Next two chapters: $Y = f(\underline{x}) + \varepsilon$

$f(\underline{x}) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$

$= \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \cdots + \beta_q X_p^2 X_2$

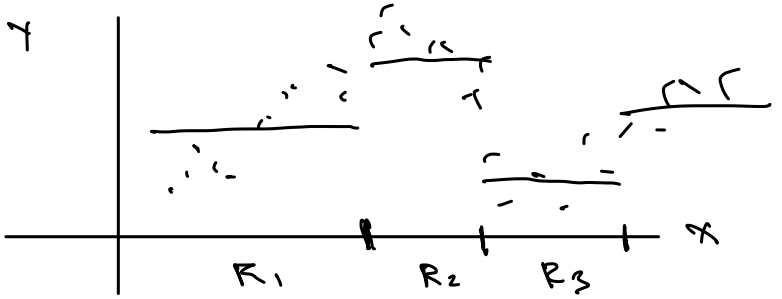
9 Tree-based methods

Trees work for both regression & classification by stratifying or segmenting feature space into disjoint regions, providing a simple predictor in each region.

- Trees are easy to explain & interpret
- " have nice graphical properties
- " can handle qualitative predictors
- " tend to perform poorly on prediction when compared to linear models [but there is a fix for this]

CART = Classification & Regression Trees

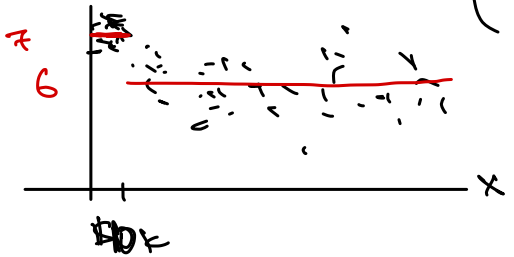
9.1 Regression trees



Ex y = rating of movie out of 10, x = budget in $s

simple regression tree might look like:

$$\hat{y} = \begin{cases} 7.03 & x < \$10{,}250 \\ 6.07 & x \geq \$10{,}250 \end{cases}$$



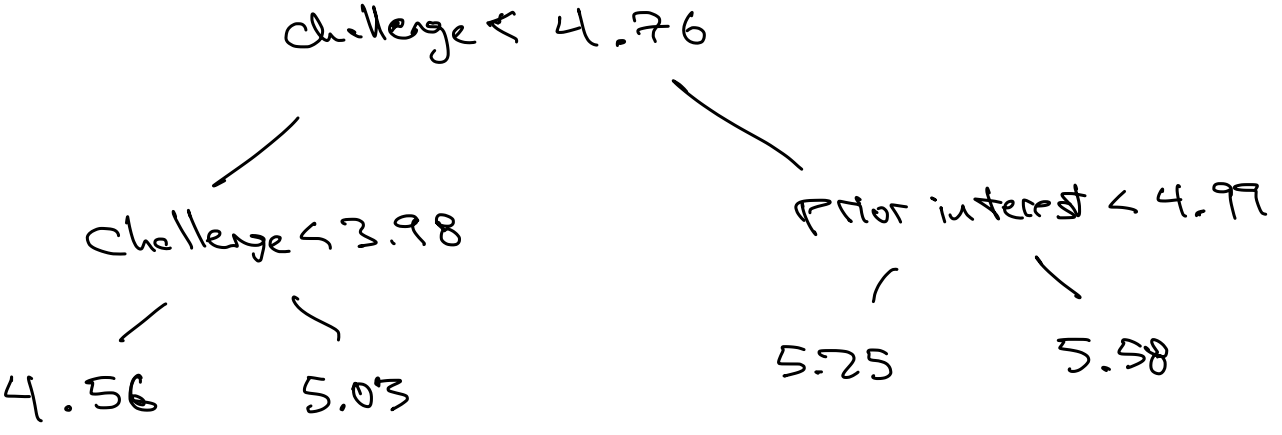
$x < \$10{,}250$ [stomp]

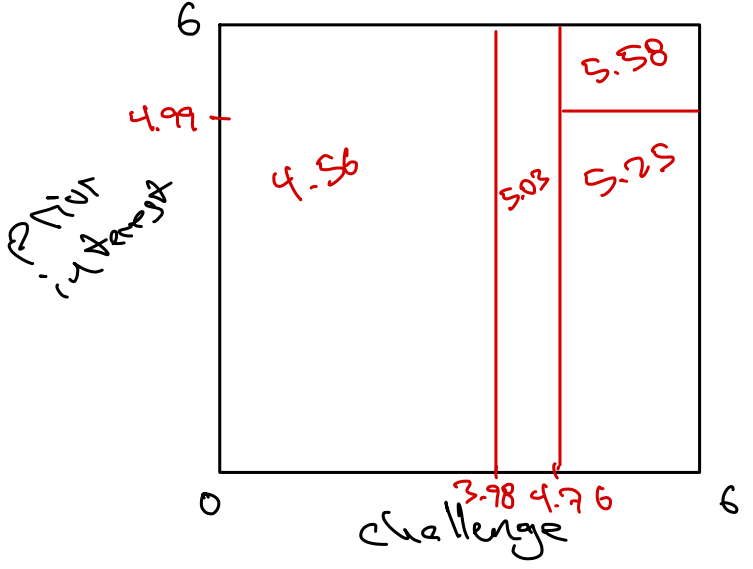
7.03 6.07

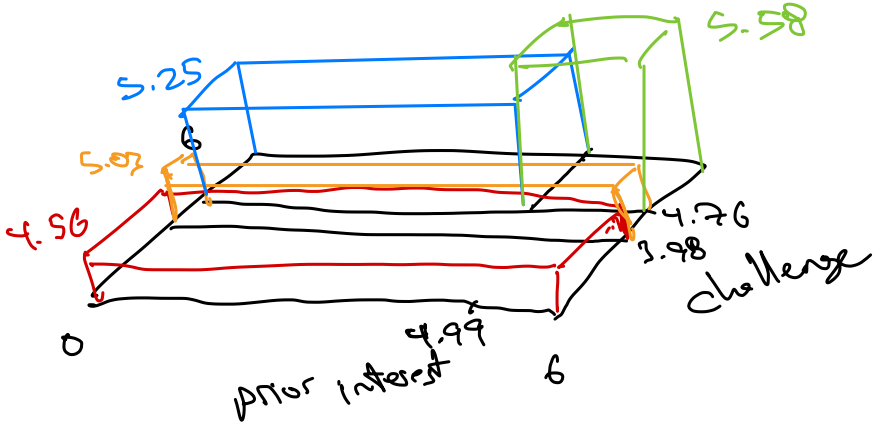
Ex Y = Instructor overall X_1 = challenge X_2 = prior interest

A fitted tree might be (1-6 scale)

- If challenge < 4.76
  - and challenge < 3.96 then $\hat{Y} = 4.56$
  - and challenge ≥ 3.96 then $\hat{Y} = 5.03$
- Else if challenge ≥ 4.76
  - and prior interest < 4.99, $\hat{Y} = 5.25$
  - and " " ≥ 4.99, $\hat{Y} = 5.58$


Challenge < 4.76
Challenge < 3.98
Prior interest < 4.99
4.56
5.03
5.25
5.58


6
4.99
Prior interest
4.56
5.03
5.58
5.25
0
3.98
4.76
6
Challenge



For continuous response Y & p features $X = \begin{pmatrix} x_1 \\ \vdots \\ x_p \end{pmatrix}$ a

regression tree is built via:

(1) Divide feature space into M disjoint regions $R_1, \ldots, R_M$

(2) To predict y in region R_m, use average of responses $\{y_i\}$ in R_m.

Model is of form

$$Y = f(\underline{x}) + \varepsilon = f(x_1, \ldots, x_p) + \varepsilon$$

where

$$f(\underline{x}) = \sum_{i=1}^{M} c_i \, 1[\underline{x} \in R_i]$$

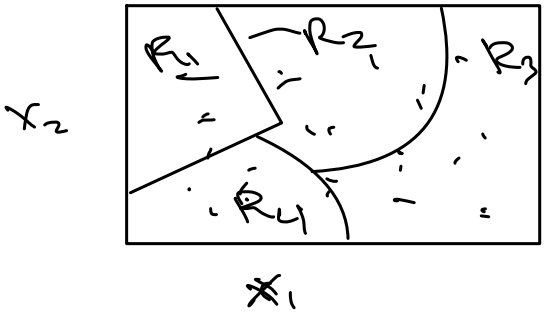
Given data $(\underline{x}_1, y_1), \ldots, (\underline{x}_n, y_n)$, want to estimate f, or equivalently, $c_1, \ldots, c_M$. If we use OLS and minimize

$$\sum_{i=1}^{n} (y_i - f(\underline{x}_i))^2$$

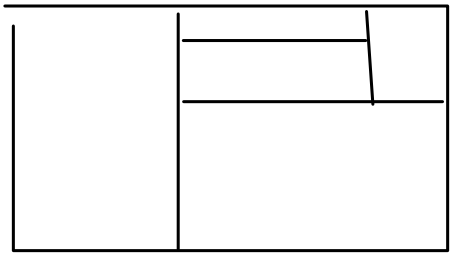
get

$$\hat{c}_i = \text{ave}\{ y_i \mid \underline{x}_i \in R_i \} \quad i = 1, \ldots, M$$

Problem How do we choose regions?



In practice the regions $R_1, \ldots, R_M$ are hyperrectangles



To find best rectangles, we use a splitting algorithm that is top-down greedy and uses recursive binary splitting.

First step, fix index m + split cutoff s, split domain into

$$R_1(m,s) = \{ \underline{x} \mid x_m < s \}$$

$$R_2(m,s) = \{ \underline{x} \mid x_m \geq s \}$$

and calculate

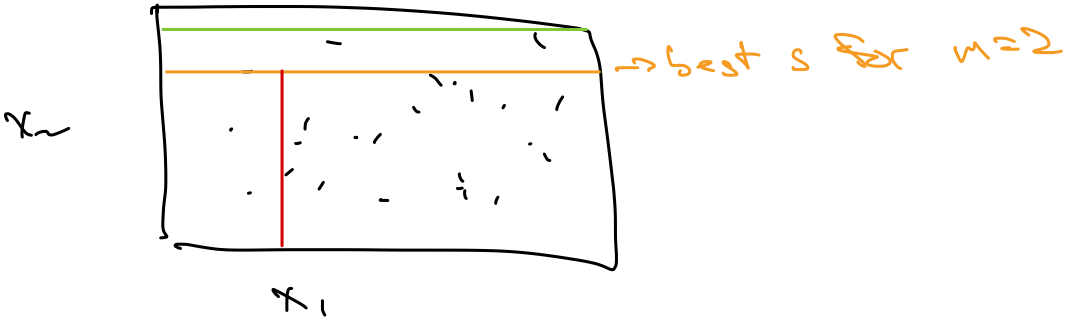
$$\min_{c_1} \sum_{\underline{x}_i \in R_1(m,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{\underline{x}_i \in R_2(m,s)} (y_i - c_2)^2$$

which happens at $\hat{c}_1 = \text{ave}\{ y_i \mid \underline{x}_i \in R_1(m,s) \}$

$\hat{c}_2 = \text{ave}\{ y_i \mid \underline{x}_i \in R_2(m,s) \}$

The final choice of (first) variable to split over & location of split is

$$\underset{(m,s)}{\text{argmin}} \left[\min_{c_1} \sum_{\underline{x}_i \in R_1(m,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{\underline{x}_i \in R_2(m,s)} (y_i - c_2)^2 \right]$$



Next step: Repeat same procedure within each $R_1(m,s)$ & $R_2(m,s)$, and iterate until some stopping criterion is reached [e.g. no fewer than 5 data points]