STAT 5630, Fall 2019

Trees and Random Forests

Xiwei Tang, Ph.D. <xt4yj@virginia.edu>

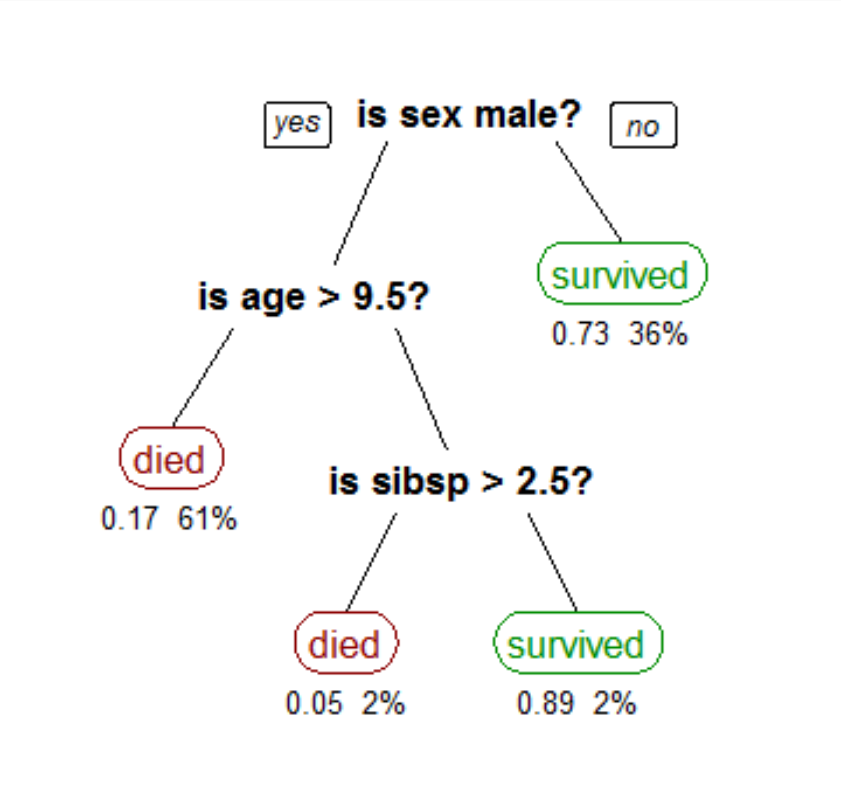
University of Virginia
October 17, 2019

Classification and regression Trees (CART)

Tree-based methods

- Tree-based methods are nonparametric methods that recursively partition the feature space into hyper-rectangular subsets, and make prediction on each subset.
- Two main streams of models:
  - Classification and regression Trees (CART): Breiman, Friedman, Olshen and Stone (1984)
  - ID3/C4.5: Quinlan, 1986, 1993.
- In statistics, we usually focus on CART.

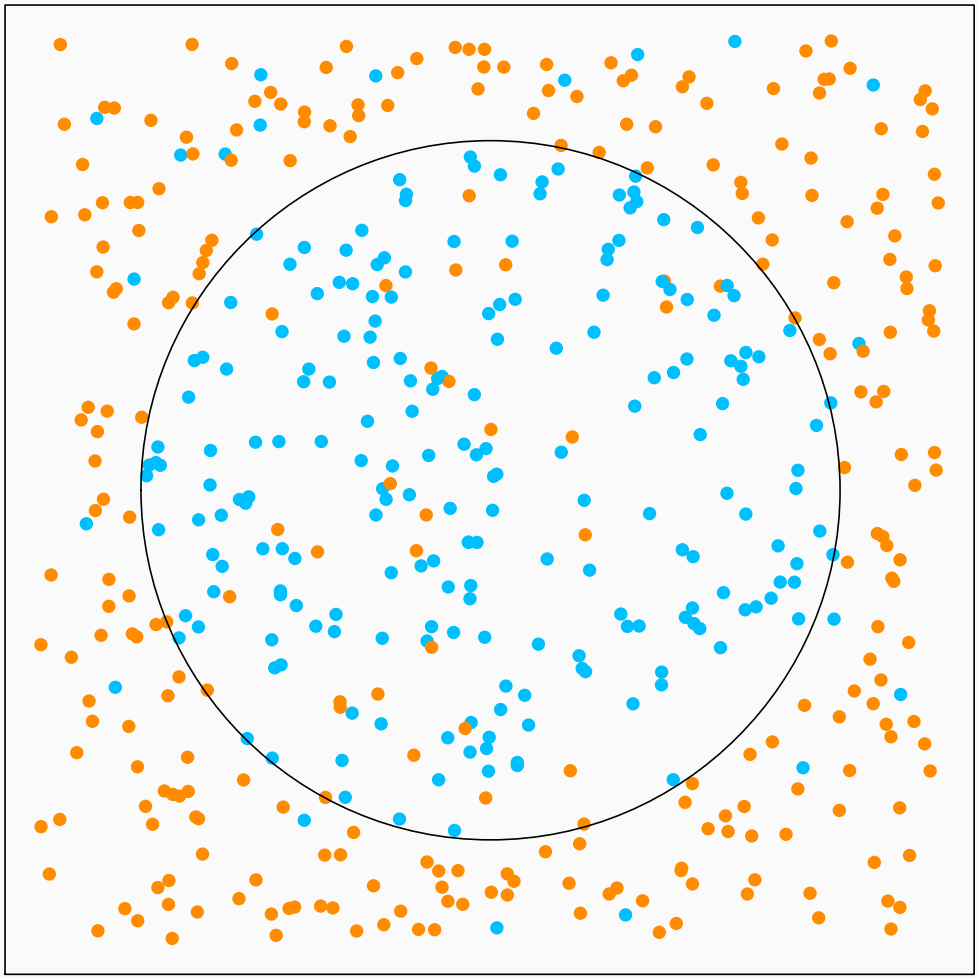
Titanic Survivals



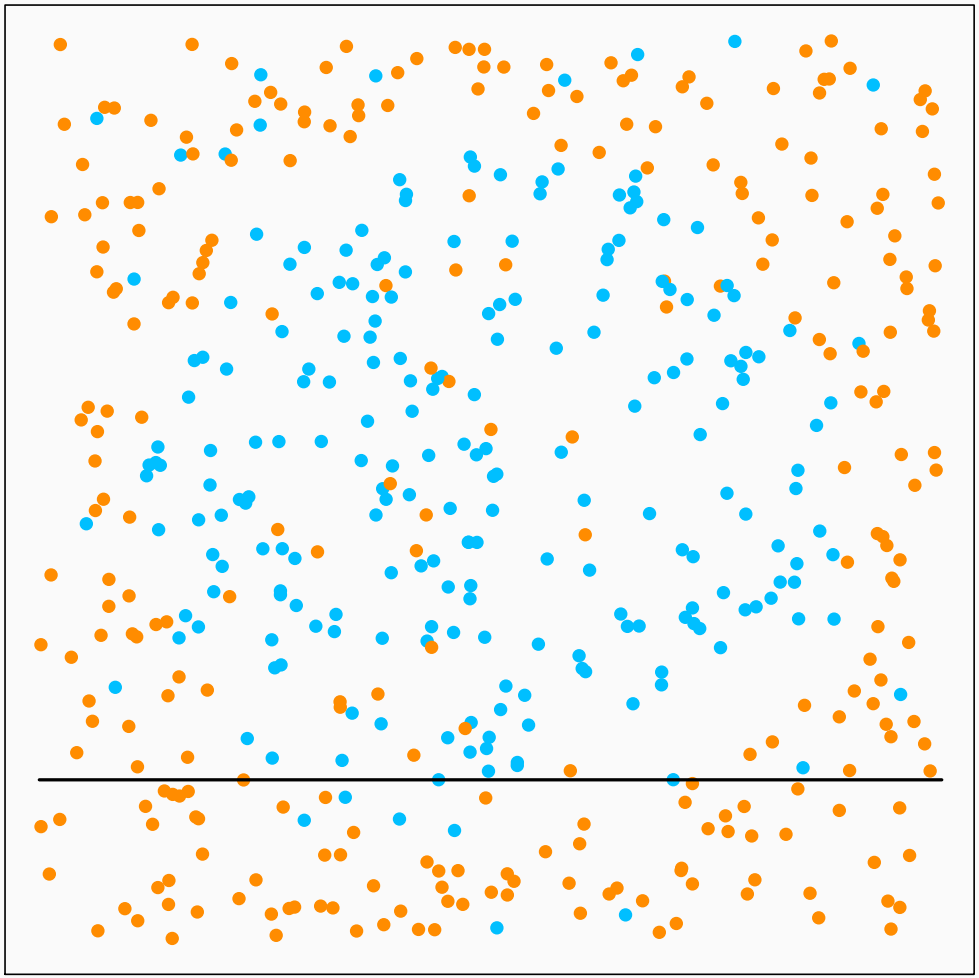
Classification and regression Trees

- How tree works in classification?
- Example: independent x_1 and x_2 from uniform $[-1, 1]$, 90% to observed class 1 (blue) within the circle $x_1^2 + x_2^2 < 0.6$, and 90% to observed class 2 (orange) outside the circle.
- All existing classification methods that we have introduced require either transformation of the space (SVM) or distance measure (kNN, kernel)
- Tree solves this by directly cutting the feature space using a binary splitting rule in the form of $\mathbf{1}\{x \leq c\}$

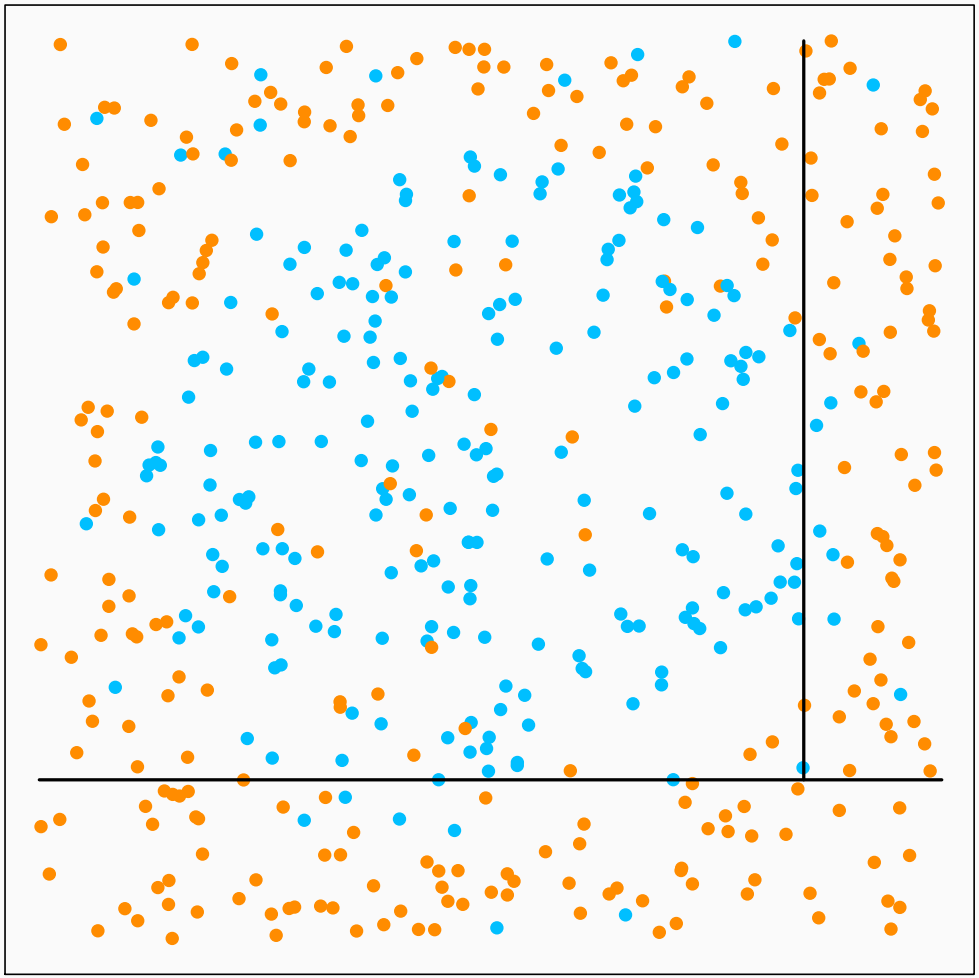
Example



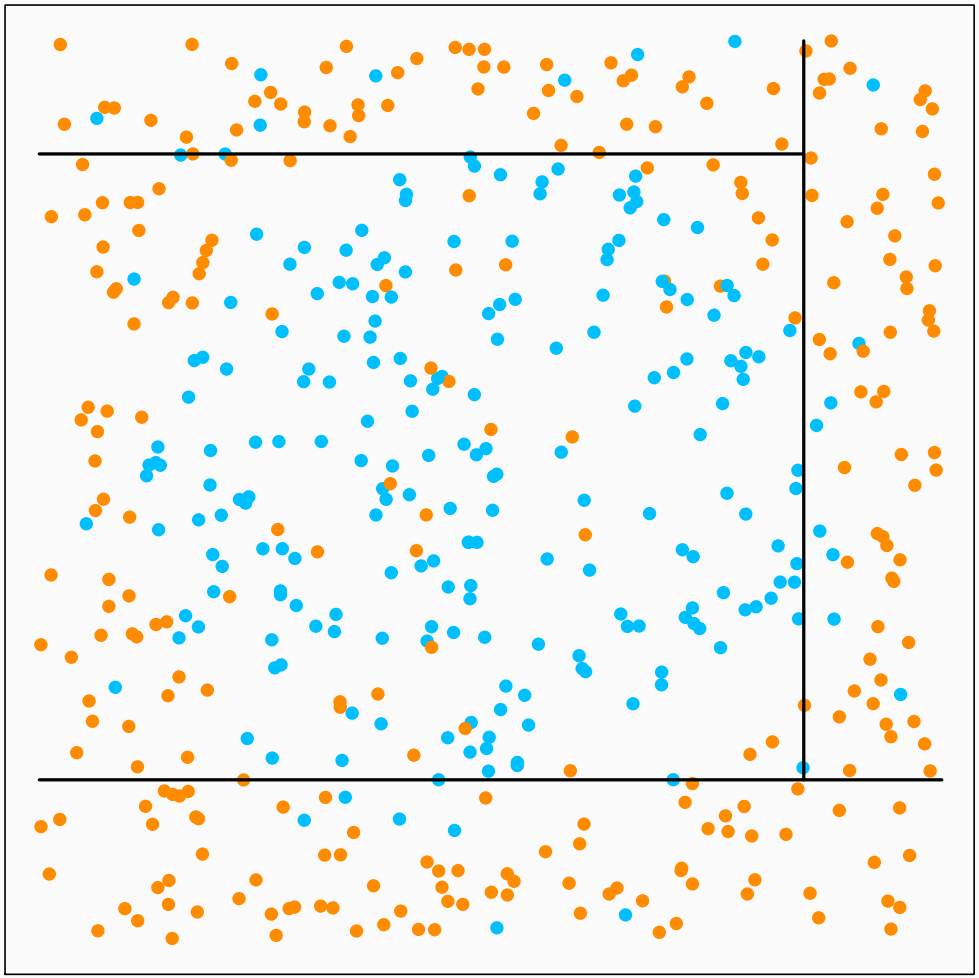
Example



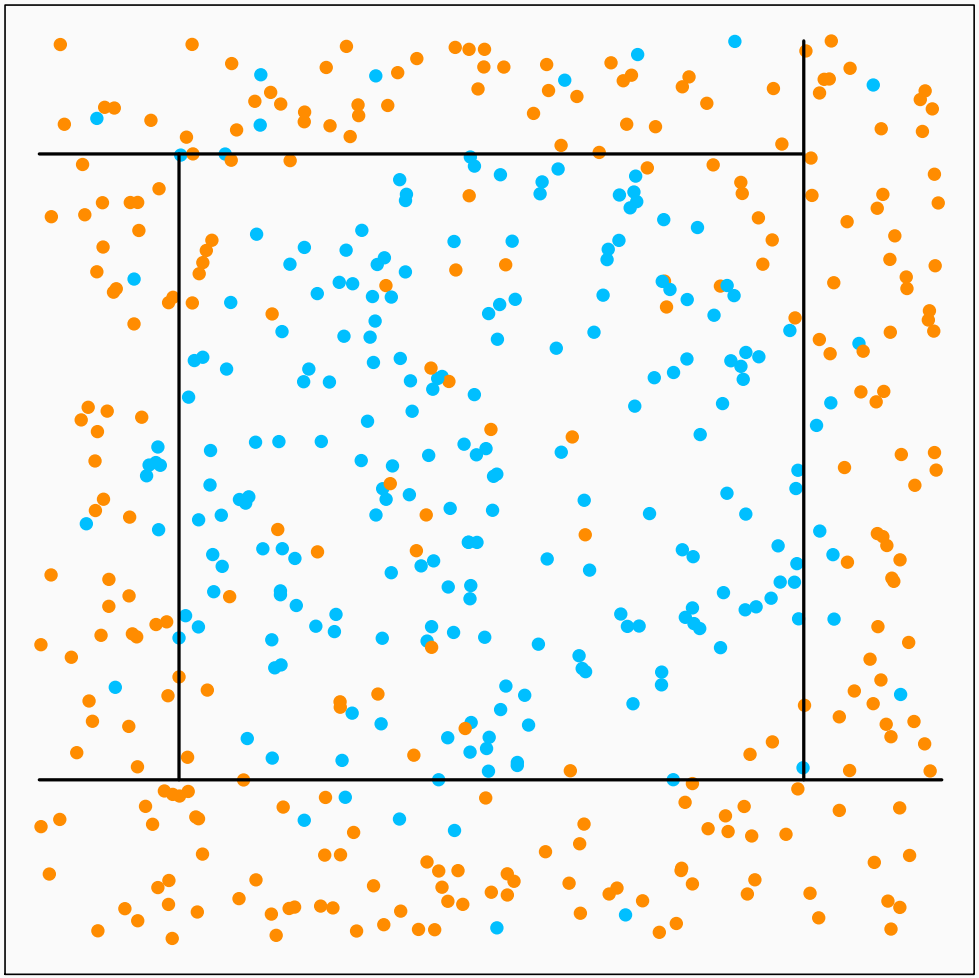
Example



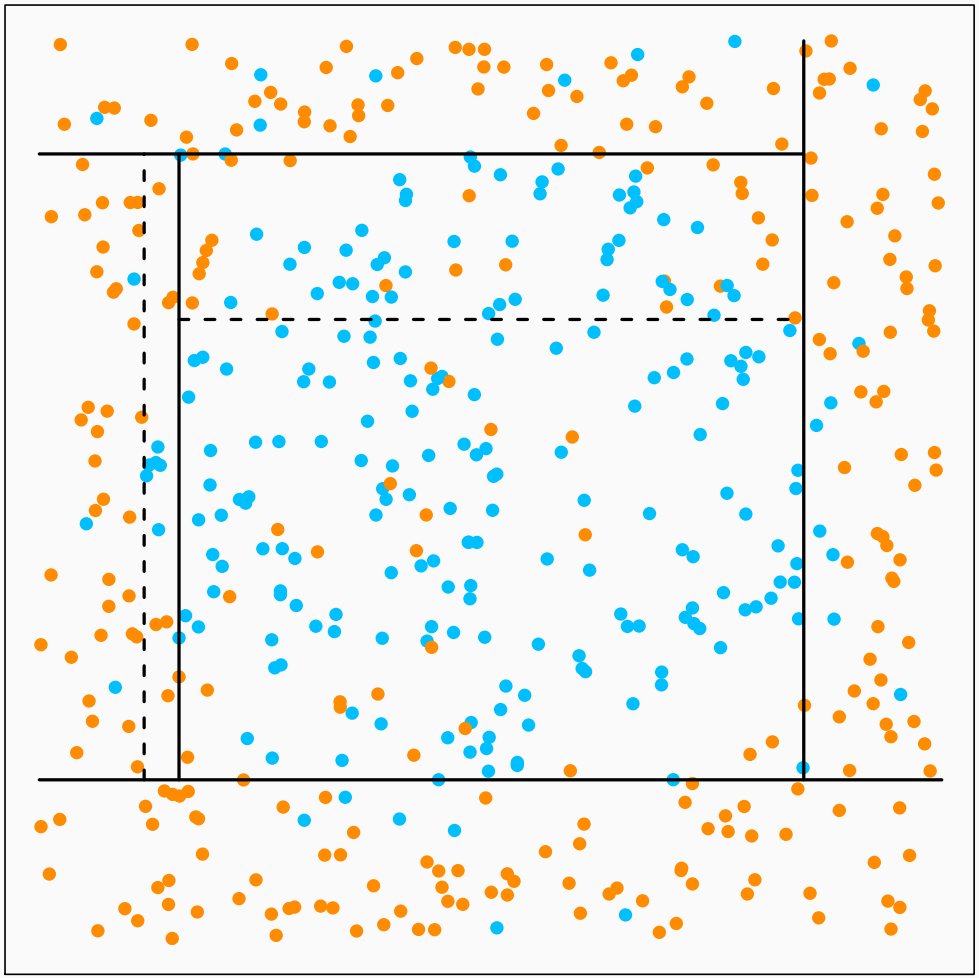
Example



Example



Example

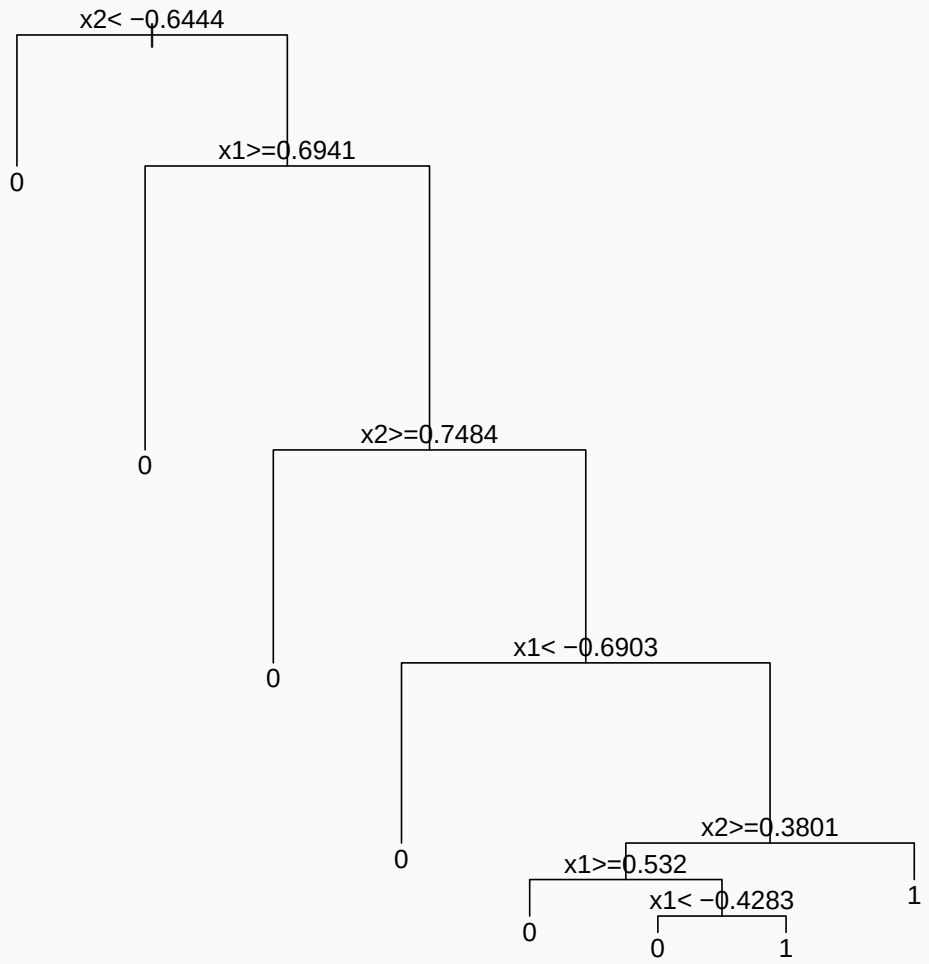


Example

- rpart is one of the popular packages that provide CART fitting and plot
- Other choices include tree, party.
- Read the reference manual carefully!!!

```
> library(rpart)
> x1 = runif(500, -1, 1)
> x2 = runif(500, -1, 1)
> y = rbinom(500, size = 1, prob = ifelse(x1^2 + x2^2 < 0.6,
    0.9, 0.1))
> cart.fit = rpart(as.factor(y)~x1+x2, data = data.frame(x1, x2,
    y))
> plot(cart.fit)
> text(cart.fit)
```

Example

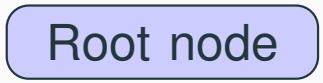

x2< −0.6444
0
x1>=0.6941
0
x2>=0.7484
0
x1< −0.6903
0
x2>=0.3801
x1>=0.532
0
x1< −0.4283
0
1
1

Why tree-based methods?

- Tree-based methods: non-parametric, flexible model structure
- Can handel high-dimensional data without modifying the algorithm (CART may not work well in this setting)
- Single tree model — simple decision rules, interpretable
- Ensemble tree model — high prediction accuracy

Single tree methods: recursive partitioning

- How tree-based methods work?
  - Initialized the root node: all training data

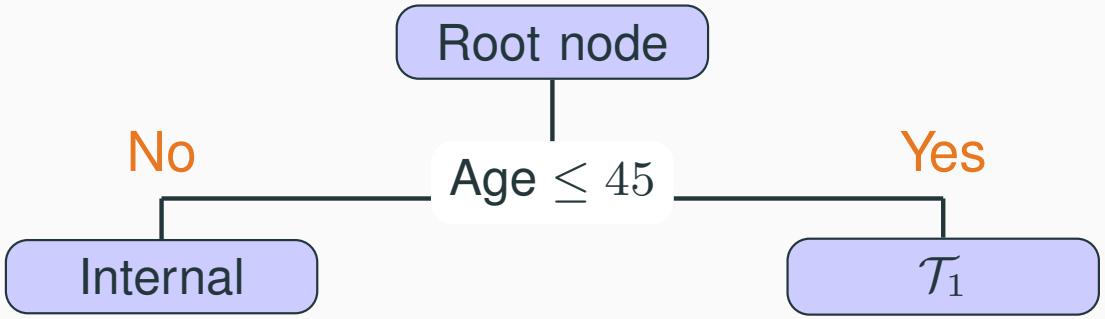


Single tree methods: recursive partitioning

- How tree-based methods work?
  - Initialized the root node: all training data
  - Find a splitting rule $\mathbf{1}\{X^{(j)} \leq c\}$ and split the node

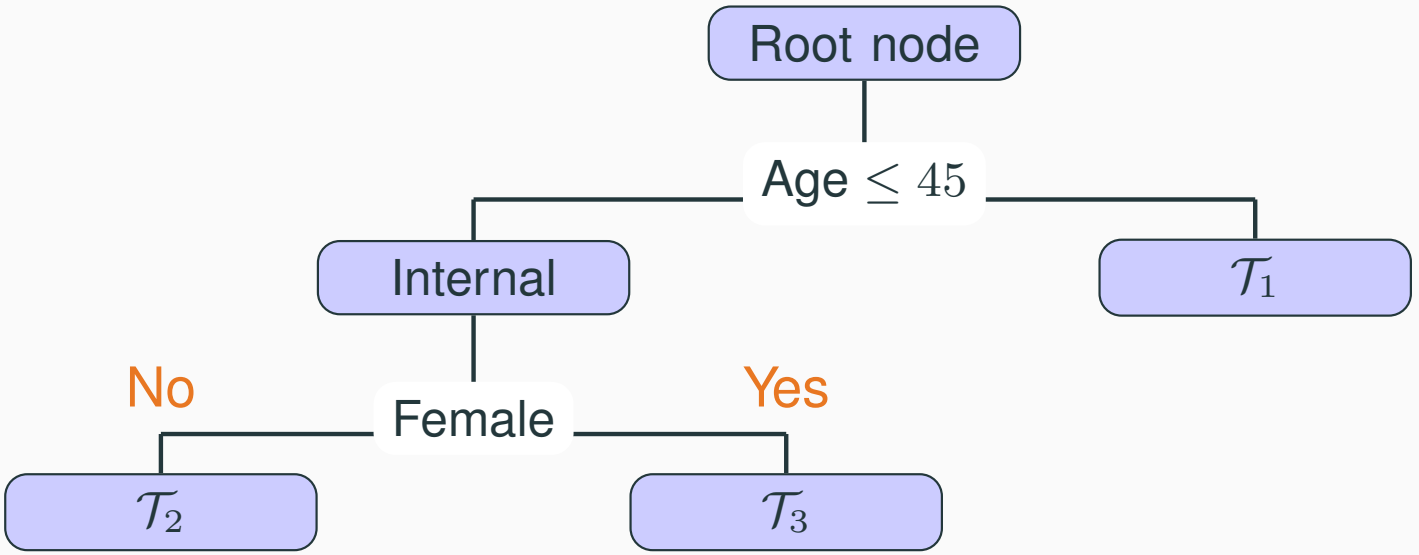
Single tree methods: recursive partitioning

- How tree-based methods work?
  - Initialized the root node: all training data
  - Find a splitting rule $\mathbf{1}\{X^{(j)} \leq c\}$ and split the node
  - Recursively apply the procedure on each daughter node



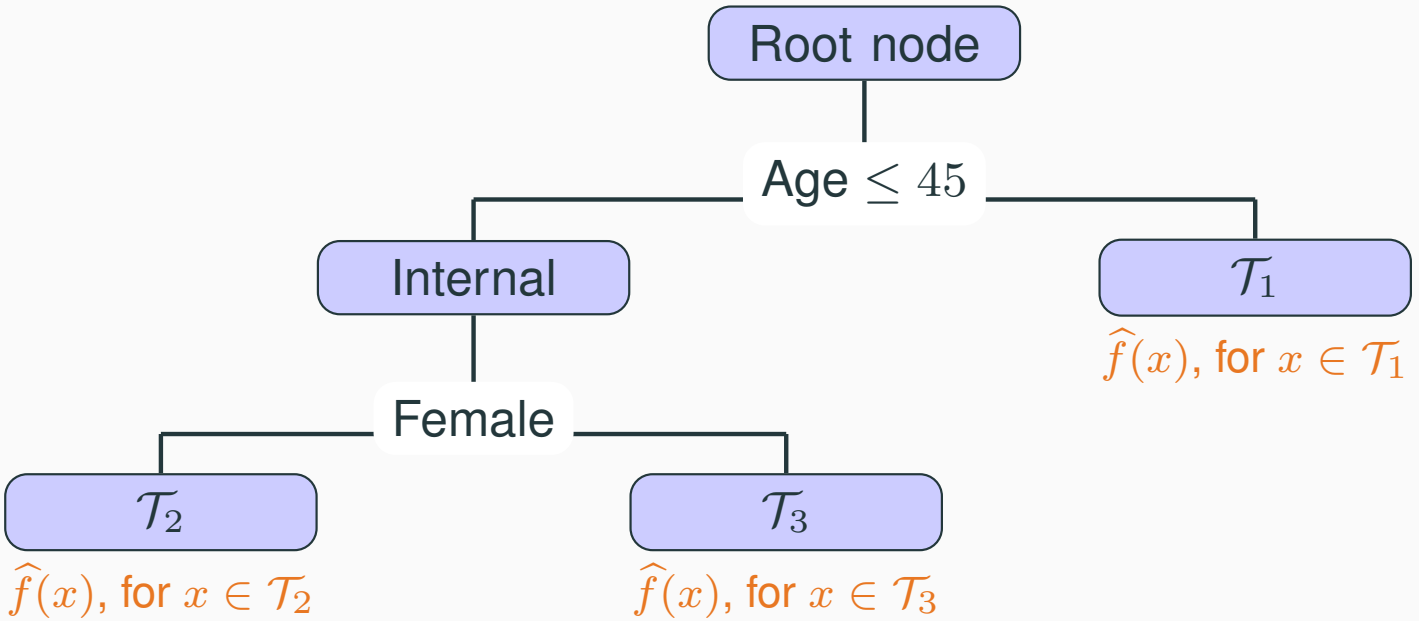
Single tree methods: recursive partitioning

- How tree-based methods work?
  - Initialized the root node: all training data
  - Find a splitting rule $\mathbf{1}\{X^{(j)} \leq c\}$ and split the node
  - Recursively apply the procedure on each daughter node



Single tree methods: recursive partitioning

- How tree-based methods work?
  - Initialized the root node: all training data
  - Find a splitting rule $\mathbf{1}\{X^{(j)} \leq c\}$ and split the node
  - Recursively apply the procedure on each daughter node
  - Predict each terminal node using within-node data



Classification and regression Trees

- How to construct the splitting rules?
- For classification problems
  - continuous predictors
  - categorical predictors
- For regression problems
- Tree pruning

Splitting rules for continuous predictors

- Splitting of continuous predictors are in the form of $\mathbf{1}\{X^{(j)} \leq c\}$
- Consider a node $\mathcal{T}$, we want to split this node into two child nodes.
- We first evaluate the overall impurity of $\mathcal{T}$ by the Gini index (CART), assuming that there are K different classes of the outcome, $1, \ldots, K$,

$$\text{Gini}(\mathcal{T}) = \sum_{k=1}^{K} \widehat{p}_k(1 - \widehat{p}_k) = 1 - \sum_{k=1}^{K} \widehat{p}_k^2$$

where $\widehat{p}_k = \frac{\sum_i \mathbf{1}\{y_i = k\}\mathbf{1}\{x_i \in \mathcal{T}\}}{\sum_i \mathbf{1}\{x_i \in \mathcal{T}\}}$ is the within node frequency of class k.

Impurity Measures

- Now propose a split: $\mathbf{1}\{X^{(j)} \leq c\}$.
- This separates all subjects in the node into two disjoint parts: $\mathcal{T}_L$ and $\mathcal{T}_R$
- For each child node, we can again evaluate Impurity$(\mathcal{T}_L)$ and Impurity$(\mathcal{T}_R)$
- The reduction of impurity is measured by

$$\text{score} = \text{Gini}(\mathcal{T}) - \left(\frac{N_{\mathcal{T}_L}}{N_{\mathcal{T}}} \text{Gini}(\mathcal{T}_L) + \frac{N_{\mathcal{T}_R}}{N_{\mathcal{T}}} \text{Gini}(\mathcal{T}_R) \right),$$

  where $N_{\mathcal{T}_L}$, $N_{\mathcal{T}_R}$ and $N_{\mathcal{T}}$ are the sample size for the corresponding node.
- Go through all variables j and all cutting points c to find the split with the best score
- Using `rpart` or `tree`, the magnitude of this score is reflected by the hight of each split (plot in page 13)

Other Measures

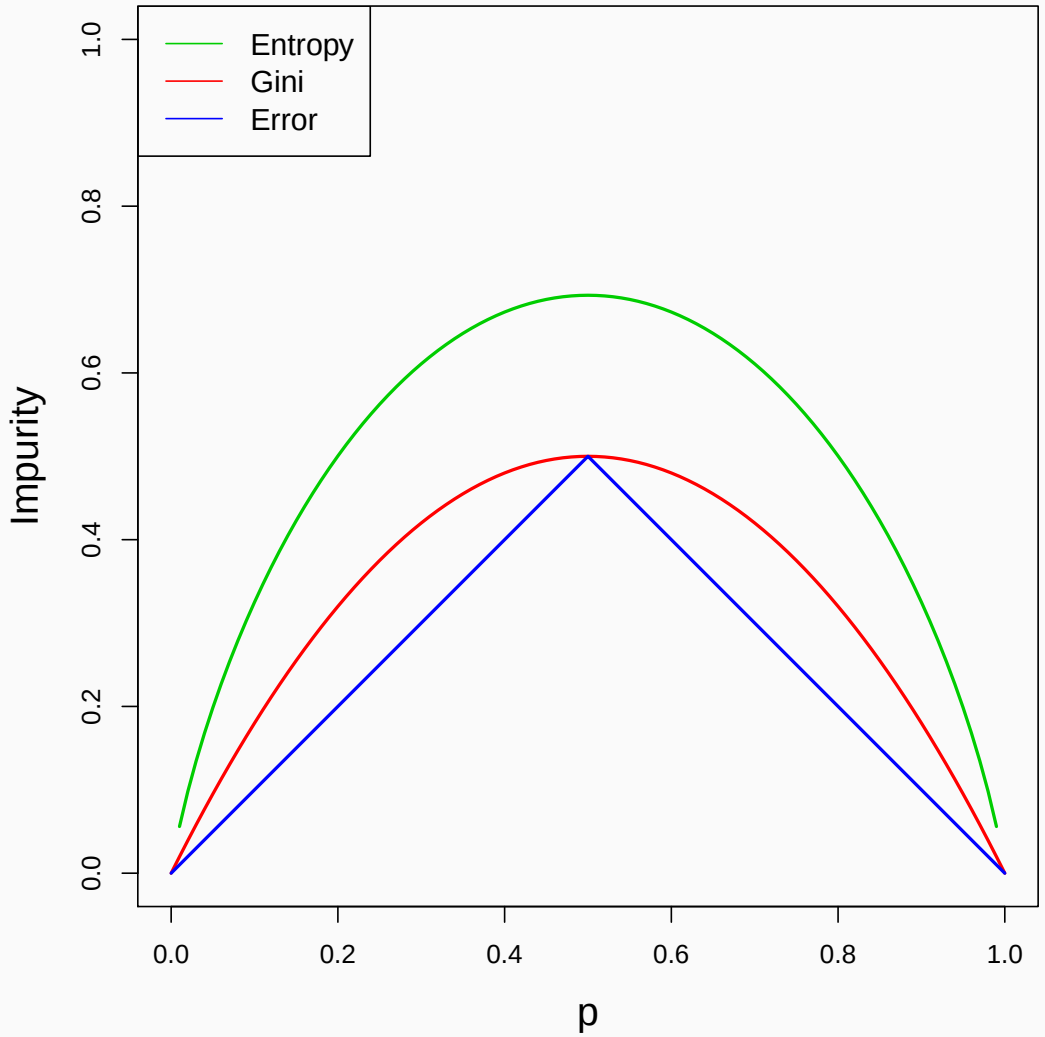
- Gini index is not the only measurement
- Shannon entropy

$$\text{Entropy}(\mathcal{T}) = -\sum_{k=1}^{K} \widehat{p}_k \log(\widehat{p}_k)$$

- Similarly, we can use Shannon entropy to define the reduction of impurity and search for the best splitting rule
- Misclassification error

$$\text{Error}(\mathcal{T}) = 1 - \max_{k=1,\ldots,K} \widehat{p}_k$$

Comparing Measures



Comparing Measures

- Gini index and Shannon Entropy are more sensitive to the changes in the node probability
- They prefer to create more “pure” nodes
- Misclassification error can be used for evaluating a tree, but may not be sensitive enough for building the tree.

Categorical Predictors

- For categorical predictor $X^{(j)}$ talking values in $\{1, \ldots, M\}$, we search for a subset A of $\{1, \ldots, M\}$, and evaluate the child nodes created by the splitting rule

$$\mathbf{1}\{X^{(j)} \in A\}$$

- Maximum of $2^{M-1} - 1$ number of possible splits
- When M is too large, this can be computationally intense.
- Some heuristic methods are used, such as randomly sample a subset of categories to one child node, and compare several random splits.

Tree for Regressions

- When the outcomes y_i's are continuous, all we need is a corresponding impurity measure and score
- Consider the weighted variance reduction:

$$\text{score}_{\text{Var}} = \text{Var}(\mathcal{T}) - \left(\frac{N_{\mathcal{T}_L}}{N_{\mathcal{T}}} \text{Var}(\mathcal{T}_L) + \frac{N_{\mathcal{T}_R}}{N_{\mathcal{T}}} \text{Var}(\mathcal{T}_R) \right)$$

where for any node $\mathcal{T}$, $\text{Var}(\mathcal{T})$ is just the variance of the node samples:

$$\text{Var}(\mathcal{T}) = \frac{1}{N_{\mathcal{T}}} \sum_{i \in \mathcal{T}} (y_i - \overline{y}_{\mathcal{T}})^2$$

Overfitting and Tree Pruning

- A large tree (with many splits) can easily overfit the data
- Small tree may not capture important structures
- Tree size is measured by the number of splits
- Balancing tree size and accuracy is the same as the “loss + penalty” framework
- One possible approach is to split tree nodes only if the decrease in the loss exceed certain threshold, however this can be short-sighted
- A better approach is to grow a large tree, then prune it

Cost-Complexity Pruning

- Fit the entire tree T_{max} (possibly one observation per terminal node). Specify a complexity parameter α.
- For any sub-tree of T_{max}, denoted as $T \preceq T_{max}$, calculate

$$C_\alpha(T) = \sum_{\text{all terminal nodes } t \text{ in } T} N_t \cdot \text{Impurity}(t) + \alpha|T|$$
$$= C(T) + \alpha|T|$$

  where N_t is the number of observations in t, $|T|$ is the size of T, i.e., the number of terminal nodes
- Find T that minimize the $C_\alpha(T)$
- Large α results in small trees
- Choose α using CV

Remark

- Advantages of tree-based method:
  - handles both categorical and continuous variables in a simple and natural way
  - Invariant under all monotone transformations of variables
  - Robust to outliers
  - Flexible model structure, capture iterations, easy to interpret
- Limitations
  - Small changes in the data can result in a very different series of splits
  - Non-smooth. Some other techniques such as the multivariate adaptive regression splines (MARS, Friedman 1991) can be used to generate smoothed models.

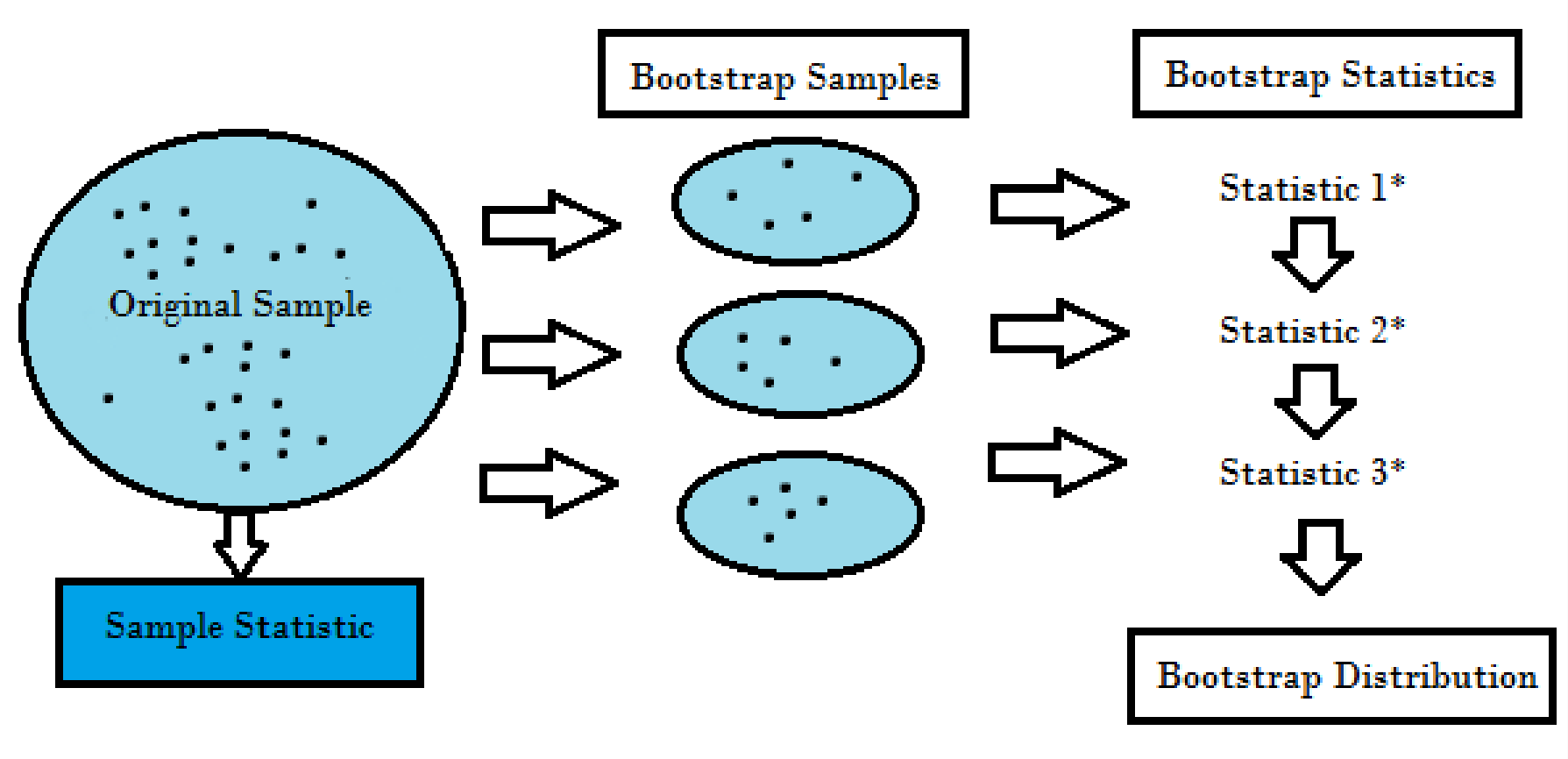
Random Forests

Weak and Strong Learners

- Whether aggregated “weak learners” (unstable, less accurate) can be a “strong learner”.
- Bagging, boosting, and random forests are all along this line.
- Bagging and random forests learn individual trees with some random perturbations, and “average” them.
- Boosting progressively learn models with small magnitude, then “add” them.
- In general, Boosting, Random Forests $\succ$ Bagging $\succ$ Single Tree.

Bootstrapping

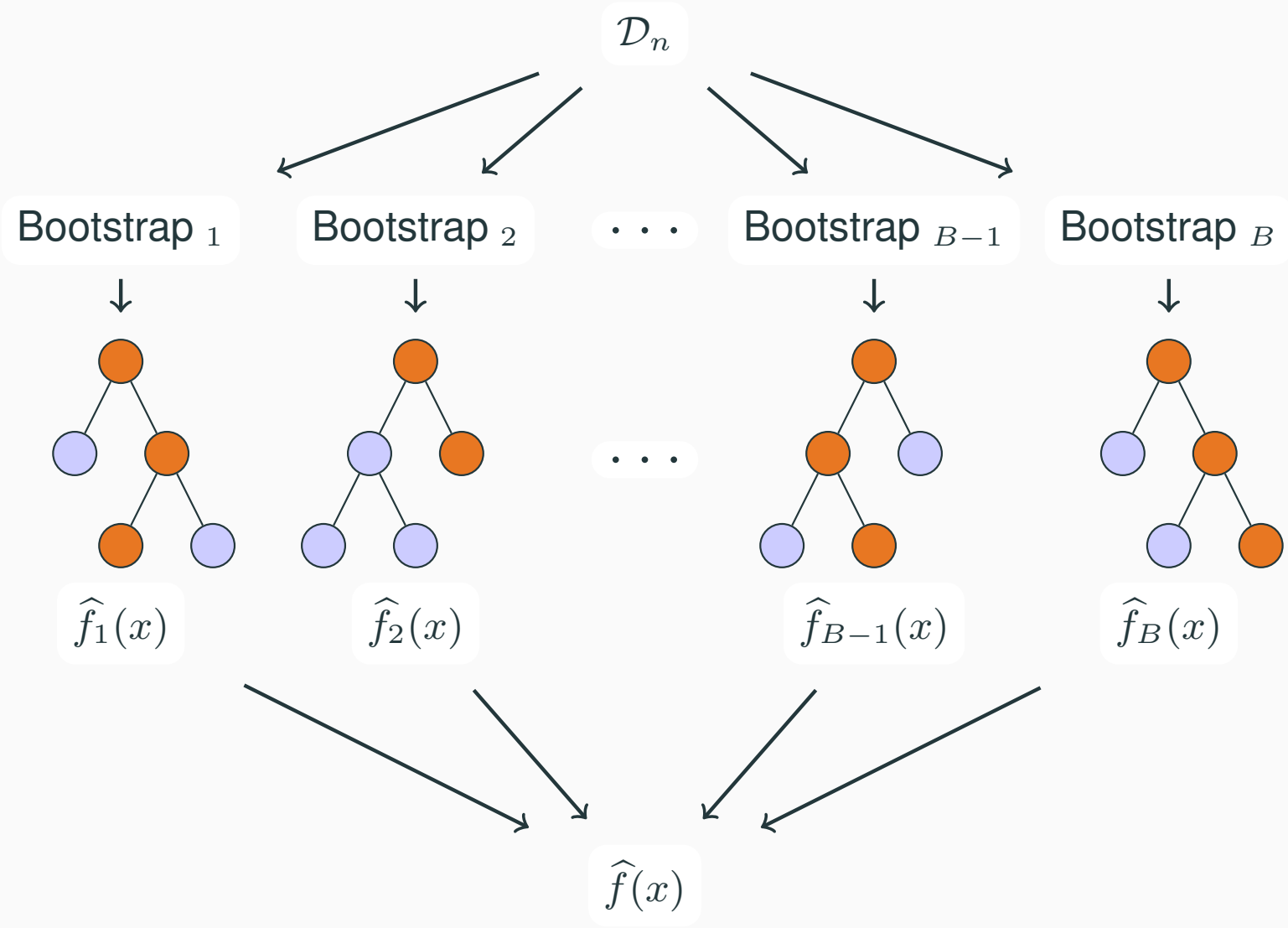
- Random Sampling with Replacement



Bagging Predictors

- Bagging stands for "Bootstrap aggregating"
- Draw M bootstrap samples from the training dataset, fit CART to each, then average
- Motivation: CART is unstable as we discussed earlier, however, perturbing and averaging can improve stability and leads to better accuracy

Ensemble of trees



Bagging Predictors

- Bootstrap sample with replacement: some observations can be repeated multiple times. $\sim 63.2\%$ unique samples
- Fit a CART model to each bootstrap sample (may require tuning using CV).
- To combine each learner, for classification problems:

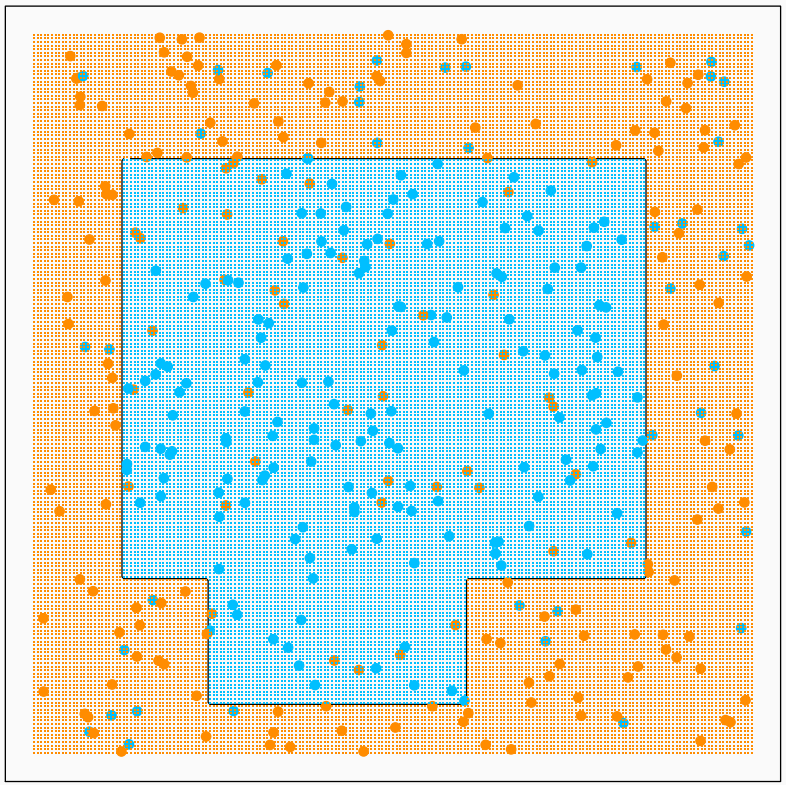
$$\widehat{f}_{\text{bag}}(x) = \text{Majority Vote}\left\{\widehat{f}_b(x)\right\}_{b=1}^{B},$$

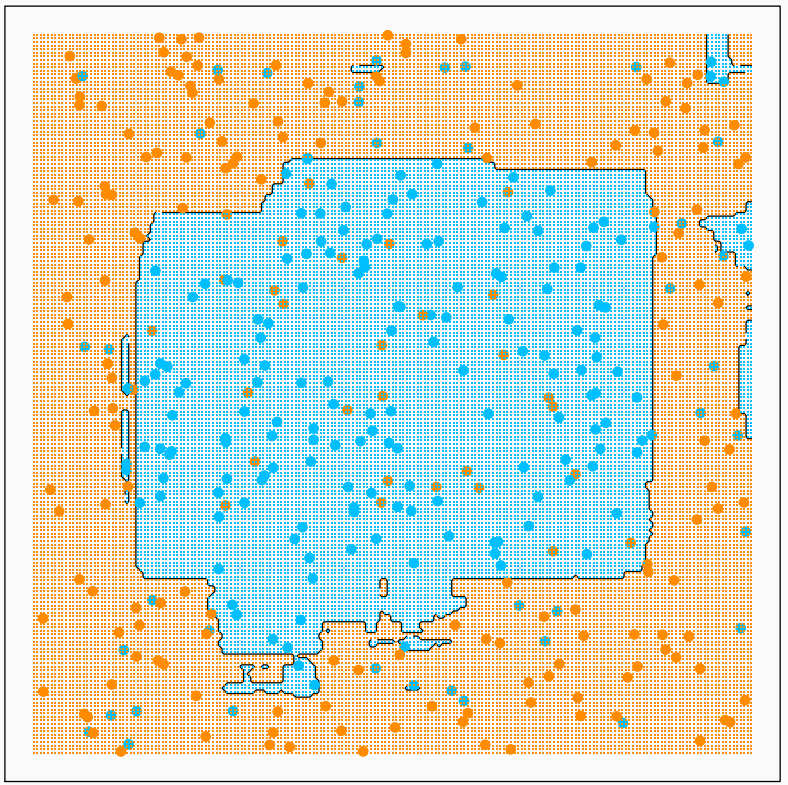
and for regression problems:

$$\widehat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B}\widehat{f}_b(x),$$

- Dramatically reduce the variance of individual learners
- CART can be replaced by other weak learners

CART vs. Bagging





CART vs. Bagging on our previous example, both are pruned

Remarks about Bagging

- Bagging can dramatically reduce the variance of unstable “weak learners” like trees, leading to improved prediction
- The simple structure of trees will be lost due to bagging, hence it is not easy to interpret
- Different trees have high correlation which makes averaging not very effective

Random Forests

- “The Random Subspace Method for Constructing Decision Forests” by Ho (1998) greatly influenced Breiman’s idea of random forests
- In Ho’s method, each tree is constructed using a randomly selected subset of features
- Random forests take a step forward: each splitting rule only consider a random subset

Tuning parameters: $mtry$

- An important tuning parameter of random forests is $mtry$
- At each split, randomly select $mtry$ variables from the entire set of features $\{1, \ldots, p\}$
- Search the best variable and splitting point out of these $mtry$ variables
- Split and proceed to child nodes
- This procedure turns out working remarkably well, even in high-dimensional data

Tuning parameters: n_{min}

- Another important tuning parameter is n_{min} (terminal node size)
- Random forests does not perform pruning anymore
- Instead, splitting does not stop until the terminal node size is less or equal to n_{min}, and the entire tree is used.
- n_{min} controls the trade-off between bias and variance in each tree
- In the most extreme case, $n_{min} = 1$ means exactly fit each observation, but this is not 1-NN!

Tuning parameters

- A summary of important tuning parameters in Random forests (using R package randomForest)

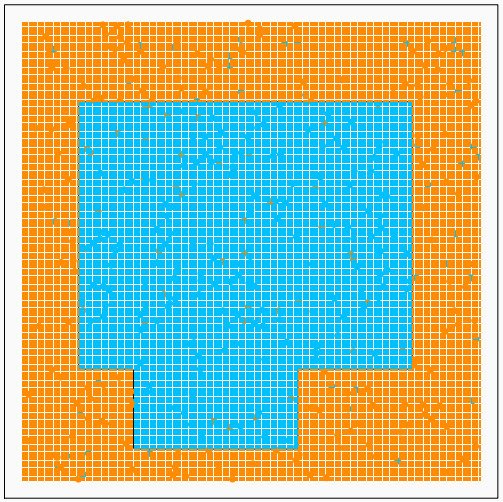
– ntree : number of trees, set it to be large. Default 500.

– mtry : number of variables considered at each split. Default $p/3$ for regression, $\sqrt{p}$ for classification.

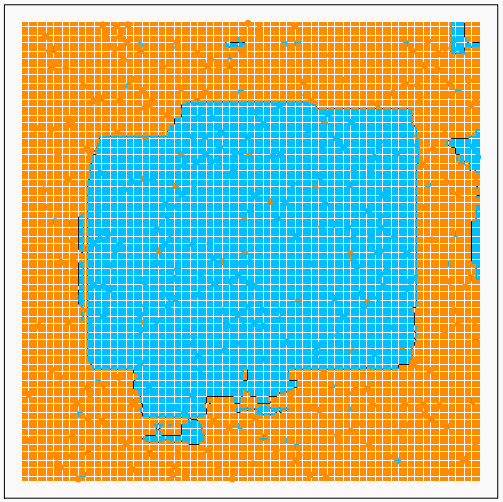
– nodesize : terminal node size, same as n_{min}. Default 5 for regression, 1 for classification

- Overall, tuning is quite crucial in random forests

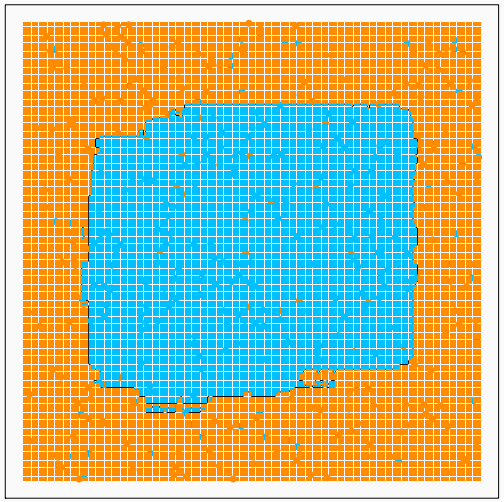
CART vs. Bagging vs. RF



CART



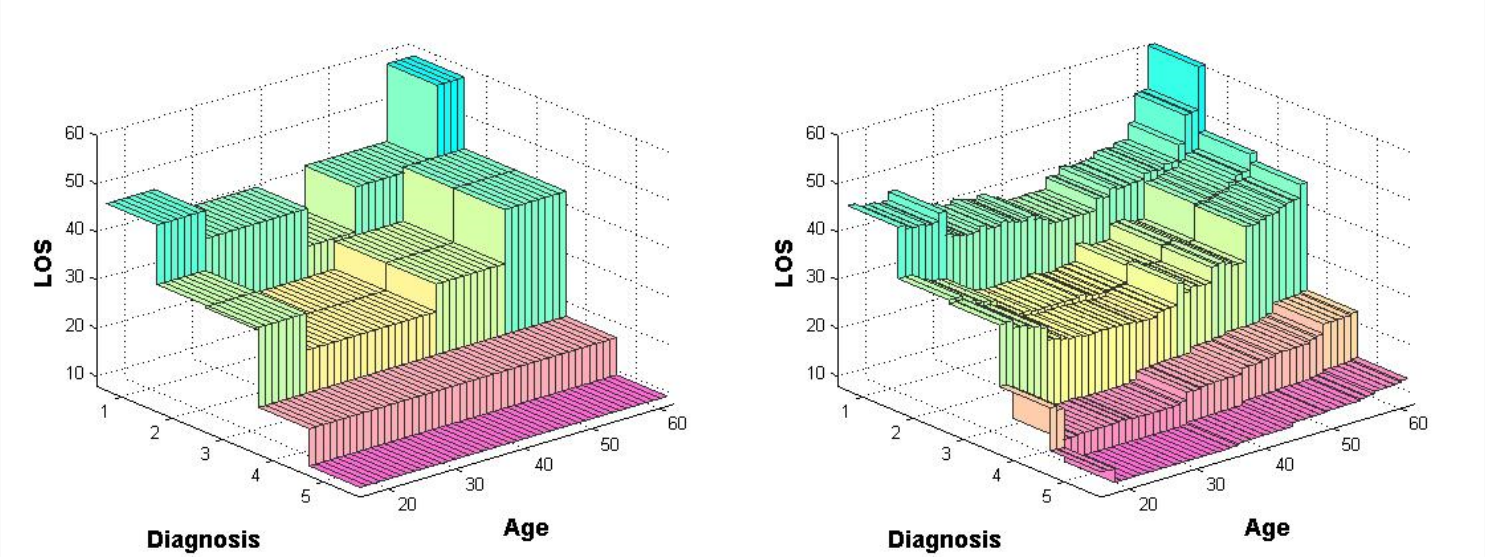
Bagging



RF

RF: ntree = 1000, mtry = 1, nodesize = 25

CART vs. RF in regression



CART RF

Model patients' length of stay (LOS) in hospital: Diagnosis (categorical) and Age (continuous)