Contents

1 Tree-Based methods
- Motivations

Contents

1 Tree-Based methods
- Motivations
- Tree building process

2 Ensembling with trees : Boosting and Randomization

Contents

1 Tree-Based methods
- Motivations
- Tree building process

2 Ensembling with trees : Boosting and Randomization

Contents

1 Tree-Based methods
- Motivations
- Tree building process

2 Ensembling with trees : Boosting and Randomization

Contents

1 Tree-Based methods
- Motivations
- Tree building process

2 Ensembling with trees : Boosting and Randomization

Contents

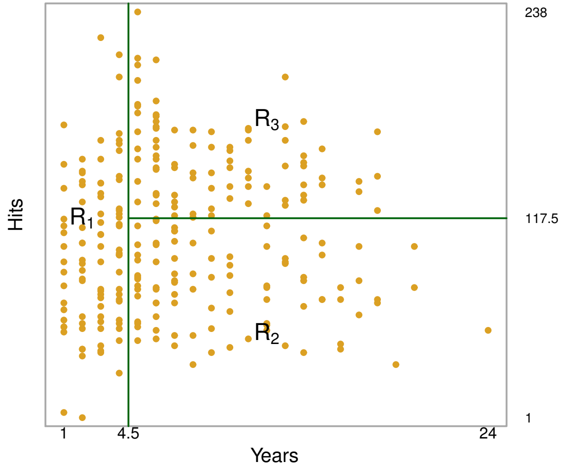
1 Tree-Based methods
- Motivations
- Tree building process

2 Ensembling with trees : Boosting and Randomization

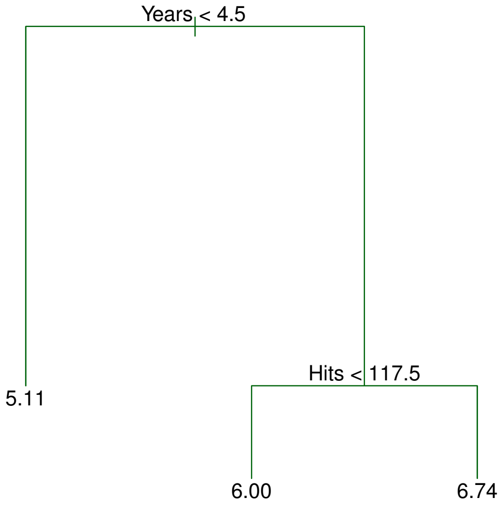
Outline

1. Tree-Based methods
   - Motivations
   - Tree building process
2. Ensembling with trees : Boosting and Randomization
   - Bagging
   - Randomisation : Random Forest algo
   - Randomisation : Gradient Boosting algo
   - Summary

- Parametric setup is not adapted for all situations : Tree-Based methods and other non-parametric are more oriented on rules in order to separate regions of the sample space ;
- These methods works for both Regression and Classification ;
- Example : Regression tree for Baseball player's salary. Predict player's salary with two predictors Hits, Years. Here we predict the Log(Salary)



Samples used to compute the regression tree described later



Rule oriented prediction (if true go on the left)

Outline

1. Tree-Based methods
   - Motivations
   - Tree building process
2. Ensembling with trees : Boosting and Randomization
   - Bagging
   - Randomisation : Random Forest algo
   - Randomisation : Gradient Boosting algo
   - Summary

As usual, the method could be translated in a minimization process :

- In theory, the regions could have any shape. However, we choose to divide the predictor space into high-dimensional rectangles, or **boxes**, for simplicity and for ease of interpretation of the resulting predictive model.
- The goal is to find boxes $R_1, ..., R_J$ that minimize the RSS, given by
  $\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$
  where $\hat{y}_{R_j}$ is the mean response for the training observations within the jth box R_j.
- Unfortunately, it is computationally infeasible to consider every possible partition of the feature space into J boxes.
- For this reason, we take a **top-down**, **greedy** approach that is known as recursive binary splitting.

More details on the tree-building process :

- The approach is **top-down** because it begins at the top of the tree and then successively splits the predictor space ; each split is indicated via two new branches further down on the tree.
- It is **greedy** because at each step of the tree-building process, the best split is made at that particular step, rather than looking ahead and picking a split that will lead to a better tree in some future step.
- We first select the predictor X_j and the cutpoint s such that splitting the predictor space into the regions $\{X|Xj < s\}$ and $\{X|Xj \geq s\}$ leads to the greatest possible reduction in RSS.
- Next, we repeat the process, looking for the best predictor and best cutpoint in order to split the data further so as to minimize the RSS within each of the resulting regions.

More topics :

- Choice of a stop criterion ;
- Tree pruning ;
- Choice of the best subtree, using a penalisation parameter α which is tuned by cross-validation ;
- Classification tree : same principles but using a different cost function : Gini or Cross-Entropy.
- See slide Advantages and Disadvantages of Trees

Contents

1 Tree-Based methods

Contents

1 Tree-Based methods

2 Ensembling with trees : Boosting and Randomization

Contents

1 Tree-Based methods

2 Ensembling with trees : Boosting and Randomization
- Bagging

Contents

1 Tree-Based methods

2 Ensembling with trees : Boosting and Randomization
- Bagging
- Randomisation : Random Forest algo

Contents

1 Tree-Based methods

2 Ensembling with trees : Boosting and Randomization
- Bagging
- Randomisation : Random Forest algo
- Randomisation : Gradient Boosting algo

Contents

1 Tree-Based methods

2 Ensembling with trees : Boosting and Randomization
- Bagging
- Randomisation : Random Forest algo
- Randomisation : Gradient Boosting algo
- Summary

Outline

1 Tree-Based methods
- Motivations
- Tree building process

2 Ensembling with trees : Boosting and Randomization
- Bagging
- Randomisation : Random Forest algo
- Randomisation : Gradient Boosting algo
- Summary

Bagging

- We generate B different bootstrapped training data sets. We then train our method on the bth bootstrapped training set in order to get $\hat{f^{*b}}(x)$, the prediction at a point x. We then average all the predictions to obtain $\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f^{*b}}(x)$
- This process works directly for prediction trees ;
- For classification trees we use a **majority vote** among the decision trees to obtain the correct class.
- Weak learner paradigm

Outline

1 Tree-Based methods
- Motivations
- Tree building process

2 Ensembling with trees : Boosting and Randomization
- Bagging
- Randomisation : Random Forest algo
- Randomisation : Gradient Boosting algo
- Summary

- Random forests provide an improvement over bagged trees by way of a small tweak that decorrelates the trees. This reduces the variance when we average the trees.
- As in bagging, we build a number of decision trees on bootstrapped training samples.
- But when building these decision trees, each time a split in a tree is considered, a random selection of m predictors is chosen as split candidates from the full set of p predictors. The split is allowed to use only one of those m predictors.
- A fresh selection of m predictors is taken at each split, and typically we choose $m \approx \sqrt{p}$

Outline

1 Tree-Based methods
- Motivations
- Tree building process

2 Ensembling with trees : Boosting and Randomization
- Bagging
- Randomisation : Random Forest algo
- Randomisation : Gradient Boosting algo
- Summary

- Like bagging, boosting is a general approach that can be applied to many statistical learning methods for regression or classification. We only discuss boosting for decision trees.
- Recall that bagging involves creating multiple copies of the original training data set using the bootstrap, fitting a separate decision tree to each copy, and then combining all of the trees in order to create a single predictive model.
- Notably, each tree is built on a bootstrap data set, independent of the other trees.
- Boosting works in a similar way, except that the trees are grown sequentially : each tree is grown using information from previously grown trees.
- **XgBoost** is the state-of-the-art implementation see github

Outline

1 Tree-Based methods
- Motivations
- Tree building process

2 Ensembling with trees : Boosting and Randomization
- Bagging
- Randomisation : Random Forest algo
- Randomisation : Gradient Boosting algo
- Summary

- Decision trees are simple and interpretable models for regression and classification
- However they are often not competitive with other methods in terms of prediction accuracy
- Bagging, random forests and boosting are good methods for improving the prediction accuracy of trees. They work by growing many trees on the training data and then combining the predictions of the resulting ensemble of trees.
- The latter two methods, random forests and boosting, are among the state-of-the-art methods for supervised learning. **However their results can be difficult to interpret**.