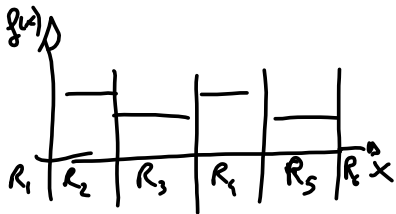


Trees (regression trees) $\rightarrow$ bagging (trees)
 $\rightarrow$ random forests $\rightarrow$ boosting



IDEA: approximate the function $f(x)$ with a piecewise constant

$$f(x) = \sum_{m=1}^M c_m I(x \in R_m)$$

We need to decide:

- M : how many splits in the x -axis (suppose $p=1$);
- R_m : where to place the splits;
- c_m : which value for the constant.

because we use the sum of squares as loss function

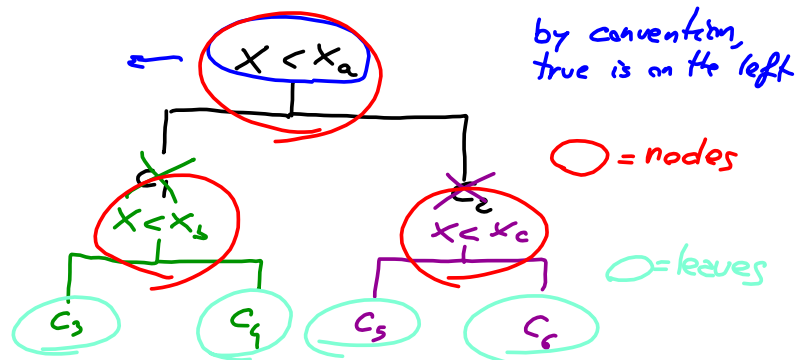
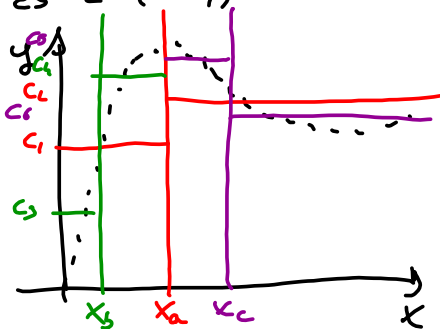
The last is the simplest: $c_m = \text{ave}(y_i | x_i \in R_m) = \frac{\sum_{i: x_i \in R_m} y_i}{\#x_i \in R_m}$ $\#$ = how many

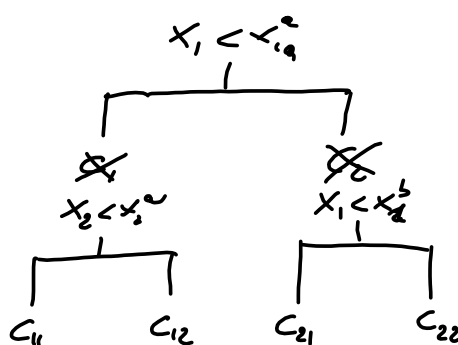
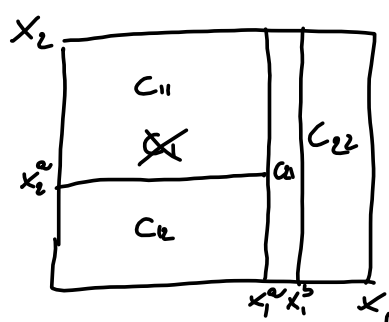
The number of splits M is related to the model complexity. Larger M , more complex the model $\rightarrow$ bias-variance trade-off. If $M=n$, complete overfitting.

Intuitively, we will make more splits where the function is steeper.

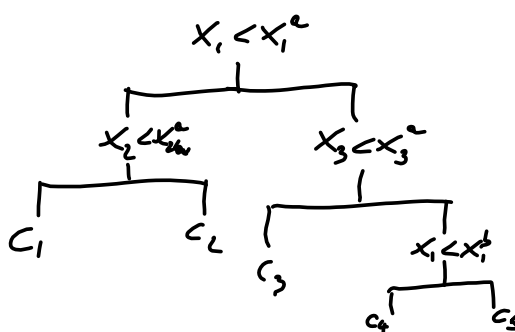
This idea scales well in higher dimensions. For example, when $p=2$, we will have regions R_m that are rectangular, but the idea is the same

The nice characteristic of this method is that the model can be represented as a (binary) tree





$p=2$



$p=3$

The tree representation is not as attractive as a 'normal plot', but:

- can be used for any value of p ;
- it is easy to storage (only need the nodes and the leaves)
- it is really easy to explain

To grow a tree

$$f(x) = \sum_{m=1}^M c_m I(x \in R_m)$$

where $c_m, m=1, \dots, M$ are constants; I is the indicator function (1 if $x \in R_m$, 0 otherwise); and $R_m, m=1, \dots, M$ are p -dimensional rectangles

Finding in one-shot both M and R_m is computationally infeasible, because we need to find:

- in which dimension we compute the split;
- ~~the~~ which value for the split

$\Rightarrow$ greedy approximation (we grow the tree step by step)

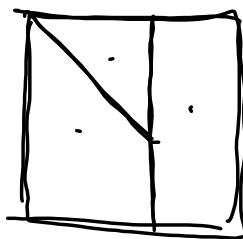
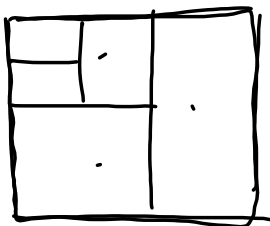
$$RSS = \sum_{i=1}^n (y_i - f(x_i))^2 = \sum_{m=1}^M \left(\sum_{i: x_i \in R_m} (y_i - c_m)^2 \right) = \sum_{m=1}^M RSS_m$$

$$\text{since } \arg \min_x \left(\sum_{i=1}^n (z_i - x)^2 \right) = \bar{z} \Rightarrow c_m = \frac{\sum_{i: x_i \in R_m} y_i}{\# x_i \in R_m}$$

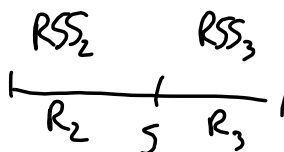
For each dimension of x , $j=1, \dots, p$

find the best split $s \rightarrow R_1(j, s), R_2(j, s)$

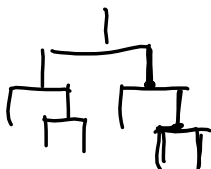
$$\min_{j, s} \left[\min_{c_1} \sum_{x_i \in R_1(j, s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j, s)} (y_i - c_2)^2 \right]$$



- we only perform myopic optimization (divide in two regions all the times)
- suboptimal (no guarantee to find the global minimum)
- acceptable solution among the computationally feasible ones.



$$s: \max_s \left(\underbrace{RSS_1 - (RSS_2 + RSS_3)}_{\text{diff}} \right)$$



we can proceed with the splitting until all leaves contain only one observation.
minimum RSS (train data) $\rightarrow$ overfitting

$\rightarrow$ we need to find a way when to stop

- we may decide to stop when the improvement in terms of RSS obtained by making a new split is smaller than desired. This does not really work, because a split with "small improvement" can lead to a larger split later.
- actually done: fit a tree as big as possible, potentially with one observation per leaf (most commonly a larger value, let us 5, is used), and then we remove the nodes that are not useful $\rightarrow$ pruning

To prune a tree

minimizing the RSS leads to overfitting
 SOLUTION: add a penalty term to the loss function that penalize complexity

$$C_\alpha(T) = \sum_{n=1}^{|T|} \text{RSS}_m^{(T)} + \alpha |T|$$

book's notation

↳ $n_m Q_m(T)$

where $n_m = \# x_i \in R_m$

$$Q_m(T) = \frac{1}{n_m} \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2$$

Where α is the tuning parameter:

- it can be shown that for each α there is a unique tree which minimizes $C_\alpha(T)$ (Breiman et al, 1984, Proposition 3.7)

⇒ we need to find the best value for α to find the best tree
 cross-validation ↗ smallest test error

Prediction of a new observation

- starting from the root node, we follow the path (T/F for each node) until we reach a leaf. The value associated with that leaf is the predicted value

Example: $\hat{f}(x_0 = 1.2) = 0.5359$ (right-left-right)
 $1.2 \not\geq 0.56 \rightarrow 1.2 < 1.57 \rightarrow 1.2 \not\geq 0.71$

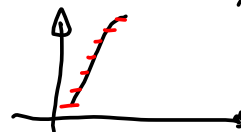
Discussion

→ Advantages of trees:

- simplicity and easiness of communication
- compact formulation → (portable)
- fast to compute (thanks to the greedy approach)
- easiness to extend to other loss functions
- insensible to presence of missing data
- easy to combine continuous and dichotomous/categorical covariates
- automatic variable selection (those variables that are not used in any nodes can be completely removed)

→ Disadvantages

- instability of the results (high variance) → bagging, random forest
- cannot be used for online regression (if I get a new observation, I have to start over)
- hard to approximate very steep functions
- there are no statistical procedure associated (difficult to quantify the uncertainty)
- we do not have a good way to understand the importance of a variable



Bagging

- we use bootstrapping to improve the estimate

Bootstrap **AGG**regret **ING**

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

where $\hat{f}_b^*$, $b = 1, \dots, B$ is the prediction obtained on a bootstrap sample b

$$Z = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$$

$$Z_b^* = \{(x_1^*, y_1^*), \dots, (x_n^*, y_n^*)\}_{b=1, \dots, B}$$

Often, bagging is used with trees, $\hat{f}_b^*$ is a statistical tree.

The strongest improvement made by bagging (a method) is to reduce the variance same bias

Consider (x_i, y_i) , $i = 1, \dots, n$, independently generated from $Z \sim \mathcal{D}$

$$f_{\text{bag}}(x) = E_{\mathcal{D}}[\hat{f}^*(x)]$$

Note that $\hat{f}_{\text{bag}}(x)$ is an estimation of $f_{\text{bag}}(x)$

$$\begin{aligned} E_{\mathcal{D}}[(Y - \hat{f}^*(x))^2] &= E_{\mathcal{D}}[(Y - f_{\text{bag}}(x) + f_{\text{bag}}(x) - \hat{f}^*(x))^2] \\ &= E_{\mathcal{D}}[(Y - f_{\text{bag}}(x))^2] + E_{\mathcal{D}}[(f_{\text{bag}}(x) - \hat{f}^*(x))^2] \\ &\quad + 2 E_{\mathcal{D}}[(Y - f_{\text{bag}}(x)) \underbrace{(f_{\text{bag}}(x) - \hat{f}^*(x))}_{=0}] \\ &= E_{\mathcal{D}}[(Y - f_{\text{bag}}(x))^2] + \text{variance of } \hat{f}^*(x) \end{aligned}$$

$$E_{\mathcal{D}}[(Y - \hat{f}^*(x))^2] \geq E_{\mathcal{D}}[(Y - f_{\text{bag}}(x))^2]$$