Decision Trees

Introduction

Regression Trees

Classification trees

Aggregating trees

- Bagging
- Random forest

Recommended Reading

- An Introduction to Statistical Learning with Applications in R (ISLR), Chapter 8
- Elements of Statistical Learning (ESL), Chapters 9 and 15

Introduction

Regression Trees

Classification trees

Aggregating trees

- Bagging
- Random forest

Introduction

- Decision trees
  - Regression trees - continuous response variable
  - Classification trees - categorical response variable

- Decision/Prediction rule
  - Segment the predictor space into regions
  - Usually mean or mode of the training observations in the region where the given observation belongs
  - Collection of rules can be summarized as trees, hence the name

- Combining several decision trees improve prediction accuracy
  - Bagging
  - Random Forests
  - Boosting

Pros and Cons of Decision Trees

- simple, easy to interpret
- handles mixed data types (categorical, quantitative) easily
- captures interactions among predictors
- captures nonlinear relationship between response and predictors

- predictive accuracy often not competitive
- unstable (little perturbation to data leads to very different models)
- tree ensembles trades off interpretability for prediction accuracy

Acknowledgement

Some of the figures in this presentation are taken from ”An Introduction to Statistical Learning, with applications in R”

Introduction

Regression Trees

Classification trees

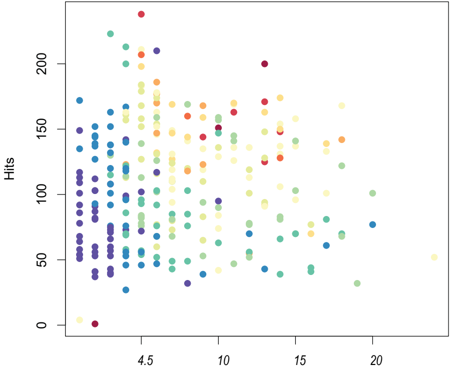
Aggregating trees

- Bagging
- Random forest

Example: Baseball Data

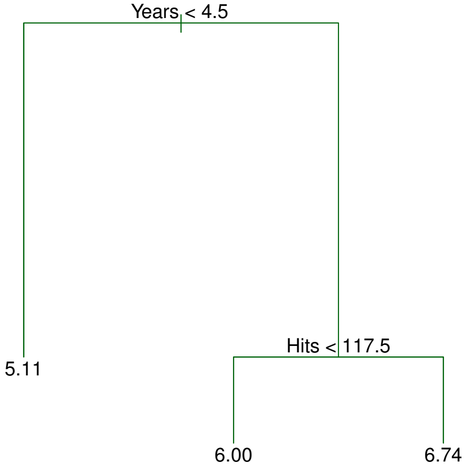
Goal:

Predict baseball players' salaries using their recent performance and history

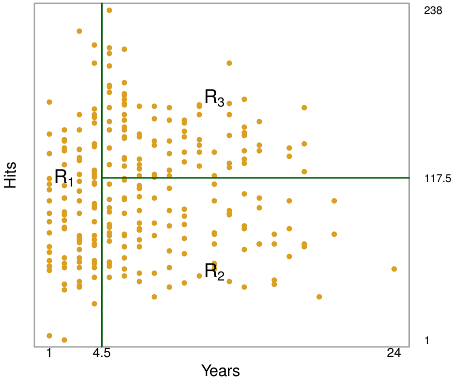


1987 Salaries of 322 major league players (color-code: high salary in red/yellow, low salary in blue/green) plotted across player's number of hits in 1986, and number of years in major league

Regression Tree on Baseball data



Regression Tree partitions predictor space



Tree segments predictor space into a partition with three regions: $R_1 = \{X | Years < 4.5\}$, $R_2 = \{X | Years \geq 4.5,\ Hits < 117.5\}$ and $R_3 = \{X | Years \geq 4.5,\ Hits \geq 117.5\}$

Regression trees, more formally

Regression: Response $y \in \mathbb{R}$, predictors $X = (X_1, \ldots, X_p) \in \mathbb{R}^p$

At a high-level, two-step process:

- Divide predictor space $X_1, X_2, ..X_p$ into J non-overlapping regions $R_1, R_2, ..R_J$
- Every observation in a R_j has same prediction i.e. $\hat{y}_{R_j} = \frac{1}{n}\Sigma_{j \in R_j} y_j$ where n is the number of training observations in R_j.

A closer look at Step 1:

- We want to find boxes $R_1, R_2, ..R_J$ that minimize the RSS given by

$$RSS = \sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

- Unfortunately, minimization problem computation intensive
- Method: Recursive binary splitting

Recursive binary splitting

- Top-down recursive greedy approach
- Top-down recursive: Start with all observations and split into two branches at each level of the tree
- Greedy: Best split is made at each step without looking ahead

- Choose a predictor X_j and a cutpoint s that minimizes the RSS for the resulting tree

$$\begin{aligned} R_1(j,s) &= \{X|X_j < s\}, \quad R_2(j,s) = \{X|X_j \geq s\} \\ RSS &= \sum_{x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2 \end{aligned}$$

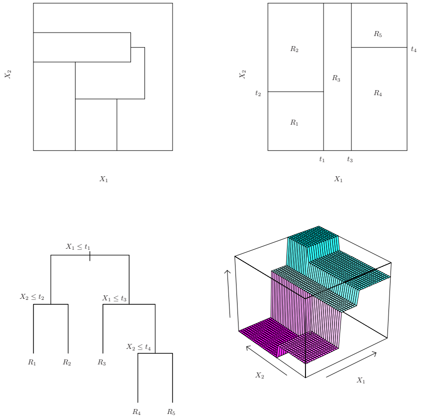
- This minimization problem can be solved efficiently!!

Prediction

Recall the two-step process:

- Divide predictor space $X_1, X_2, ..X_p$ into J non-overlapping regions $R_1, R_2, ..R_J$
- Every observation in a R_j has same prediction i.e. $\hat{y}_{R_j} = \frac{1}{n}\Sigma_{j \in R_j} y_j$ where n is the number of training observations in R_j.

Prediction with a Regression Tree



[Top Left]: A partition that could not result from recursive binary splits; **[Top Right]**: output of a recursive binary splits, **[Bottom Left]**: associated regression tree, **[Bottom Right]**: perspective plot of the response surface of the regression tree.

Tree pruning

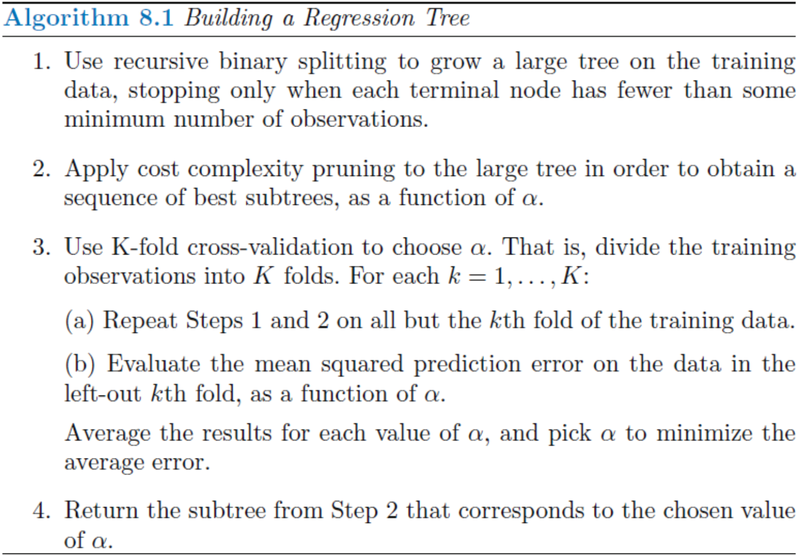
- Recursive binary splitting may be complex and overfit data
- Solution: try to obtain subtree that gives lowest test error rate
- Bias-variance tradeoff: Use a smaller tree - lower variance but some bias
- Method: Cost complexity pruning (with weakest link pruning)
  - Consider a sequence of subtree $\{T_\alpha\}$ indexed by tuning parameter $\alpha \geq 0$
  - For each α subtree T_α is the one that minimizes

$$\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$

where $|T|$ is the number of terminal nodes

- Choose α (or T_α) by cross-validation

Algorithm for building a pruned regression tree



Algorithm 8.1 *Building a Regression Tree*

1. Use recursive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations.

2. Apply cost complexity pruning to the large tree in order to obtain a sequence of best subtrees, as a function of α.

3. Use K-fold cross-validation to choose α. That is, divide the training observations into K folds. For each $k = 1, \ldots, K$:

   (a) Repeat Steps 1 and 2 on all but the kth fold of the training data.

   (b) Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of α.

   Average the results for each value of α, and pick α to minimize the average error.

4. Return the subtree from Step 2 that corresponds to the chosen value of α.

Introduction

Regression Trees

Classification trees

Aggregating trees

- Bagging
- Random forest

Classification trees

- Predict qualitative response
- Every observation in a R_j has same prediction i.e. $\hat{y}_{R_j}=$ most commonly occurring class of training observations in R_j.
- Find boxes $R_1, R_2, ..R_J$ that minimize a criteria such as E, G or D
  - Classification error rate E$= 1 - \max_k(\hat{p}_{mk})$
  - Gini index G$=\sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$
  - Cross-entropy D$= -\sum_{k=1}^{K} \hat{p}_{mk} log\, \hat{p}_{mk}$

where $\hat{p}_{mk} = \frac{no.\ of\ obs\ in\ k^{th}\ class}{no.\ of\ obs\ in\ R_m}$

Introduction

Regression Trees

Classification trees

Aggregating trees

- Bagging
- Random forest

Bagging (bootstrap aggregation)

- Aggregating information from many training set reduces variance of prediction
- **Generate B Boostrap samples from a training set**
- Grow deep trees (no pruning), one using each of the B bootstrap samples
- Use large B
- Predict Quantitative Y - use average prediction across B trees

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

- Predict Qualitative Y - use majority vote across B trees

Why Bagging?

- Deep trees have low bias, bagging reduces variance of *nonlinear* estimates
- More stable than a single decision tree
- Out-of-Bag error (OOB) estimation gives an idea of test set MSE
    - Every bootstrap sample has roughly 2/3 of training data, the other 1/3 are not used to build tree, and can be used as test data

Random forest

- De-correlates the tree
- At any split, find best split among only m *randomly selected* predictors, $m \leq p$
- Averageing uncorrelated prediction errors leads to substantial decline in variance
- For $m = p$: random forest is same as bagging
- When many correlated predictors, small m and large B give large reduction of test MSE

- Choice of m: Breiman suggested $\sqrt{p}$ for classification, $p/3$ for regression