Lecture 11 – Tree-based models – Part 1

Hamed Hasheminia

Tree-Based Methods

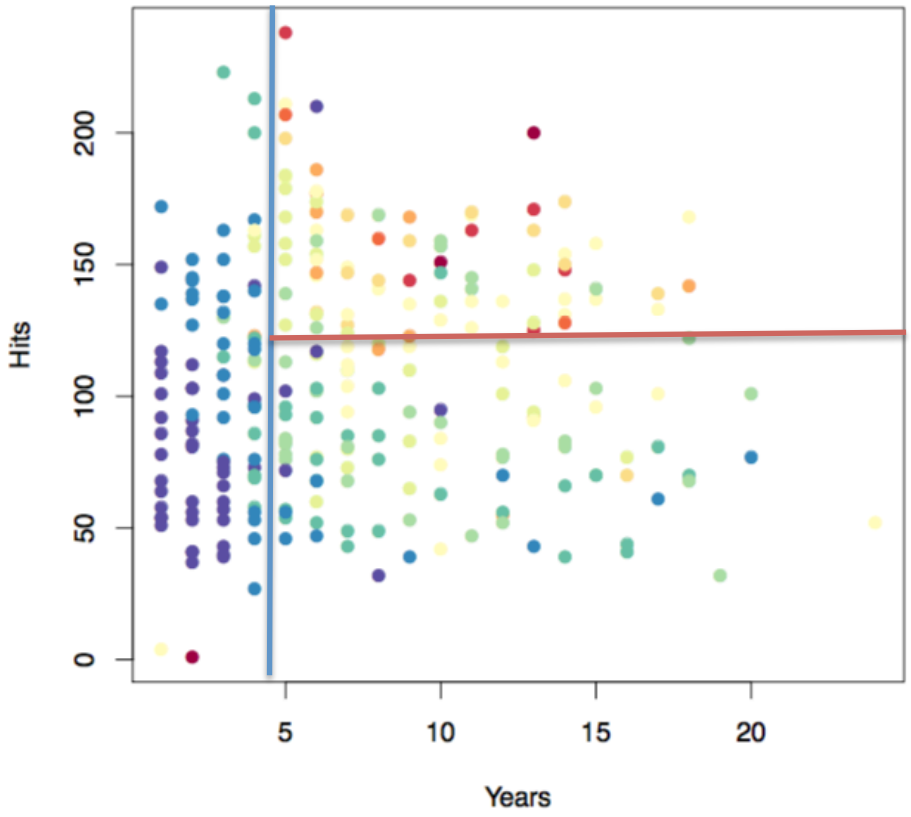
- Here we describe Tree-based methods for *regression* and *Classification*.
- These involve *Stratifying* or *Segmenting* the predictor space into a number of simple regions
- Since the set of splitting rules used to segment the predictor space can be summarized in a tree, these types of approaches are known as *decision-tree* methods.

Pros and Cons

- Tree-based methods are simple and useful for interpretation.
- However they typically are not as competitive as the best supervised learning approaches in terms of prediction accuracy.
- Hence we also discuss *bagging*, *random forests*, and *boosting* in future lectures.

Decision Trees – Regression Problems

- Salary in color-coded from low (blue, green) to high (yellow, red)

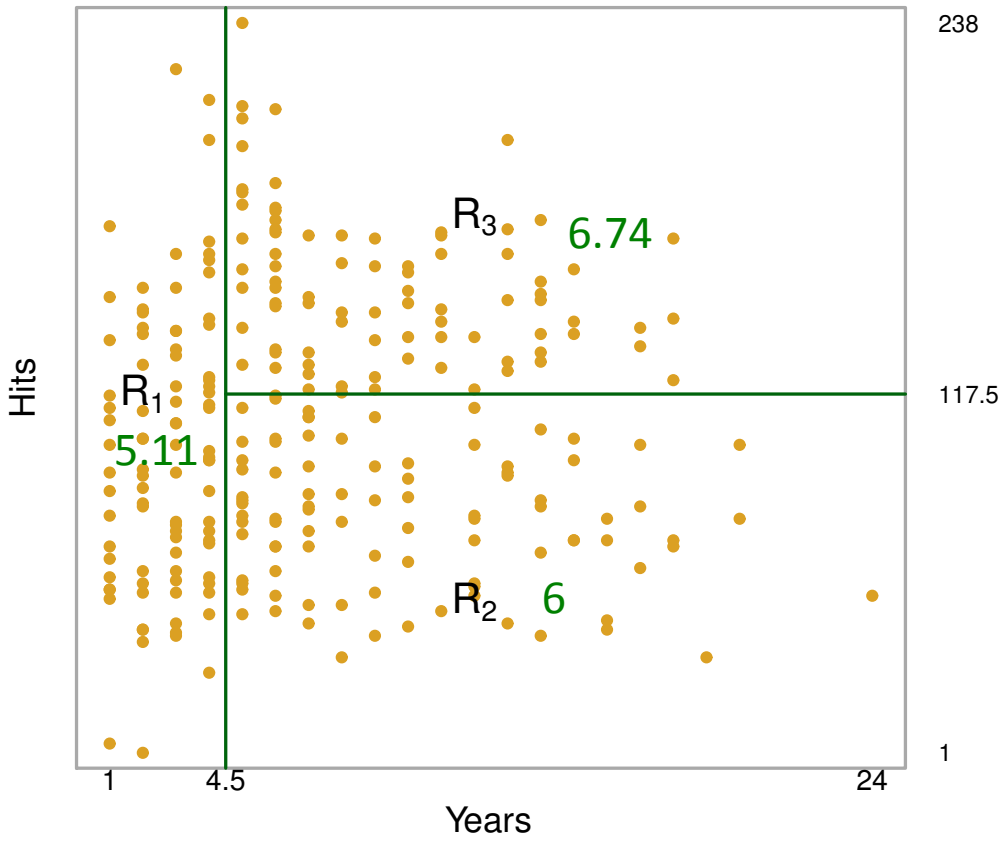


Decision Tree for these Data

Details of Previous Figure

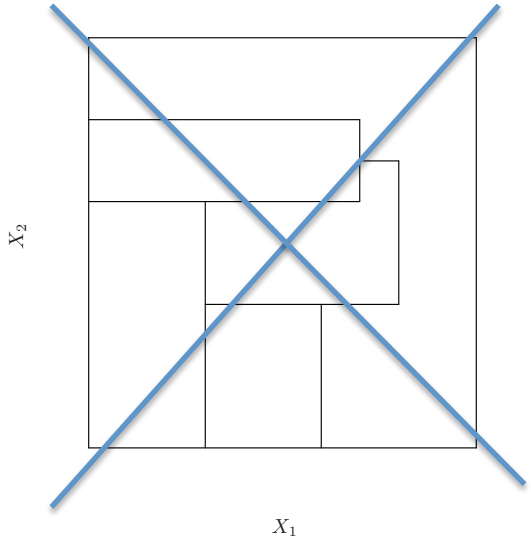
- At a given internal node, the label (*Variable < Constant*) indicates the left-hand branch emanating from that split, and the right-hand branch corresponds to *Variable >= Constant*). For instance, the split at the top of the tree results in two large branches. The left-hand branch corresponds to *Years < 4.5*, and the right-hand branch corresponds to *Years >= 4.5*
- The tree has two internal nodes and three terminal nodes, or leaves. The number in each leaf is the mean of the response for the observations that fall there.

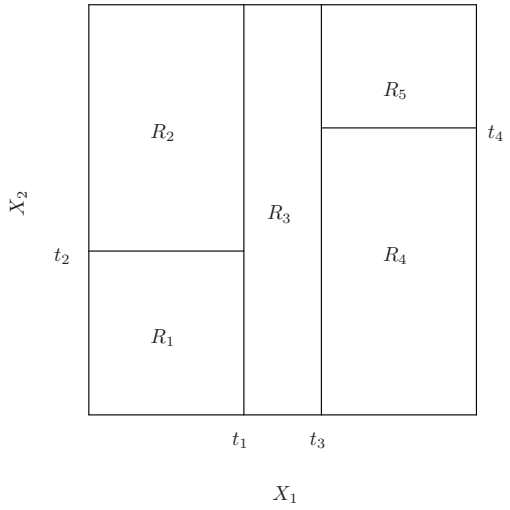
Results

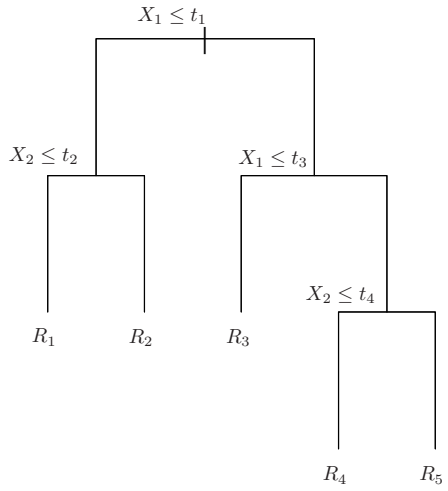


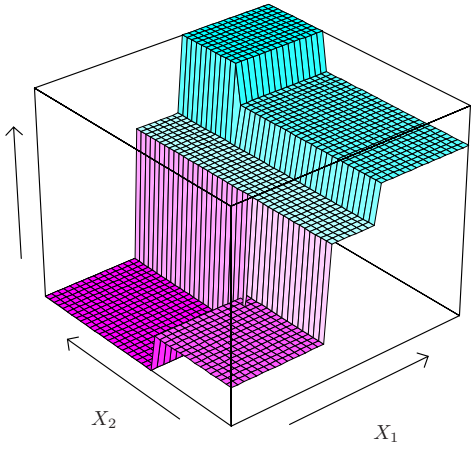
Details of tree-building Process

- We divide the predictor space — that is, the set of possible values for $X_1, X_2, \ldots, X_p$ — into J distinct and non-overlapping regions, $R_1, R_2, \ldots, R_J$.
- For every observation that falls into the region R_j, we make the same prediction, which is simply the mean of the response values for the training observations in R_j.
- 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.


X_2
X_1


R_1
R_2
R_3
R_4
R_5
t_1
t_2
t_3
t_4
X_2
X_1


$X_1 \leq t_1$
$X_2 \leq t_2$
$X_1 \leq t_3$
$X_2 \leq t_4$
R_1
R_2
R_3
R_4
R_5


X_2
X_1

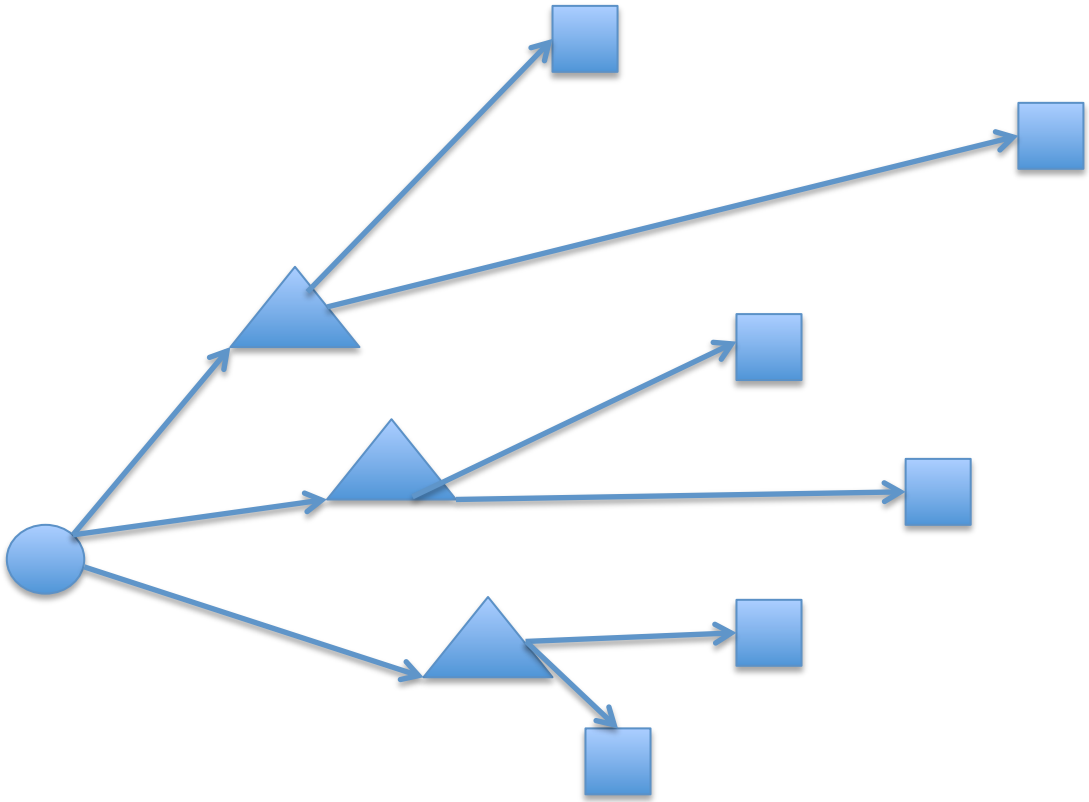
More details of the tree building process

- The goal is to find boxes $R_1,\ldots,R_j$ that minimize the RSS, given by

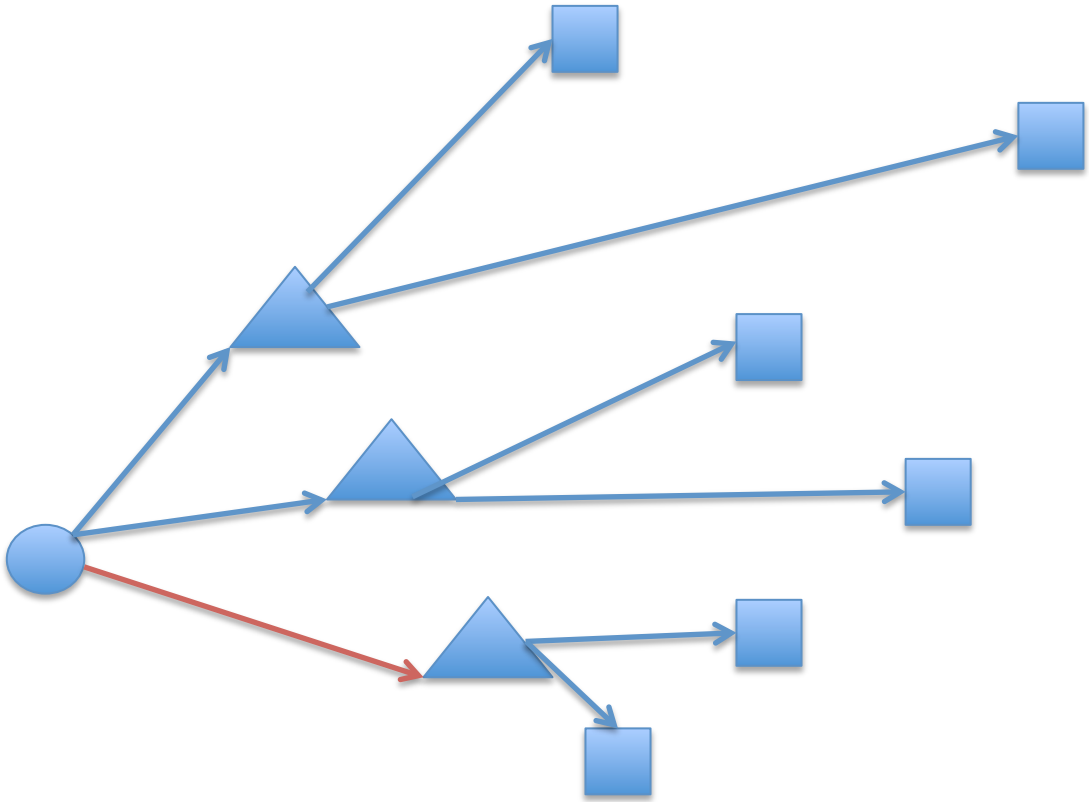
$$\sum_{j=1}^{J}\sum_{i\in R_j}(y_i-\hat{y}_{R_j})^2,$$

- Where $\hat{y}_{R_j}$ is the mean response for the training observations within the j^{th} box.
- Unfortunately, it is computationally infeasible to consider every possible partition of the feature space into J boxes.

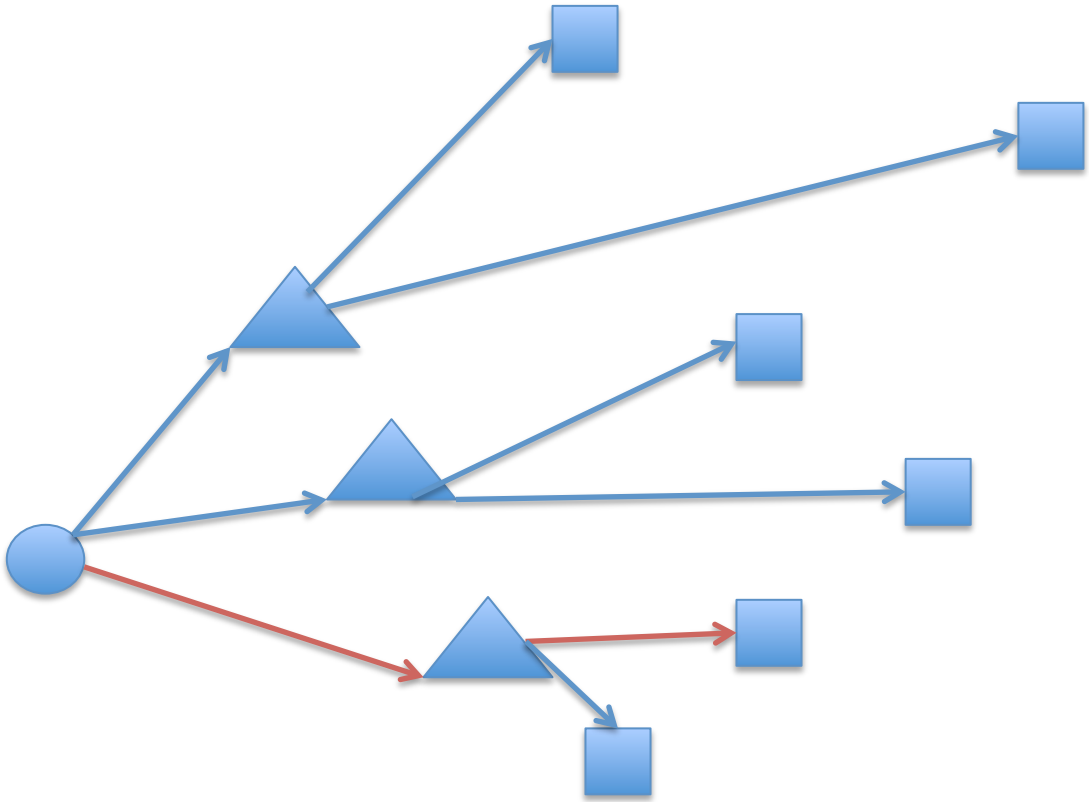
Greedy Algorithm



Greedy Algorithm

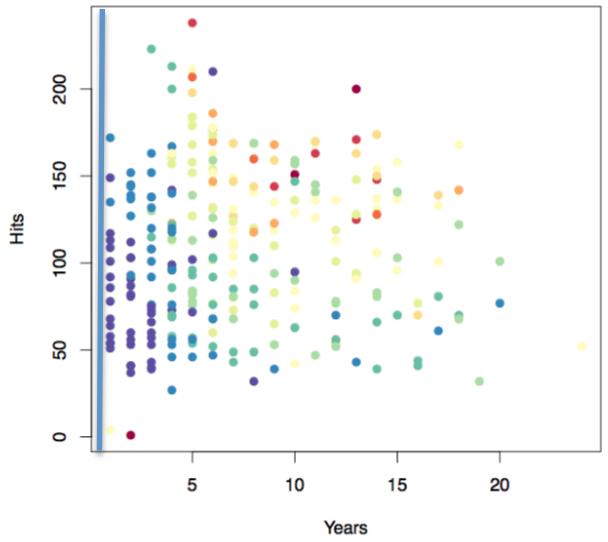


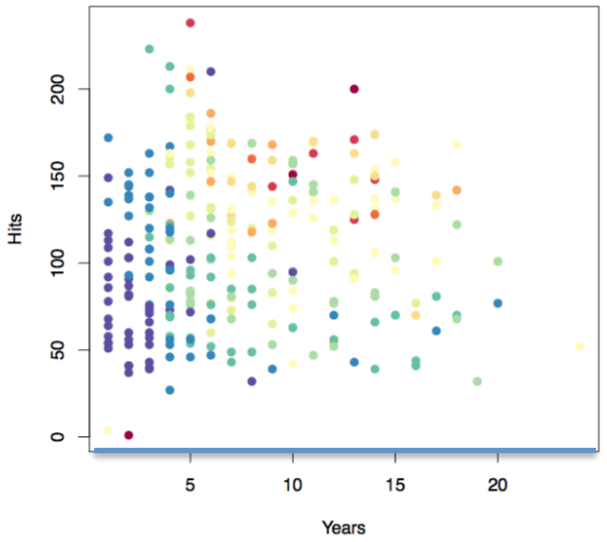
Greedy Algorithm

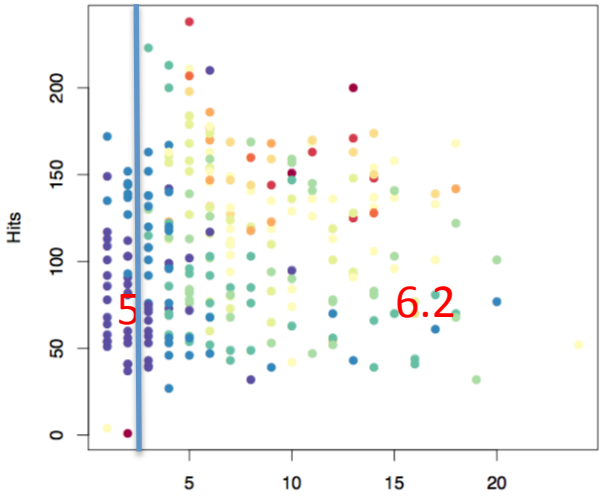


Top-Down Greedy Approach (also known as recursive binary splitting)

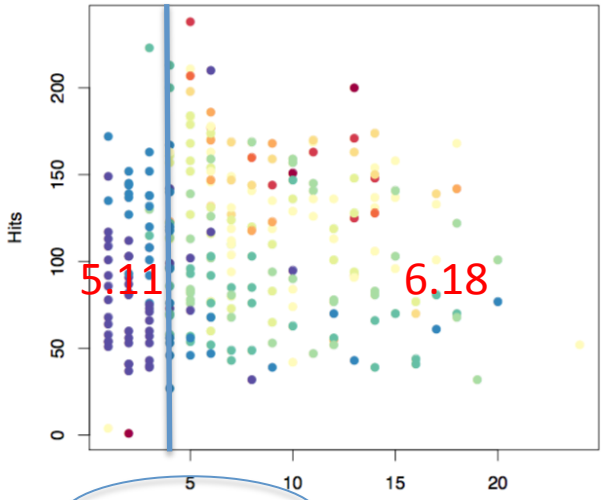
- We first select the predictor X_j and the cutpoint s such that splitting the predictor space into the regions $\{X|X_j < s\}$ and $\{X|X_j >= s\}$ leads to the greatest possible reduction in RSS.



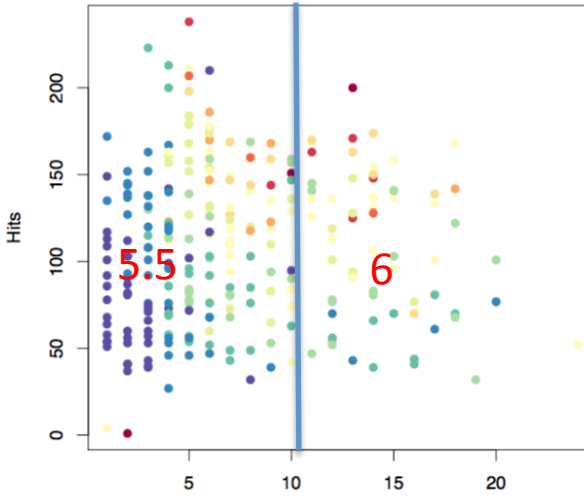




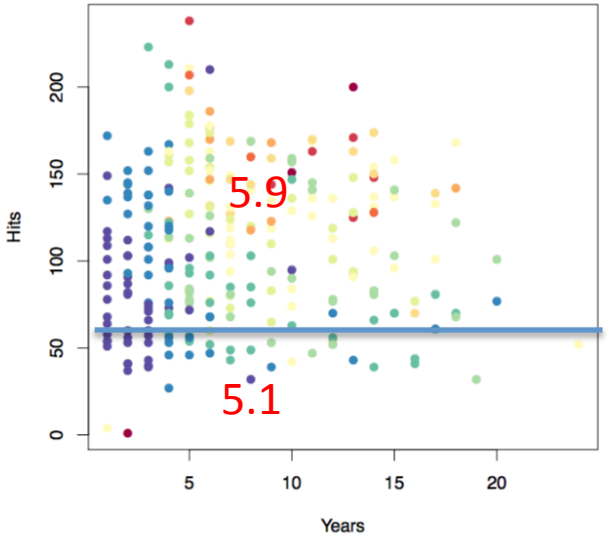
RSS = 1012



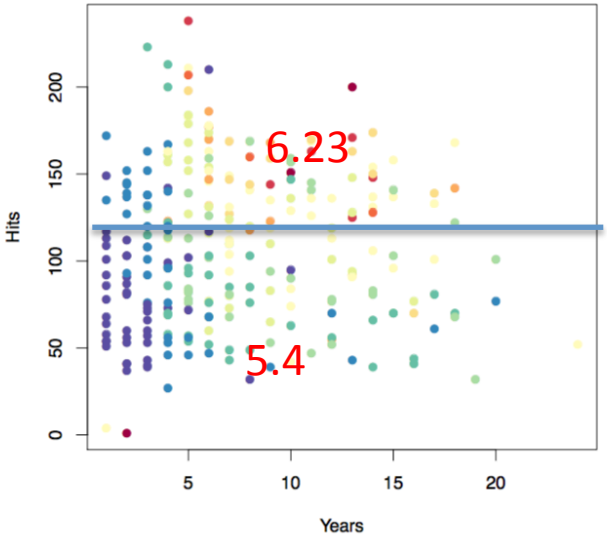
RSS = 953.2



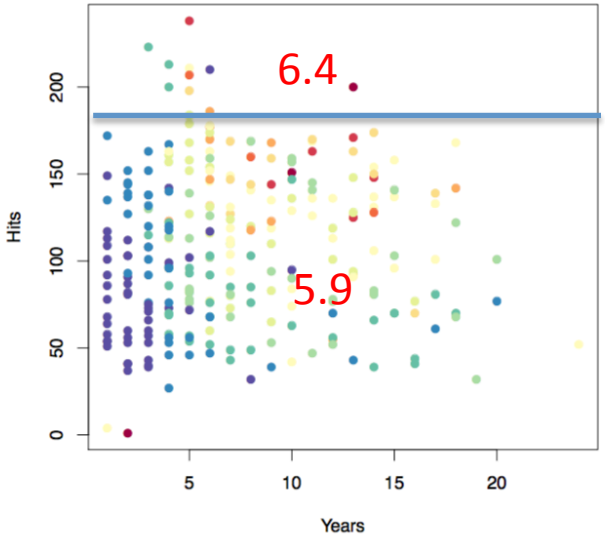
RSS = 1091.1



RSS = 1031.1



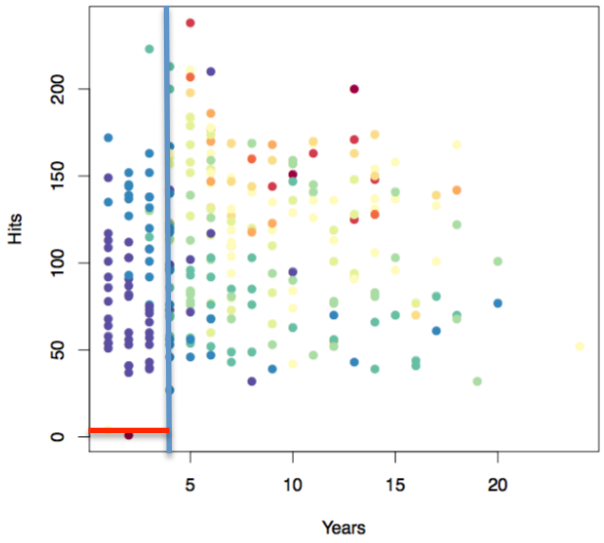
RSS = 1022.1

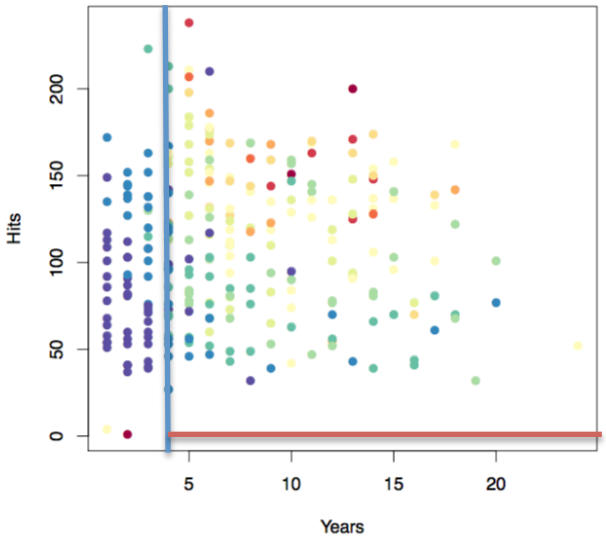


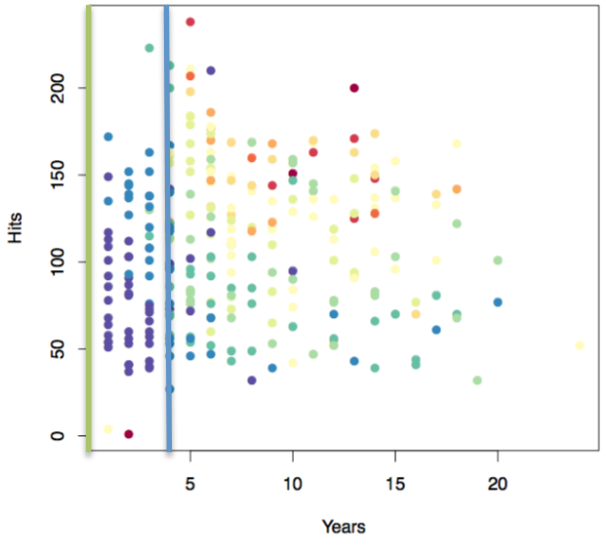
RSS = 1076.1

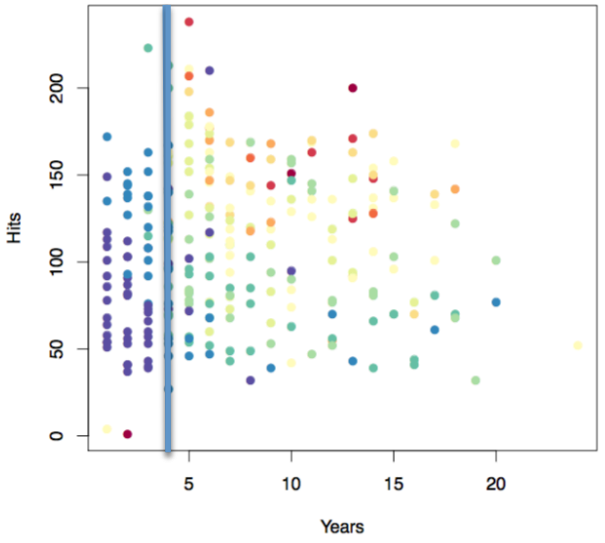
Top-Down Greedy Approach (also known as recursive binary splitting)

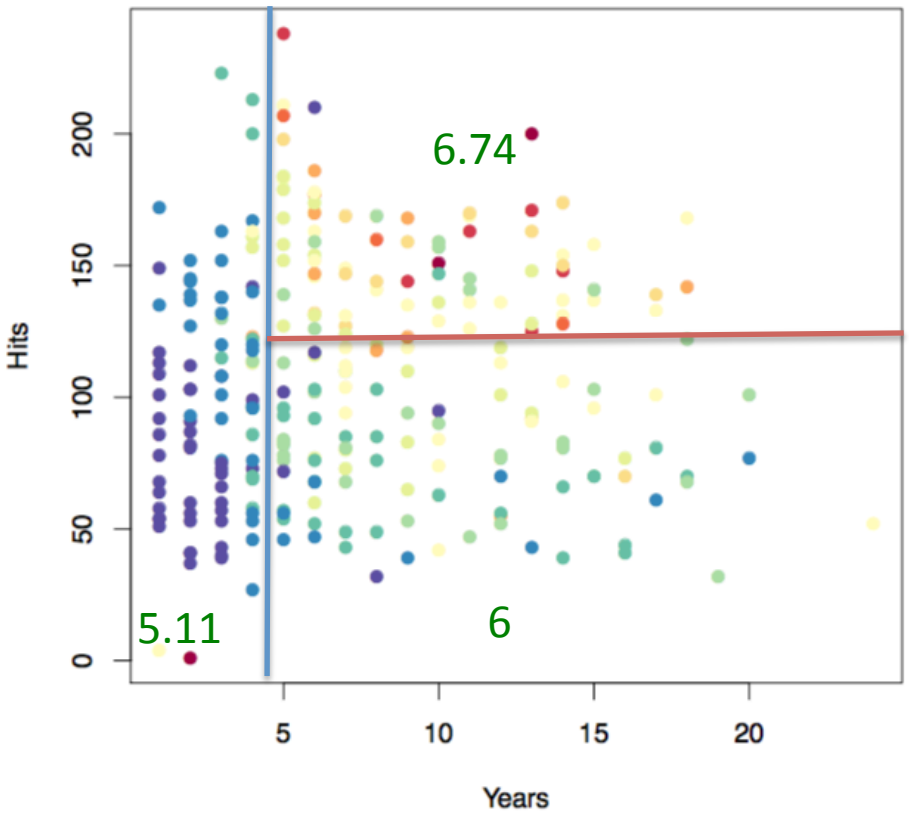
- Next, we repeat the process, looking for the best predictor and the best cutpoint in order to split the data further so as to minimize the RSS within each of the resulting regions.
- However, this time, instead of splitting the entire predictor space, we split one of the two previously identified regions. We now have three regions.


Hits
0
50
100
150
200
5
10
15
20
Years


Hits
0
50
100
150
200
5
10
15
20
Years

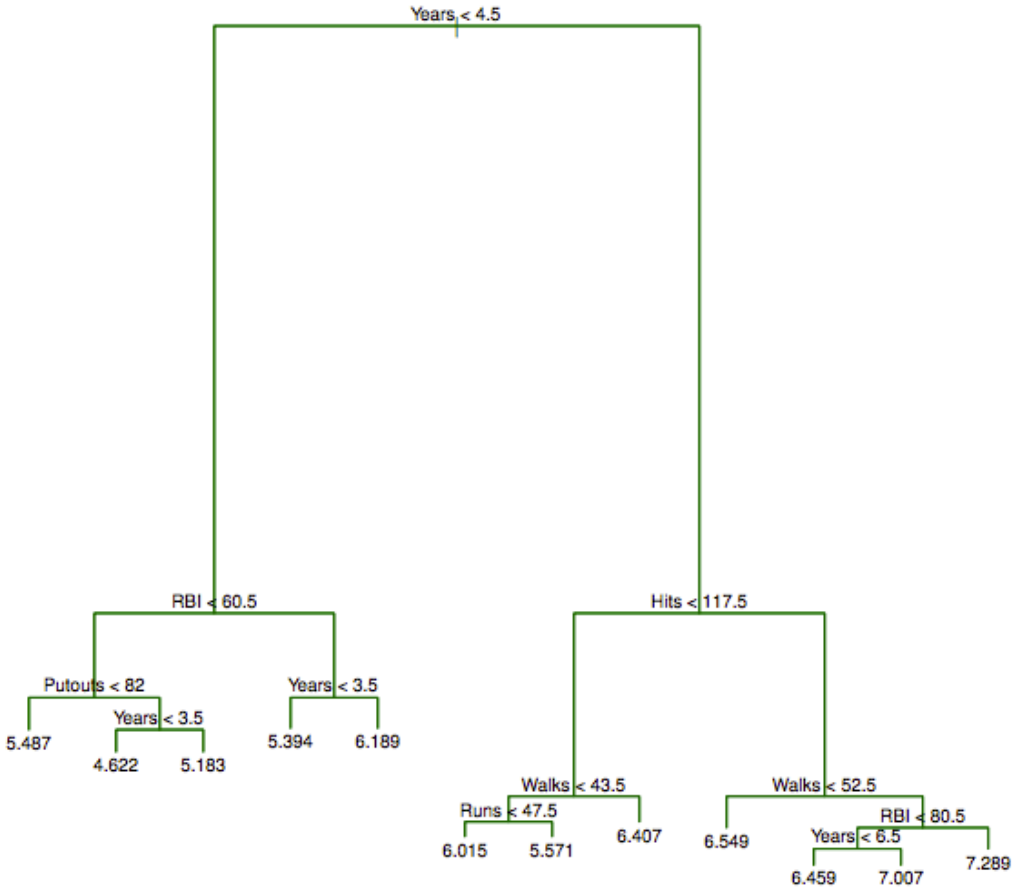

Hits
0
50
100
150
200
5
10
15
20
Years

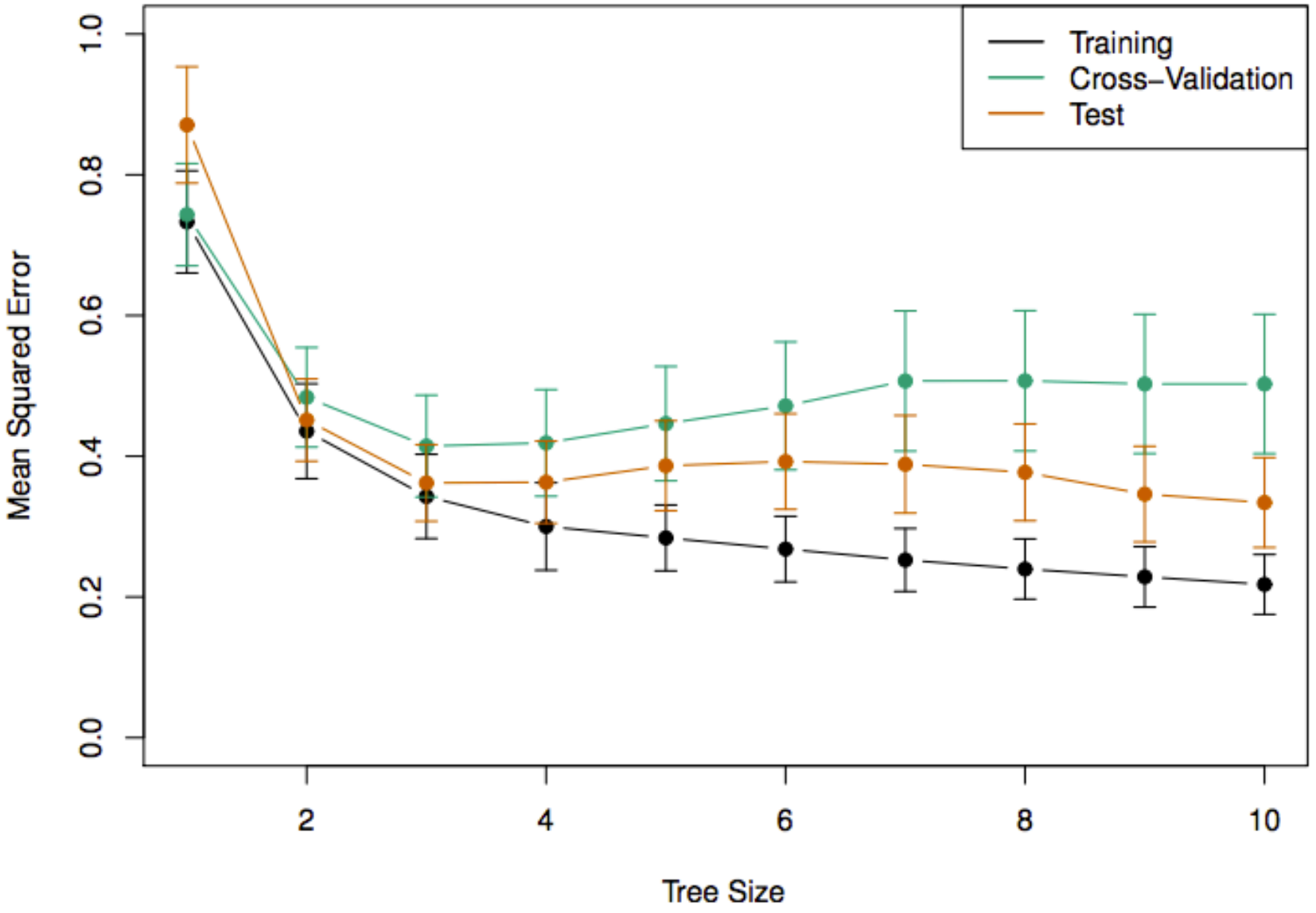

Hits
0
50
100
150
200
5
10
15
20
Years


Hits
0
50
100
150
200
5
10
15
20
Years
6.74
5.11
6


Years < 4.5
Hits < 117.5
5.11
6.00
6.74

Cross-Validation




Training
Cross–Validation
Test
Mean Squared Error
0.0
0.2
0.4
0.6
0.8
1.0
2
4
6
8
10
Tree Size

Classification Trees

- Very similar to a regression tree, except that it is used to predict a qualitative response rather than a quantitative one.
- For a classification tree, we predict that each observation belongs to the most commonly occurring class of training observations in the region to which it belongs.

Details of classification trees

A natural alternative to RSS is the *classification error rate*. this is simply the fraction of the training observations in that region that do not belong to the most common class:

$$E = 1 - \max_k(\hat{p}_{mk}).$$

Here $\hat{p}_{mk}$ represents the proportion of training observations in the mth region that are from the kth class.

Gini Index

The *Gini index* is defined by

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}),$$

a measure of total variance across the K classes. The Gini index takes on a small value if all of the $\hat{p}_{mk}$'s are close to zero or one.

For this reason the Gini index is referred to as a measure of node *purity* — a small value indicates that a node contains predominantly observations from a single class.

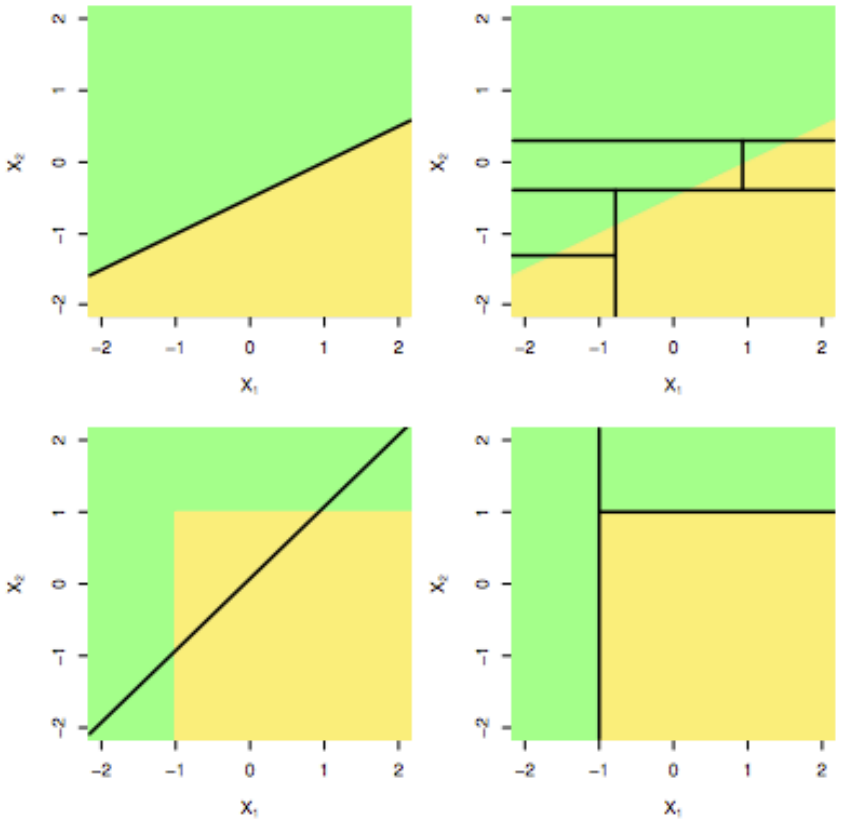
Cross-entropy

An alternative to the Gini index is *cross-entropy*, given by

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}.$$

It turns out that the Gini index and the cross-entropy are very similar numerically.

Trees Versus Linear Models



Top Row: True linear boundary; Bottom row: true non-linear boundary.

Left column: linear model; Right column: tree-based model

Summary

- Decision trees are simple and interpretable models for *regression* and *classification*
- However the are often not competitive with other methods in terms of prediction accuracy
- Unfortunately, it is computationally infeasible to consider every possible partition of the feature space into J boxes. Therefore, we use *top-down greedy approach* to solve it.

What is next?

- How can we increase predictability? (*Bagging, Boosting, Random Forest*)