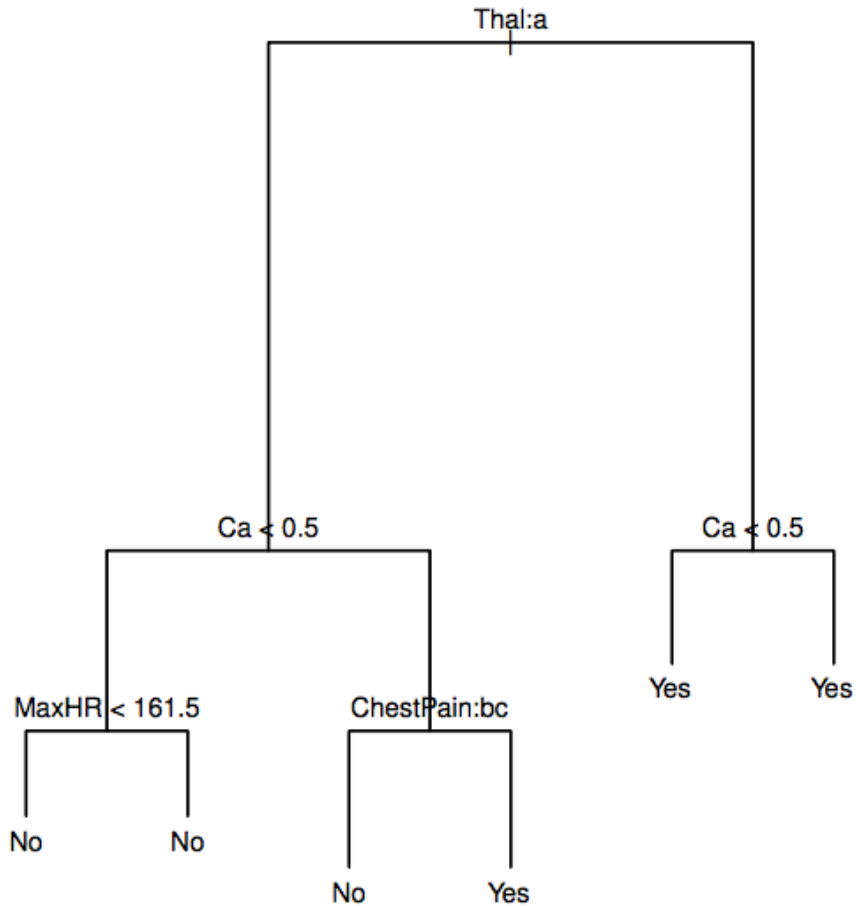
Software:

- R is the statistical computing language/environment of choice
- Freely available from: `http://cran.r-project.org/`
- Tutorials with data sets and codes available in ISLA and ESLII
- More free tutorials online

Example 1 (Classification): Predicting Heart Disease

Y: Yes if Heart Disease, No otherwise.

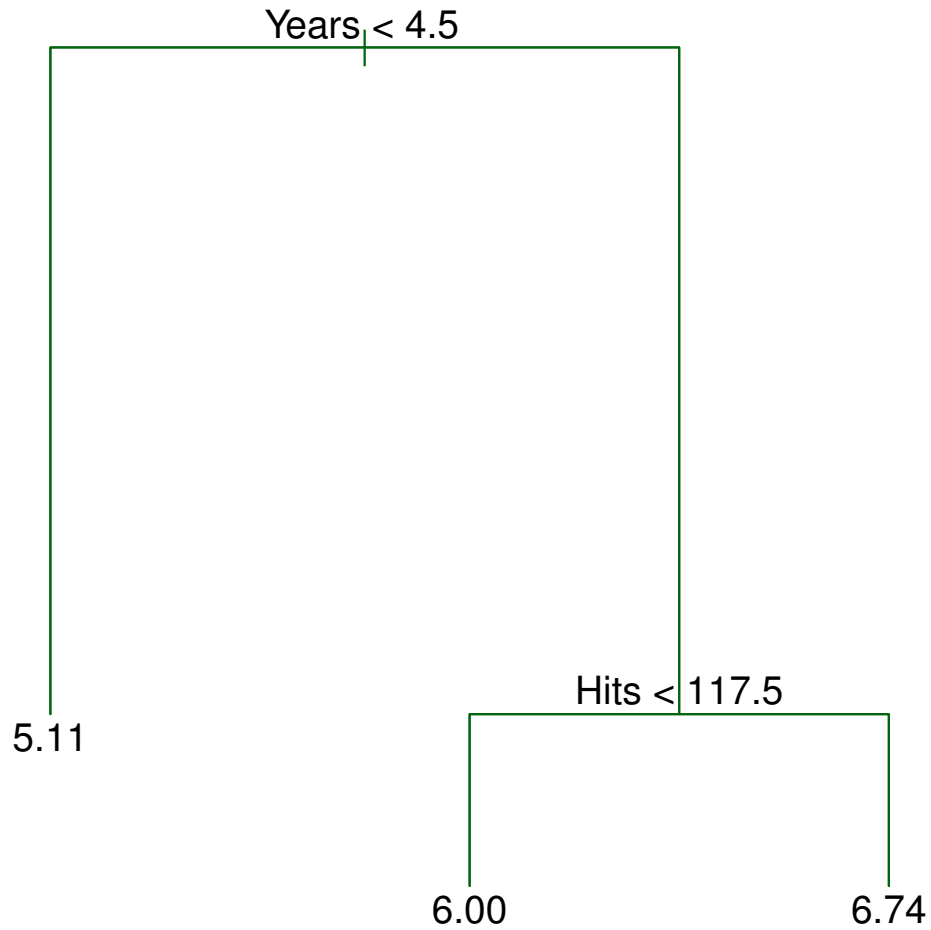
$X = \{$Thallium stress test, Ca level, Max. Heart rate, Chest Pain$\}$



Example 2 (Regression): Predicting Baseball Player Salary

Y: log-salary ('000 \$)

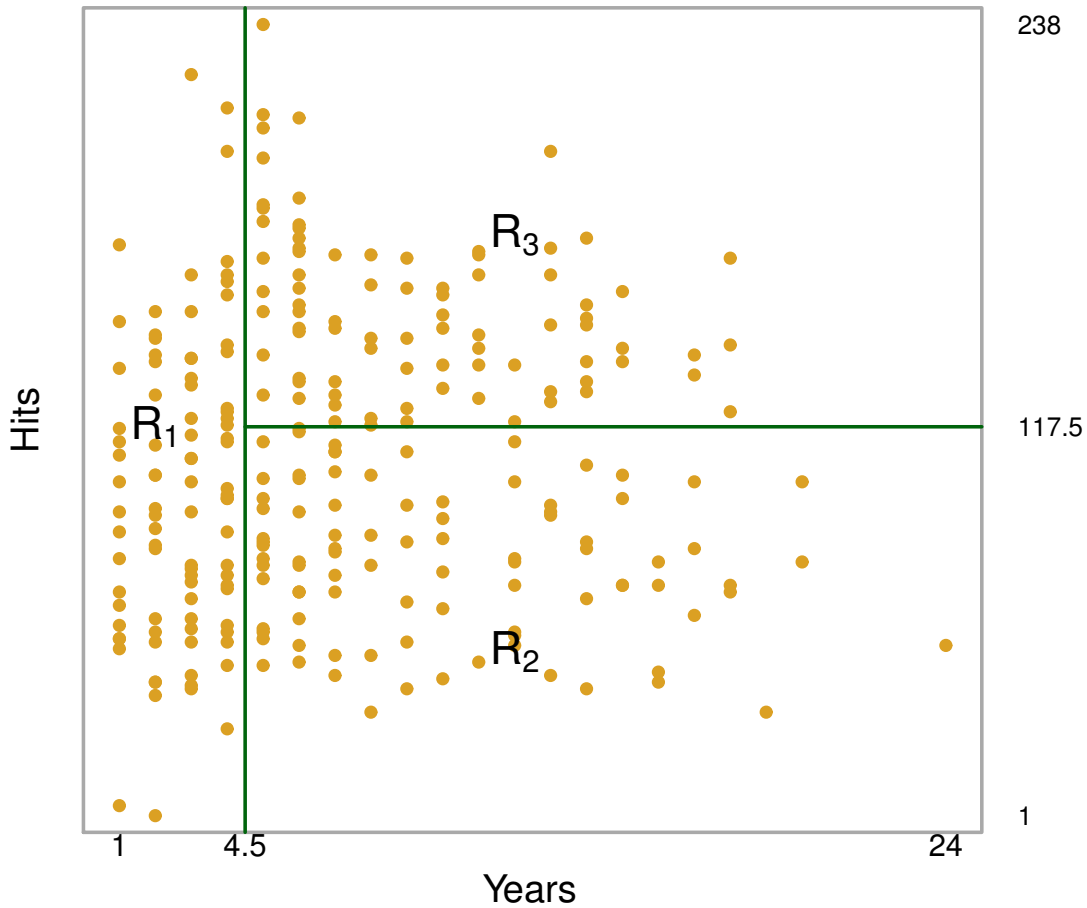
$X = \{$no. of years played, no. of hits in previous year$\}$



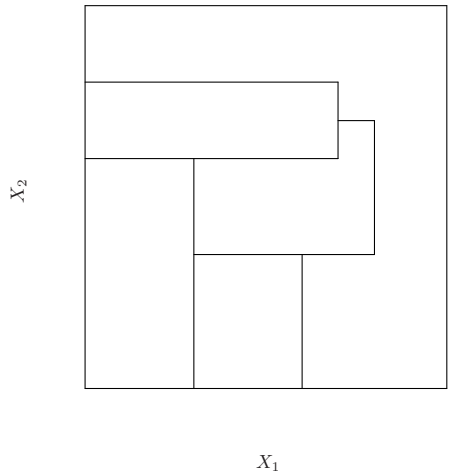
Regression Tree Example: Predicting Baseball Player Salary

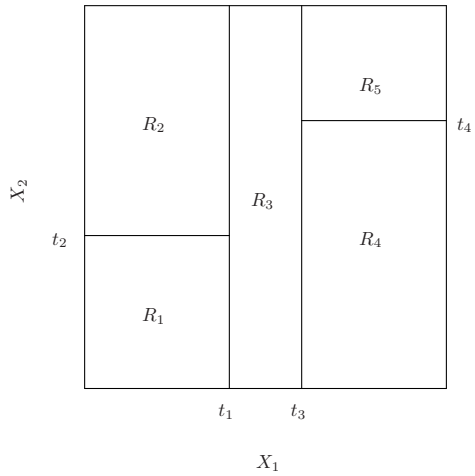
Y: log-salary ('000 $)

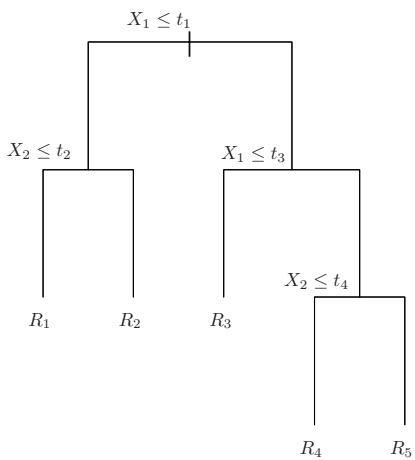
$X = \{$no. of years played, no. of hits in previous year$\}$

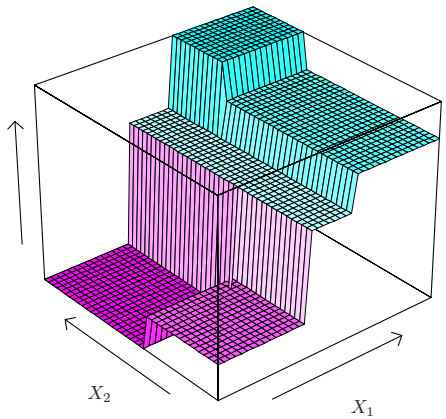


Regression Tree Schematic









Partition the feature space into a set of rectangles, then fit 0-1 (classification) or a constant (regression) in each

Components of CART

4 components:

1. Choosing the splitting predictor variable at each stage
   - Sequential optimization (i.e. finding the optimal sequence of splitting variables) is intractable
   - Thus, resort to greedy search (at each stage, find the best splitting variable)
2. Location of split
   - Split at the location that minimizes the error; straight-forward computation
3. Depth of tree (stopping rule)
   - Go too deep then can have perfect in-sample prediction (one point per rectangle), but clearly overfits
   - Too small a tree may miss important structural details
   - Cost-complexity pruning: first grow then cut (prune) back
4. Prediction at the node
   - Classification: majority class in the rectangle
   - Regression: average of outcomes in the rectangle

Steps 1 & 2. Splitting variable & location

Find splitting variable $j \in \{1, \ldots, p\}$ and a split point s, that minimize

$$\sum_{\mathbf{x}_i \in R_1(j,s)} (y_i - \hat{c}_1(s))^2 + \sum_{\mathbf{x}_i \in R_2(j,s)} (y_i - \hat{c}_2(s))^2,$$

where

$$R_1(j, s) = \{\mathbf{X} | \mathbf{X}_j \leq s\}, \quad R_1(j, s) = \{\mathbf{X} | \mathbf{X}_j > s\}$$

and

$$\hat{c}_{1,2}(j, s) = Ave(y_i | \mathbf{x}_i \in R_{1,2}(j, s)).$$

- Finding the split point s is very quick, so scan through all dimensions for the best split (j, s)
- Repeat on each of the two new regions

Step 3. Tree Size

Cost-complexity pruning:

- Grow a tree T_0 after a minimum node size. Then prune the tree.
- For any pruned tree $T \subset T_0$ (collapsing any number of internal, non-terminal nodes) and tuning parameter $\alpha \geq 0$, compute the cost-complexity criterion:

$$C_\alpha(T) = \sum_{m=1}^{|T|} \sum_{\mathbf{x}_i \in R_m} (y_i - \hat{c}_m)^2 + \alpha |T|,$$

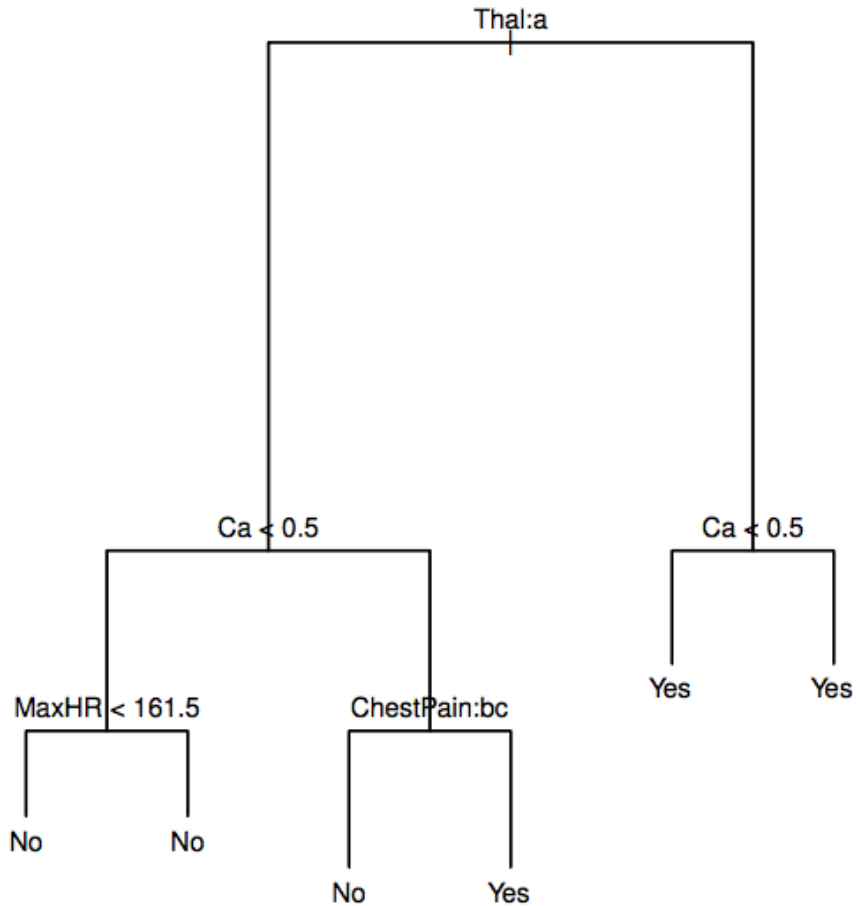
  where $|T|$ is the number of terminal nodes.
- Theorem: for each α, there is a unique smallest subtree T_α that minimizes $C_\alpha(T)$
- Find α by cross-validation

Example 1 (Classification): Predicting Heart Disease

Y: Yes if Heart Disease, No otherwise.

$X =$

{Thallium stress test, Ca level, Max. Heart rate, Chest Pain, + 9 more}

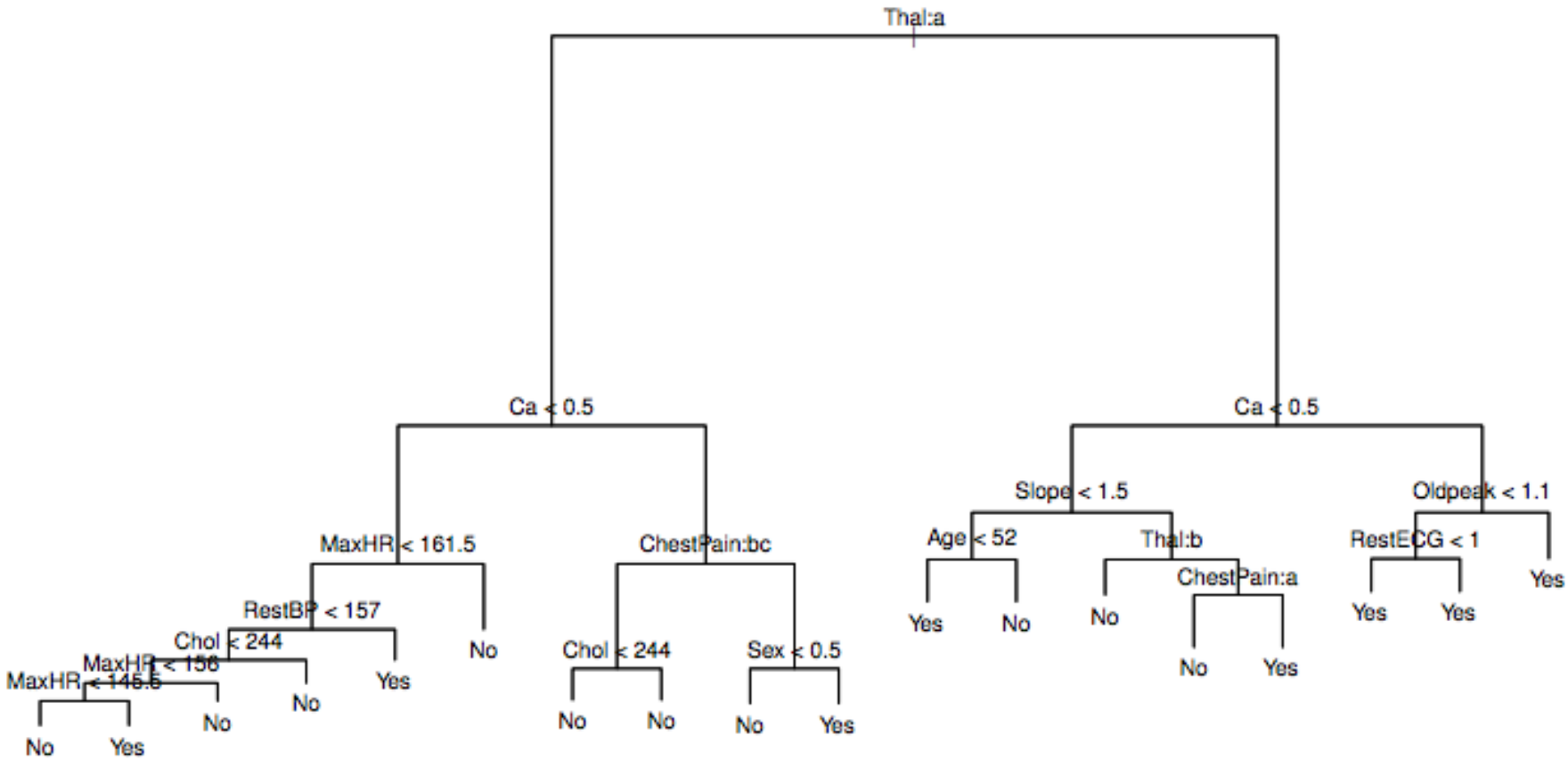


Example 1 (Classification): Predicting Heart Disease

Y: Yes if Heart Disease, No otherwise.

$X =$

{Thallium stress test, Ca level, Max. Heart rate, Chest Pain, + 9 more}

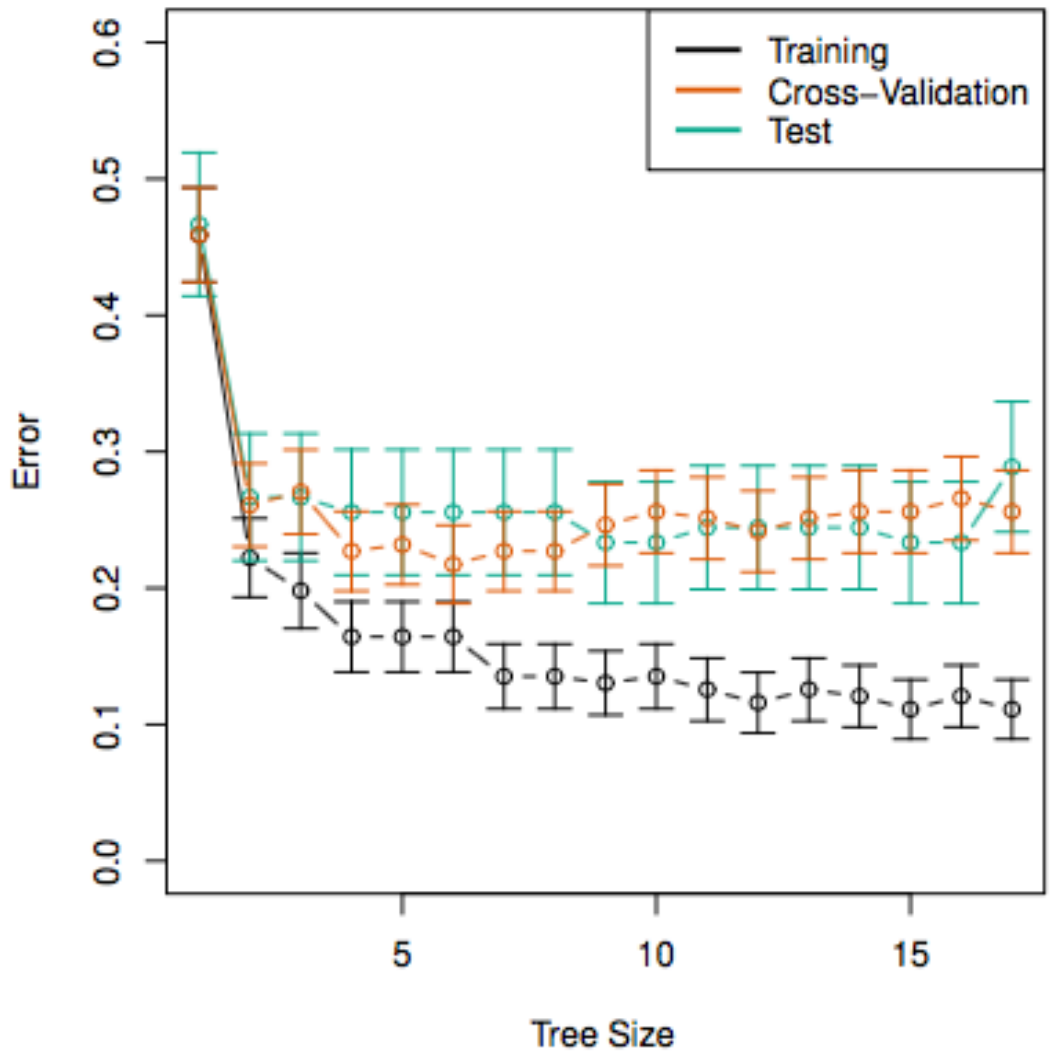


Example 1 (Classification): Predicting Heart Disease

Y: Yes if Heart Disease, No otherwise.

$X =$

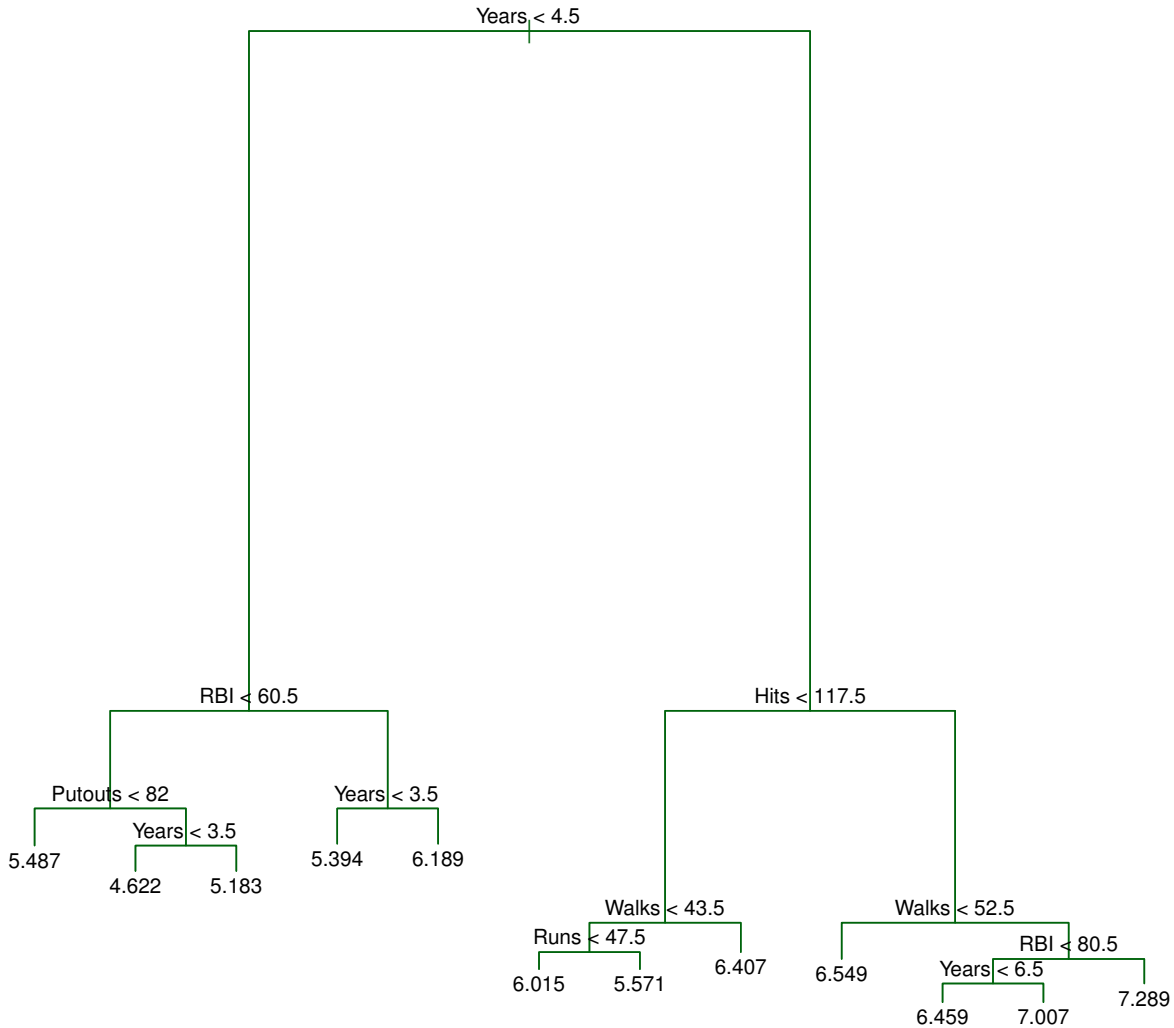
{Thallium stress test, Ca level, Max. Heart rate, Chest Pain, + 9 more}



Example 2 (Regression): Predicting Baseball Player Salary

Y: log-salary ('000 \$)

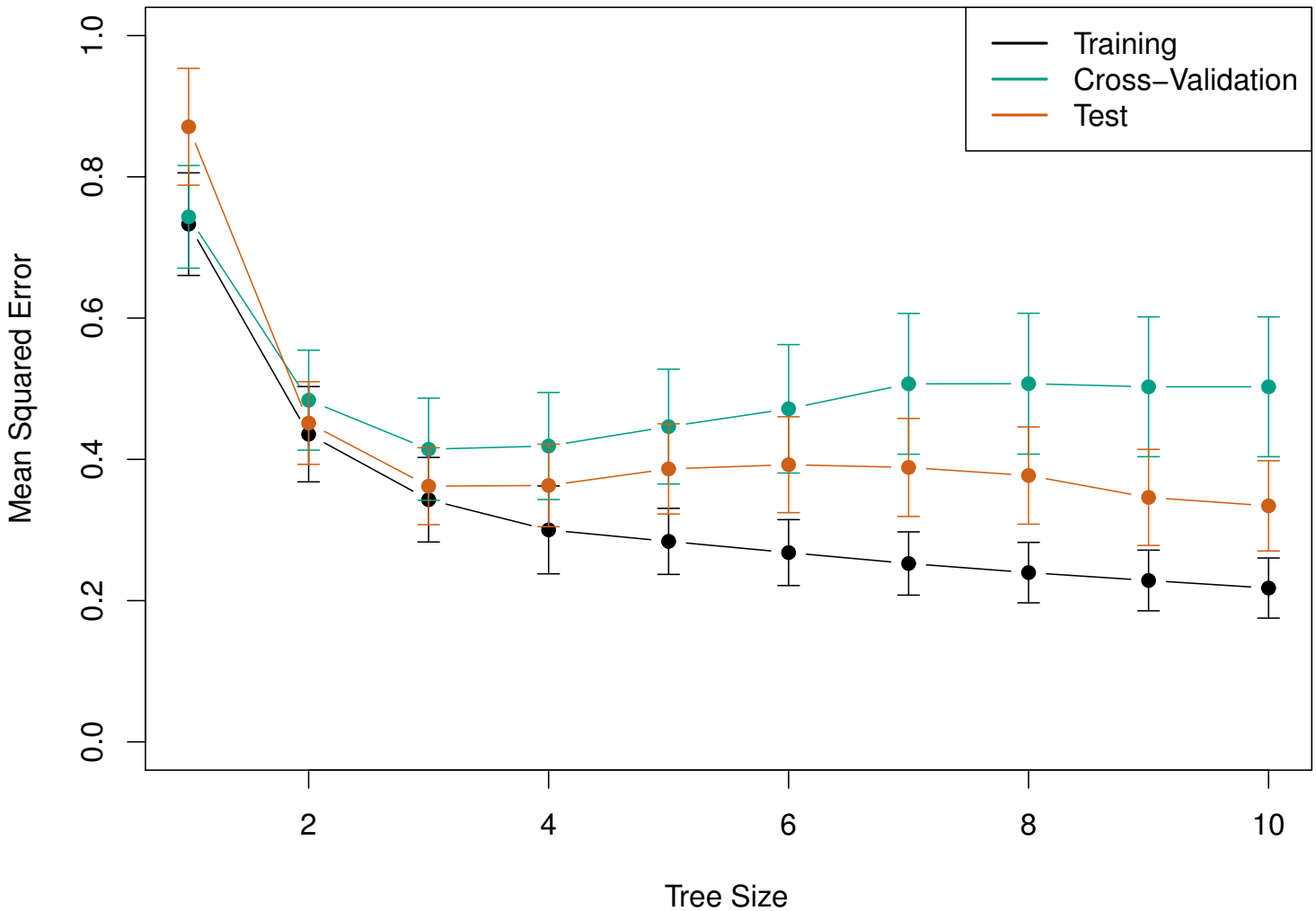
$X = \{$no. of years played, no. of hits in previous year, + 7 more$\}$



Example 2 (Regression): Predicting Baseball Player Salary

Y: log-salary ('000 \$)

$X = \{$no. of years played, no. of hits in previous year, + 7 more$\}$



CART: pros and cons

Pros:

- Quick to compute (details?) $O(pn \log n)$ for split computations, $pn \log n$ for sorting each predictor
- Interpretable - think medical charts!

Cons:

- High variance ($\Longrightarrow$ low predictability): small change in data can result in a very different series of splits, and thus, the tree
- By construction, lacks smoothness

Bagging (Bootstrap Averaging)

- Average over many samples:

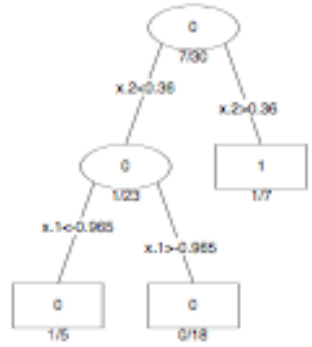
$$\hat{f}_{bag}(\mathbf{X}) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{b}(\mathbf{X}),$$

  where $\hat{f}^{b}(\mathbf{X})$ is the prediction model on the b-th bootstrapped training data set, where a bootstrapped data set is constructed by sampling from the original *with replacement*.
- Can reduce variance dramatically; but comes at the cost of interpretability, because the model can't be represented by the single tree.
- Variable Importance graph: summarize the overall importance of the predictor variables

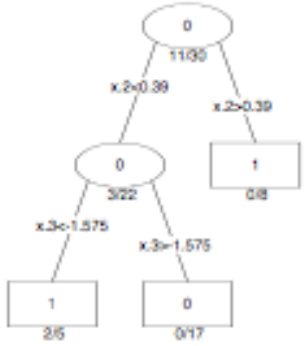
Bagging (Bootstrap Averaging)

Original Tree

Bootstrap Tree 1



Bootstrap Tree 2



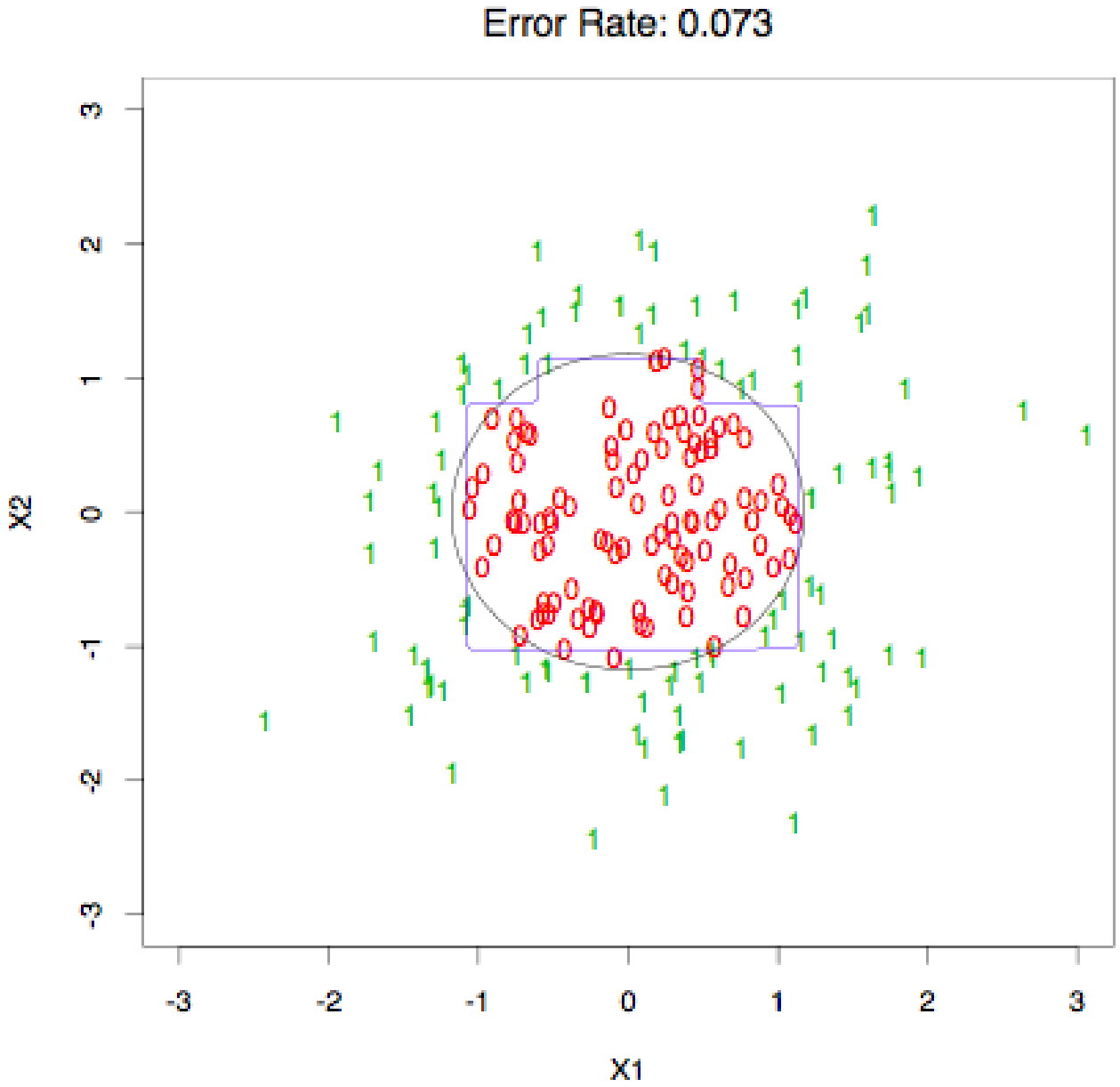
Bootstrap Tree 3

Bootstrap Tree 4

Bootstrap Tree 5

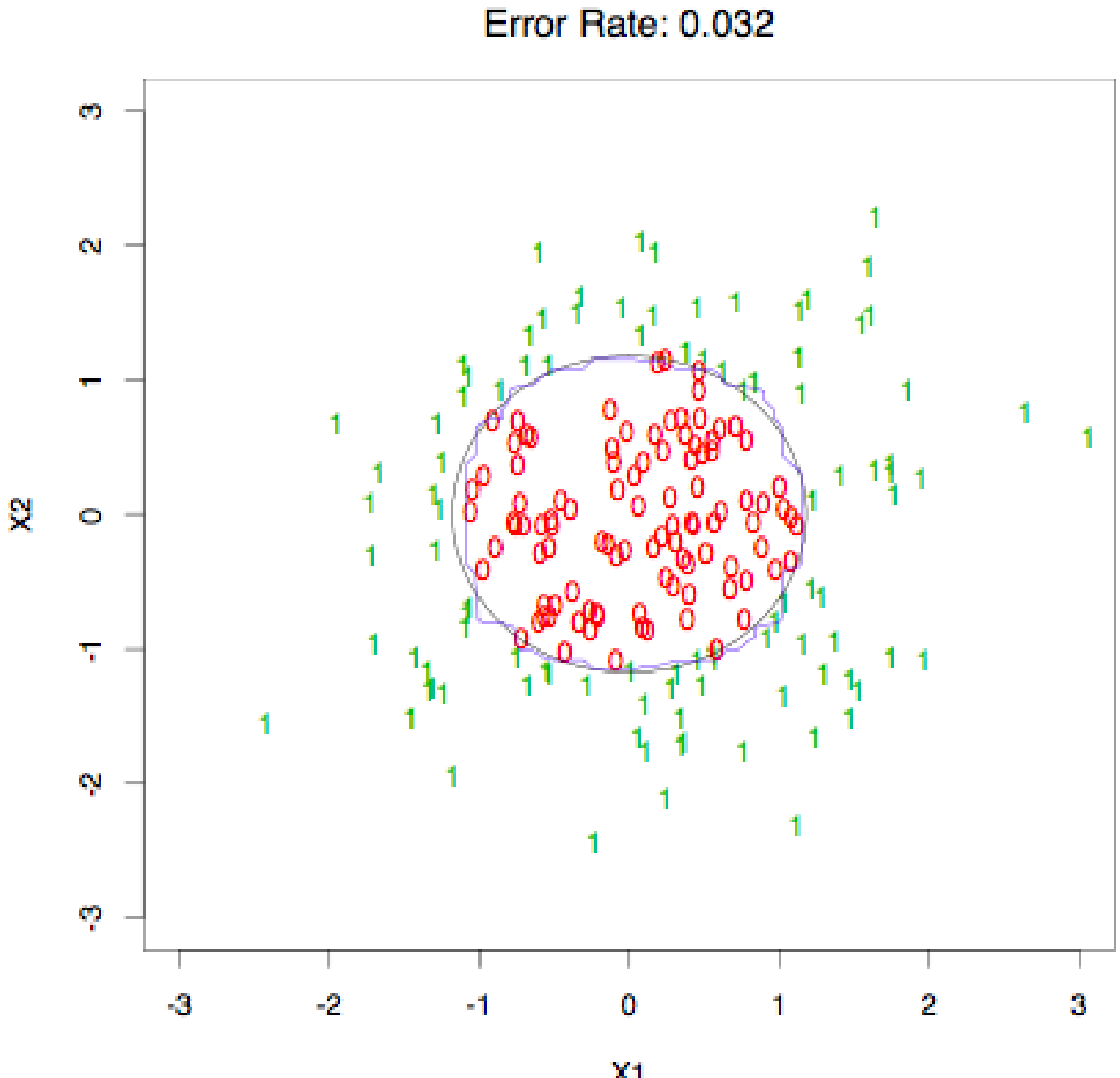
Bagging (Bootstrap Averaging)

Classification boundary without bagging:



Bagging (Bootstrap Averaging)

Classification boundary with bagging:



Random Forest: Cleverer Averaging

- Bagging (bootstrap aggregation): overlap between each bootstrap sample is large, so expect to have high correlations between the bootstrapped trees.
- Random Forest (Breiman, 2001): also average multiple trees, but reduce the correlation between them by sub-sampling covariates each time
- Result: more “independent” average of random trees, hence the name, Random Forest
- One of the most popular “off-the-shelf” methods used

Random Forest

Algorithm [ESII]

1. For $b = 1$ to B:
   - Draw a bootstrap sample Z^* of size N from training data
   - Grow a tree T_b to the bootstrapped data, until minimum node size n_{min} is reached as follows:
     - Select m variables at random from the p variables.
     - Pick the best variable/split-point among the m.
     - Split the node into two daughter nodes.
2. Output: ensemble of trees $\{T_b\}_1^B$. Prediction at a new point $\mathbf{x}$:

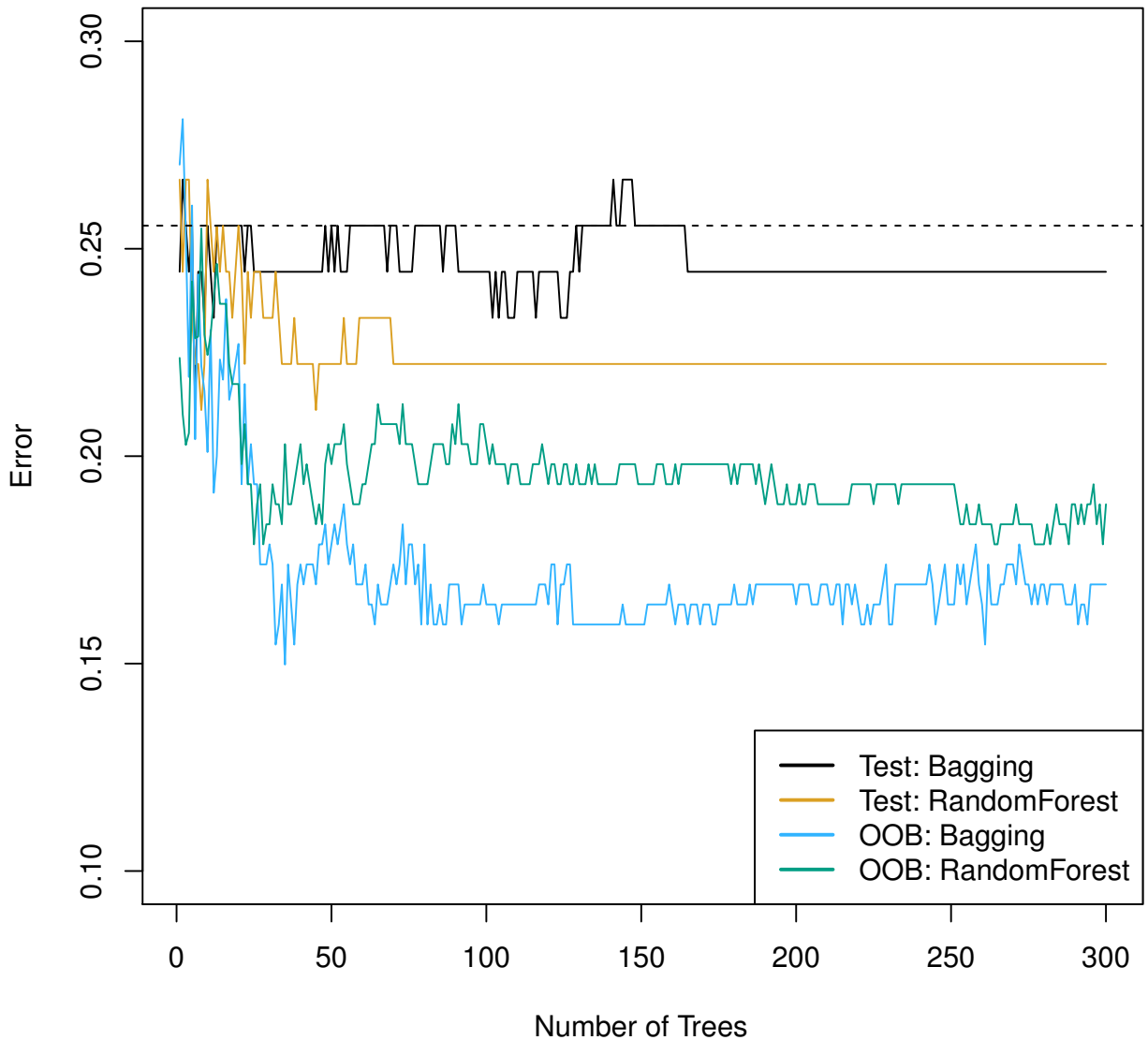
   - Regression:

$$\hat{f}_{rf}^B(\mathbf{x}) = \frac{1}{B}\sum_{b=1}^{B} T_b(\mathbf{x})$$

   - Classification: the majority vote of B trees

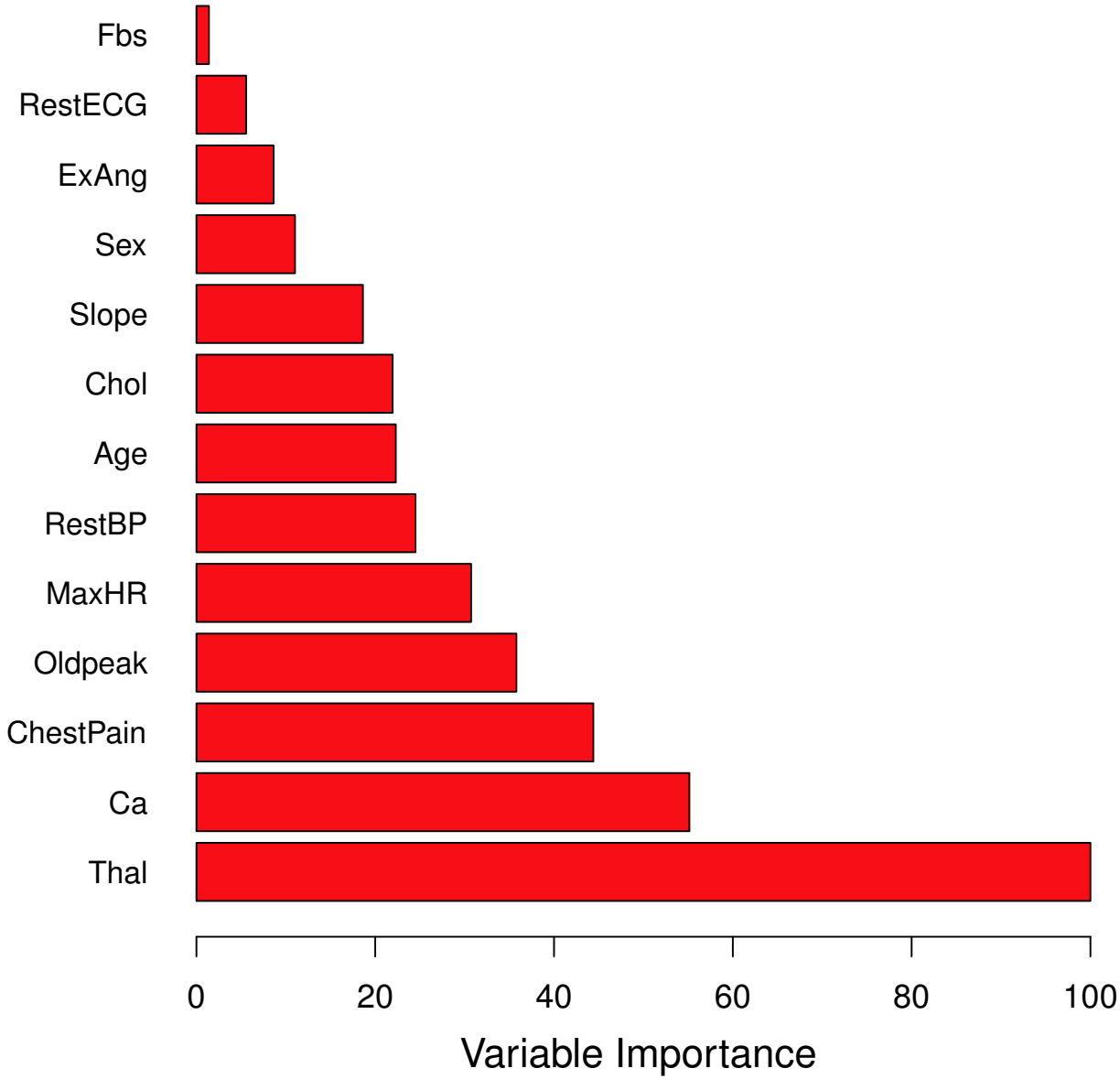
Example 1 (Classification): Predicting Heart Disease

with Bagging and Random Forest



Example 1 (Classification): Predicting Heart Disease

Variable Importance Graph



Boosting Methods

- One of the most powerful learning ideas in last 20 years
- Basic idea: Combine outputs of many "weak predictors"[1] to produce a powerful committee
- Of the many varieties, AdaBoost [Freund and Schapire (1997)] is perhaps best known

[1] A weak predictor is one whose error rate is only slightly better than random guessing

Adaboost

- Start with a weak predictor $G_1(\cdot)$ and a training data set
- Apply the weak predictor to repeatedly modified versions of the data
- Final model: $\sum_{m=1}^{M} \alpha_m G_m(x)$
- $\alpha_1, \ldots, \alpha_m$: weights computed by the algorithm; gives higher weights to more accurate predictors
- Data modification: at each step, re-weight the observations such that those with higher mis-prediction error in the previous step have their weights increased, and those with lower mis-prediction error have their weights decreased
- As algorithm proceeds, observations that are difficult to classify correctly receive ever-increasing influence.

Adaboost

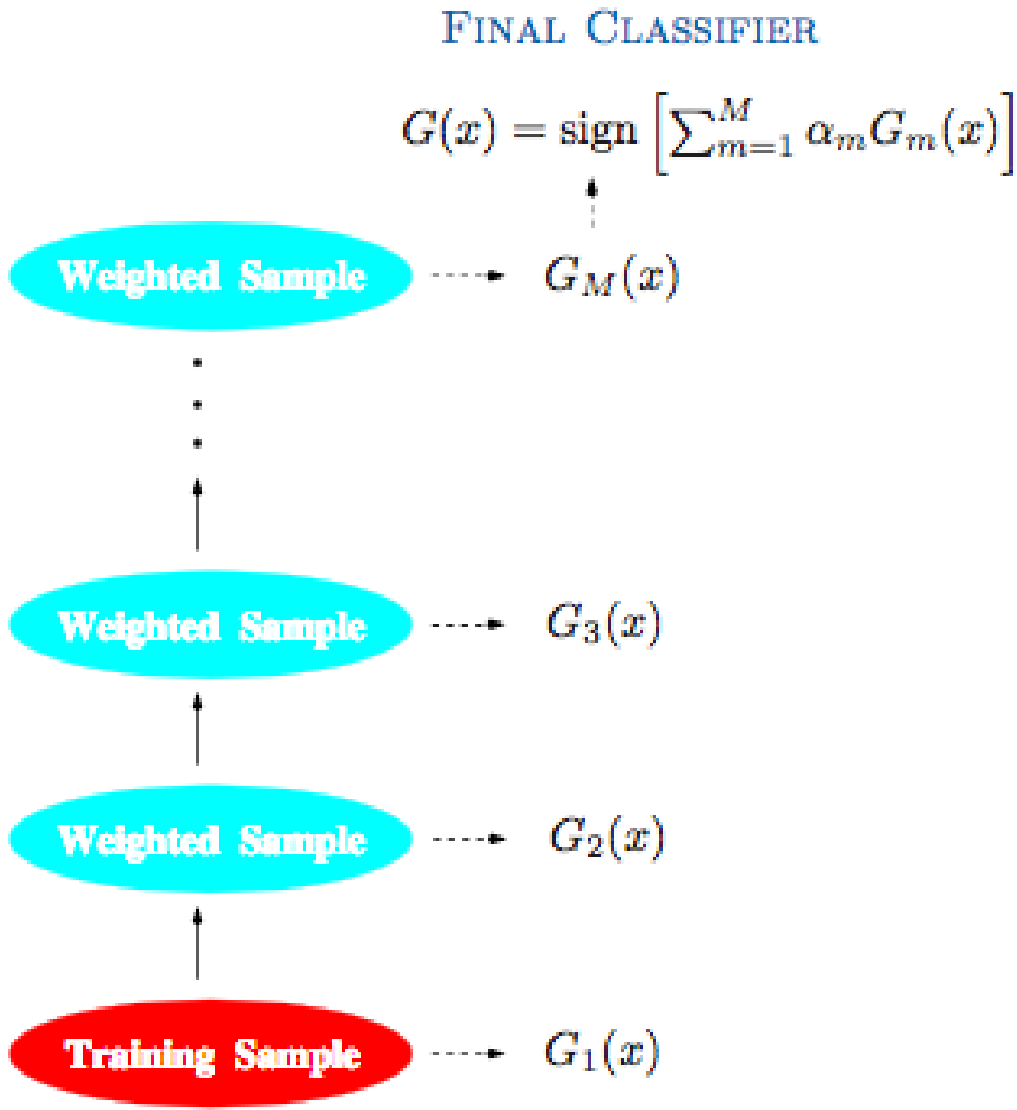
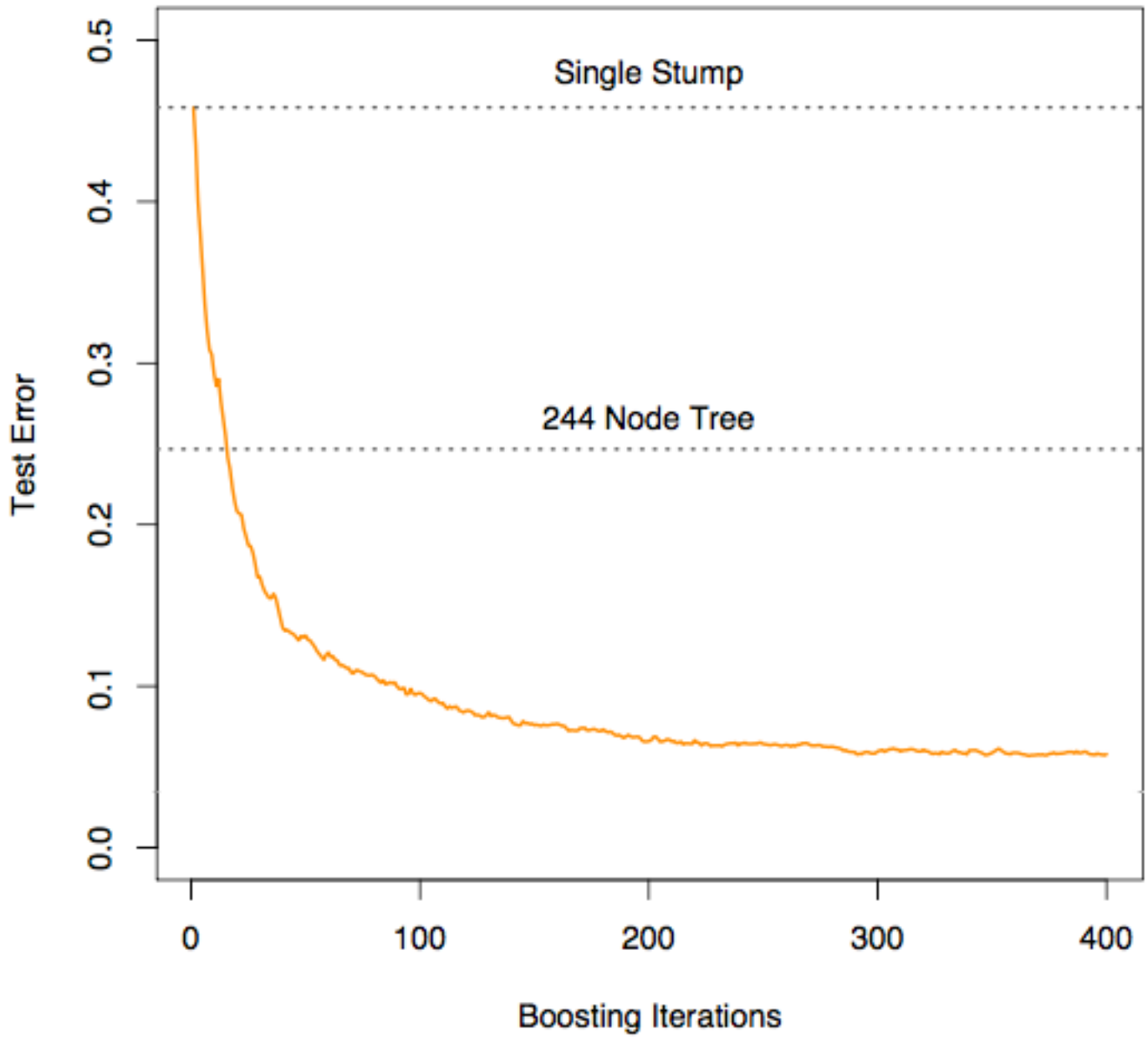


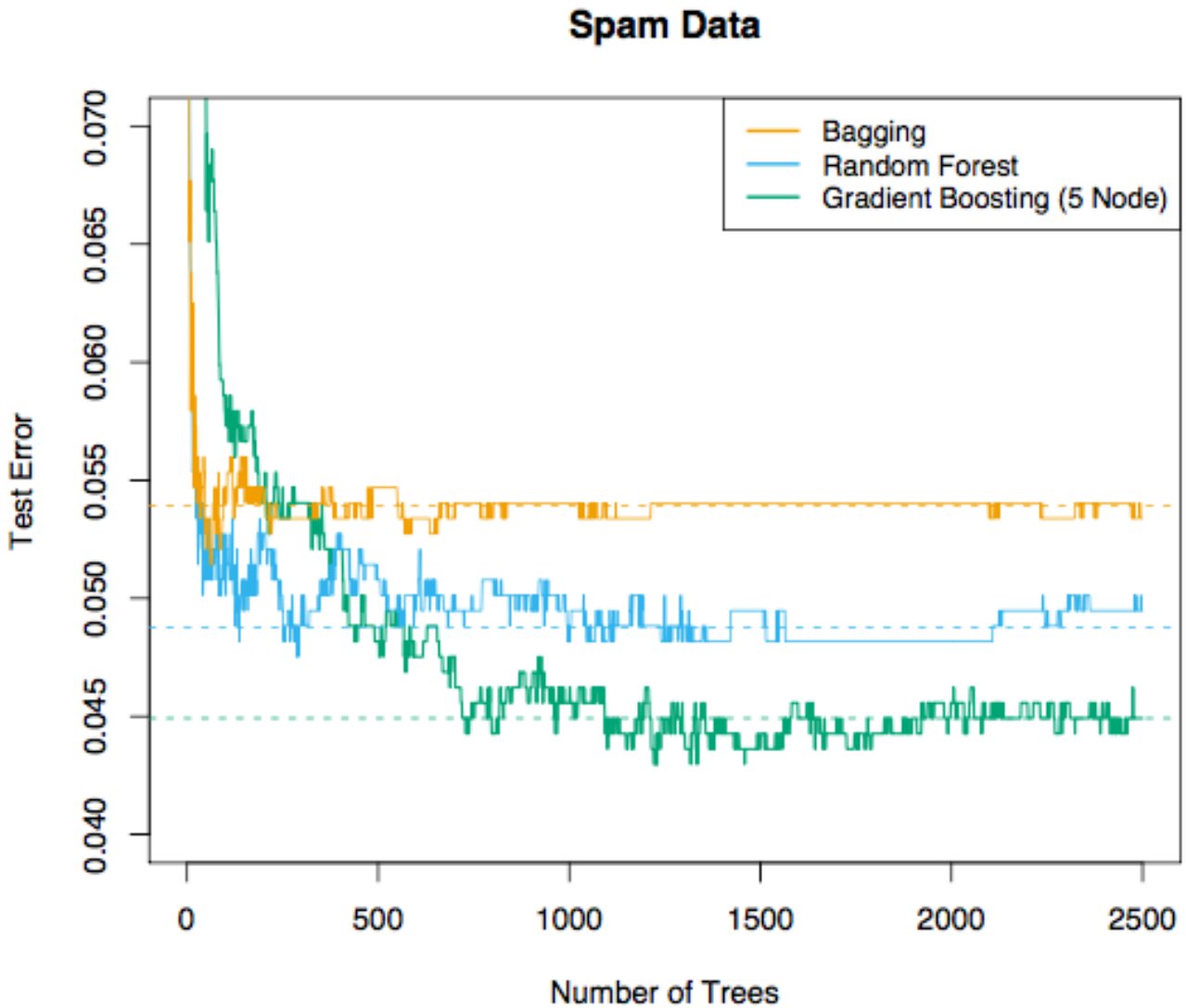
FIGURE 10.1. *Schematic of AdaBoost. Classifiers are trained on weighted versions of the dataset, and then combined to produce a final prediction.*

Adaboost


Single Stump
244 Node Tree
Test Error
0.0
0.1
0.2
0.3
0.4
0.5
0
100
200
300
400
Boosting Iterations

Example 3: Spam Email Classification

Bagging vs Random Forest vs Boosting



Application: Demand Prediction

- FLSL16, MSOM: Rue La La, online fashion sample-sales company
- Limited-time discounts on designer apparel and accessories.
- Challenge: pricing and predicting demand for products never sold before, which account for the majority of sales and revenue

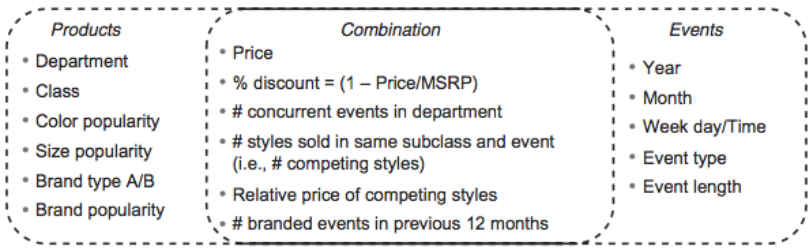
Demand model: stocked-out items

- Lost sales is a big problem — Rue La La operates in an extremely limited inventory environment (average SKU inventory is less than 10 units)
- Solution: use sales data from items that did not sell out to estimate lost sales of items that did sell out
- For each event length (1–4 days) aggregate hourly sales over all items that did not sell out in the event.
- Calculate % of sales that occurs in each hour of the event — the "demand curve." Observation: *demand rate* for each product is primarily a function of customer traffic.
- To estimate the demand for an item that did sell out, identify the time of sell-out and use the appropriate demand curve to estimate the proportion of sales that typically occur within that amount of time.

Data

- Sales transactions, 2011-2013
- Time-stamped sale of an item
- Quantity sold (style, size), price, event start date/time, event length, initial inventory
- Product characteristics (brand, size, color, MSRP (manufacturer's suggested retail price), and hierarchy classification [each item aggregates (across all sizes) to a style, styles aggregate to form subclasses, subclasses aggregate to form classes, and classes aggregate to form departments]
- Price features: actual, MSRP, percent discount off MSRP, relative price of competing styles

Figure 5 Summary of Features Used to Develop Demand Prediction Model



Demand model: first-exposure items

- Input: features, output: demand (actual or estimated)
- Multiple models tested: least squares regression, principal components regression, partial least squares regression, multiplicative (power) regression, semilogarithmic regression, and regression trees with bagging.
- Tuning any parameters: 5-fold CV
- Across all performance metrics evaluated, regression trees with bagging consistently outperformed the other regression models for all departments
- Statistics of Bagged tree (100):
  - mininum no. of observations in each node:10
  - average no. of observations: 21
  - average no. of terminal nodes: 287
- Regression trees are able to define and identify “similar” products sold in the past in order to help estimate future demand.
- Random forests and boosting not tried for “better interpretability”

Price Optimization

A discrete model:

$$\max \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} p_j \mathbb{E}[D_{ijk} | p_j, k] x_{ij}$$

$$s.t. \sum_{j \in \mathcal{M}} x_{ij} = 1 \quad \forall\, i \in \mathcal{N}$$

$$\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} p_j x_{ij} = k$$

$$x_{ij} \in \{0, 1\} \quad \forall\, i \in \mathcal{N},\ j \in \mathcal{M}$$

- $\mathcal{N}$ is the set of styles in a given subclass
- $\mathcal{M}$ is the discrete set of possible prices for each style, e.g. $\{\$24.90, \$29.90, \$34.90\}$, denote by p_j
- x_{ij} binary variable; equals 1 if style i is assigned price p_j, 0 otherwise
- D_{ijk} is the random demand of the i-th style and j-th possible price when the sum of prices of competing styles is k

Results

- Field experiment conducted
- Sales does not decrease for medium and high price point products does not decrease due to recommended price increase
- Increase in revenue of the test group by approximately 9.7% with an associated 90% confidence interval of [2.3%, 17.8%]

Summary

- Tree-based learning algorithms (CART, bagging, random forest, boosting) are among the most popular and successful methods currently used
- However, for any given problem, seldom known in advance which procedure will perform best or even well
- You can evaluate the different methods on multiple dimensions; not just predictability. Interpretability is another important dimension to consider.
- You may also be concerned about: natural handling of data from mixed type; missing values; robustness to outliers in input space; sensitivity to monotone transformations of inputs; computational scalability; automated handling of irrelevant inputs.