

# Foundations of Machine Learning

## AI2000 and AI5000

FoML-36

Tree-based methods

Dr. Konda Reddy Mopuri

Department of AI, IIT Hyderabad

July-Nov 2025



భారతీయ సాంకేతిక విజ్ఞాన సంస్థ హైదరాబాద్  
भारतीय प्रौद्योगिकी संस्थान हैदराबाद  
Indian Institute of Technology Hyderabad



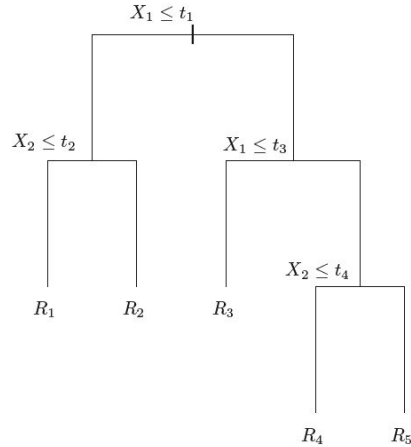
# So far in FoML

- Intro to ML and Probability refresher
- MLE, MAP, and fully Bayesian treatment
- Supervised learning
  - a. Linear Regression with basis functions
  - b. Bias-Variance Decomposition
  - c. Decision Theory - three broad classification strategies
  - d. Neural Networks
- Unsupervised learning
  - a. K-Means, Hierarchical, and GMM for clustering
- Kernelizing linear Models
  - a. Dual representation, Kernel trick, SVM (max-margin classifier)



For today

# Tree Based Learning Methods



Contents are taken from - [Intro to Statistical Learning](#)



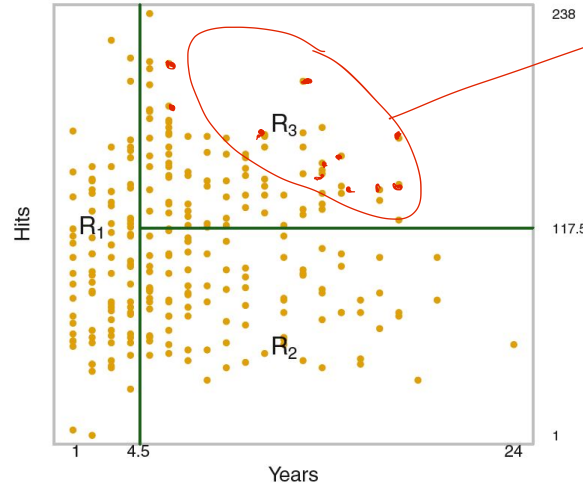
# Agenda

- Tree-based methods for
  - Regression
  - Classification
- Improvements
  - Bagging
  - Random Forests
  - Boosting



# Tree-based Methods

- Involve *stratifying* or *segmenting* the input (predictor) space



Higher La Larries

Figure credits: James et al. (ISLR)

# Tree-based Methods

- Prediction ← mean/mode of the training observations in that region



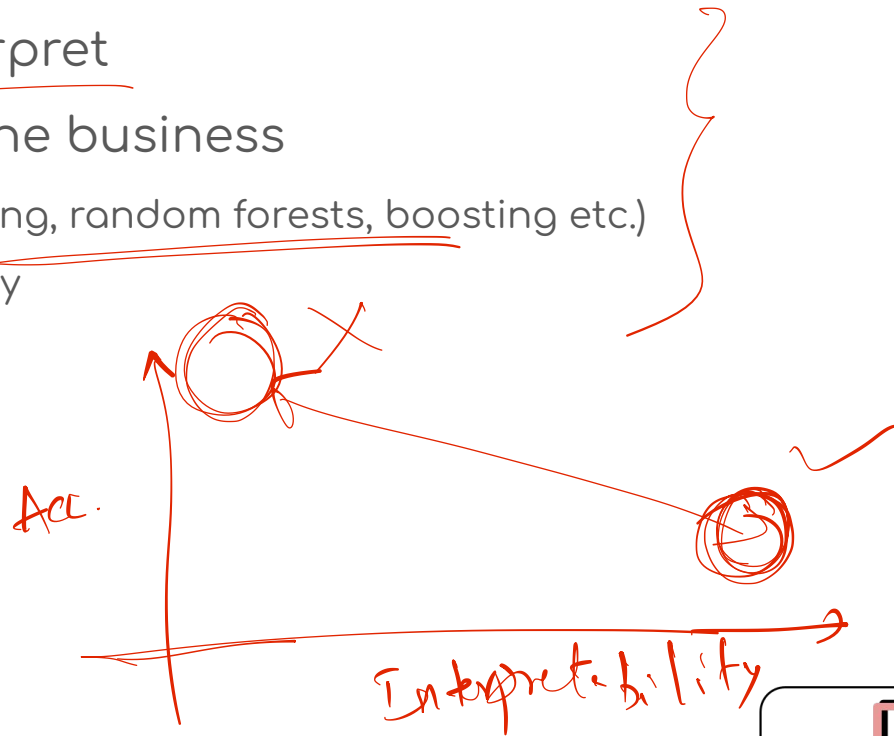
# Tree-based Methods

- Splitting rules used for segmenting can be summarized in a tree →  
Decision Trees



# Tree-based Methods

- Simple and useful to interpret
- Typically not the best in the business
  - Can be improved (e.g. bagging, random forests, boosting etc.)
  - At the cost of interpretability





# Decision Trees for Regression



# Example Problem

- Predicting the baseball players' (log) salary
- Based on the prior experience (years) and hits (in the past year)

# Predicting the Salary

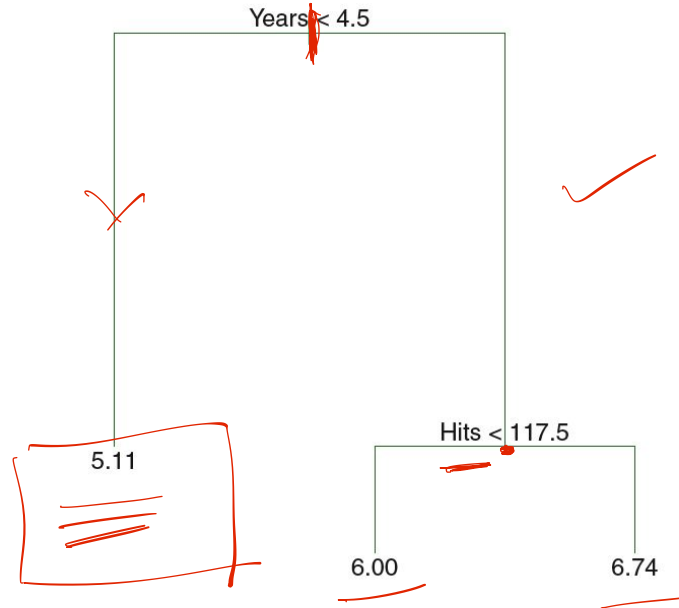
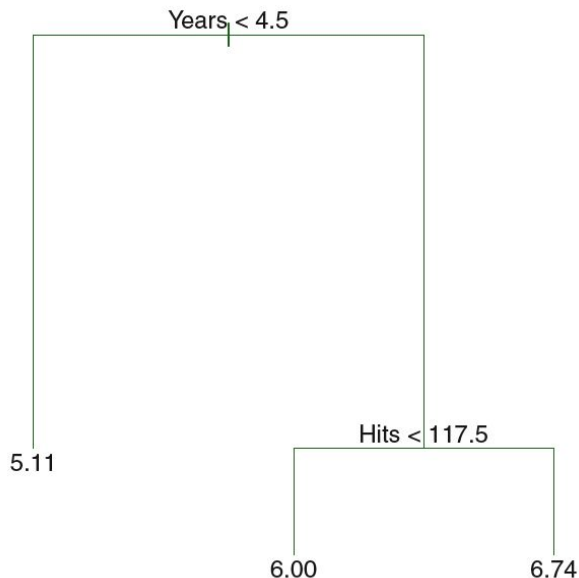


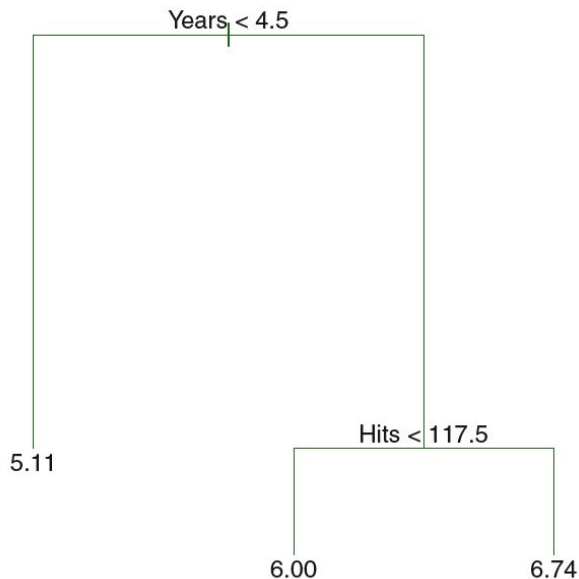
Figure credits: James et al. (ISLR)

# Predicting the Salary



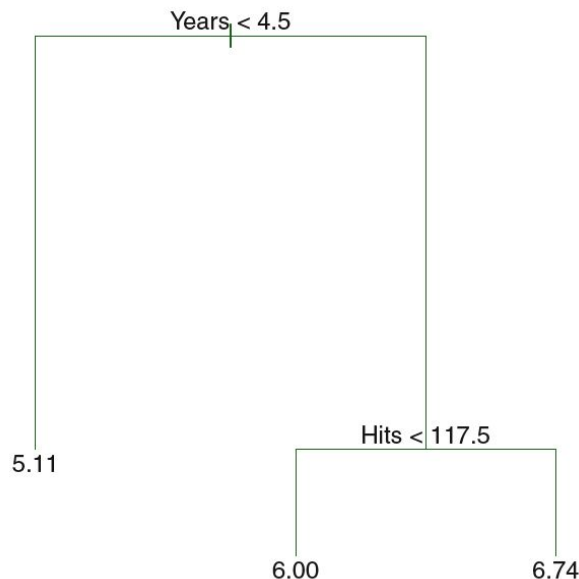
- Top split
  - Based on the experience (less than 4.5 years  $\rightarrow R_1$ )
  - Avg. salary for that split is the mean of the training samples in that region
  - $5.107 \rightarrow e^{5.107}$  thousands of USD

# Predicting the Salary



- Players above 4.5 years of experience → right split
- Further, split based on the hits in the previous year major league
- Less than 117.5 into second region ( $R_2$ ), more than that into third region ( $R_3$ )

# Predicting the Salary



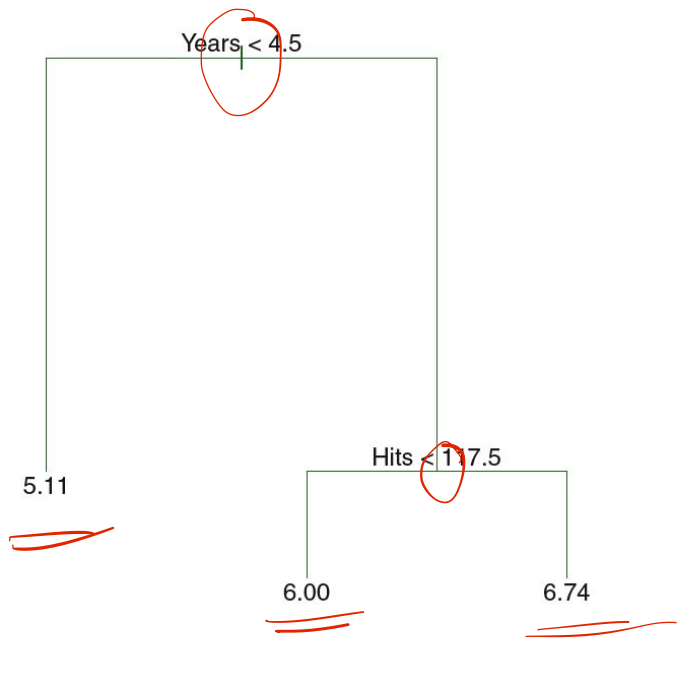
$$R_1 = \{X \mid \text{Years} < 4.5\}$$

$$R_2 = \{X \mid \text{Years} \geq 4.5, \text{Hits} < 117.5\}$$

$$R_3 = \{X \mid \text{Years} \geq 4.5, \text{Hits} \geq 117.5\}$$



# Predicting the Salary



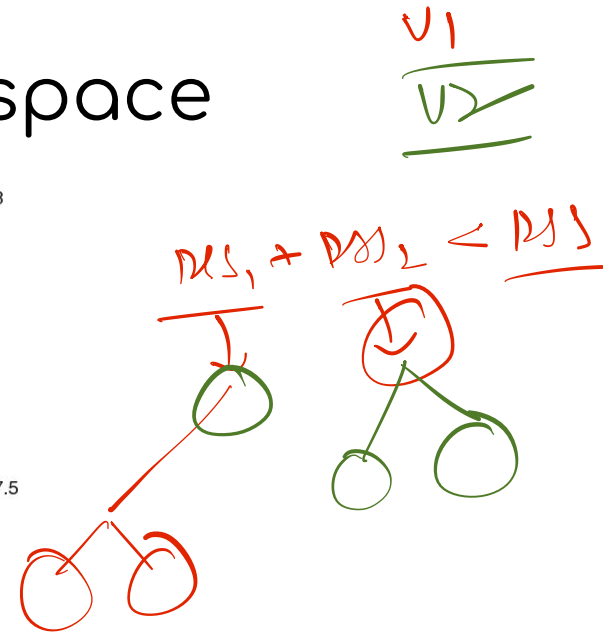
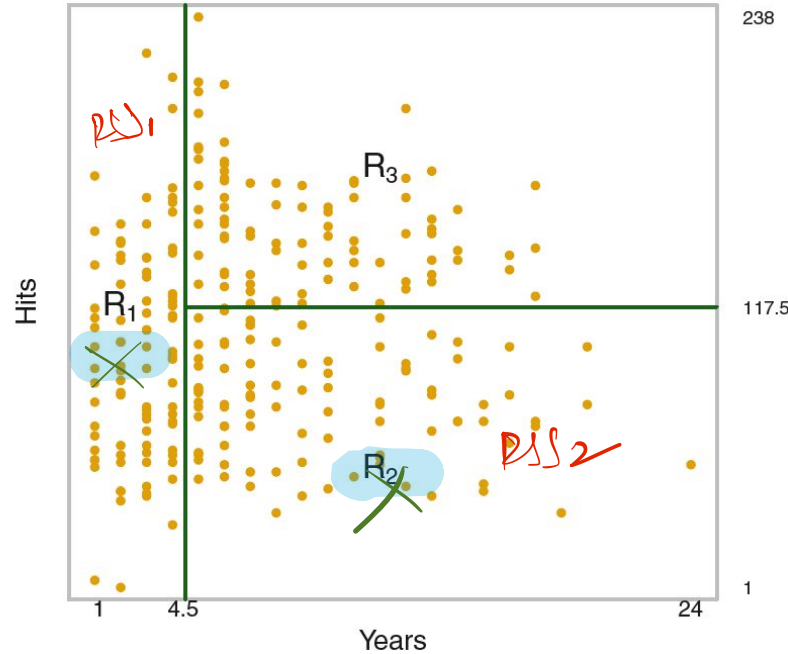
$$R_1 = \{X \mid \text{Years} < 4.5\}$$

$$R_2 = \{X \mid \text{Years} \geq 4.5, \text{Hits} < 117.5\}$$

$$R_3 = \{X \mid \text{Years} \geq 4.5, \text{Hits} \geq 117.5\}$$

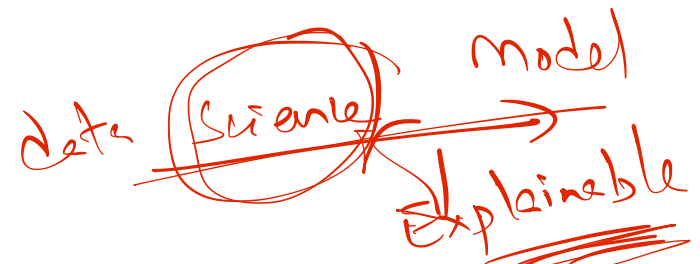
Called the 'terminal' nodes or the 'leaves' of the tree. Others where the predictor space is split is called 'internal' nodes.

# The partitions in the predictor space





# Interpreting the Tree



1. Experience is the most important factor that determines the salary
  - Players with less experience earn less
2. Given that a player is less experienced, the number of hits he made in the last year play little role in the salary
3. For the experienced players, number of hits made recently affect their salary. More hits → more salary

# Interpreting the Tree

- Probably an over-simplification of the true relation b/w {Year, Hits} and Salary
- However the advantage is that it is easier to interpret and has a nice graphical representation



# Stratification of the Feature Space

$$x \in \mathbb{R}^p$$

## Building a Regression Tree

1. Divide the predictor space (i.e set of possible values for  $X_1, X_2, \dots, X_p$ ) into  $J$  distinct and non-overlapping regions ( $R_1, R_2, \dots, R_J$ )

# Stratification of the Feature Space

## Building a Regression Tree

2. For every  $R_j$ , make the same prediction which is the mean of the response value for training samples in  $R_j$

$$\hat{y}_{R_j} = \frac{\text{mean of } y_i \text{ s.t. } x_i \in R_j}{\text{mode}}$$

# Constructing the Regions $R_j$

- Could have any shape. But for simplicity we choose high-dim rectangles



# Constructing the Regions $R_j$

- Could have any shape. But for simplicity we choose high-dim rectangles
- The goal is to find boxes  $R_1, R_2, \dots, R_J$  that minimizes the RSS

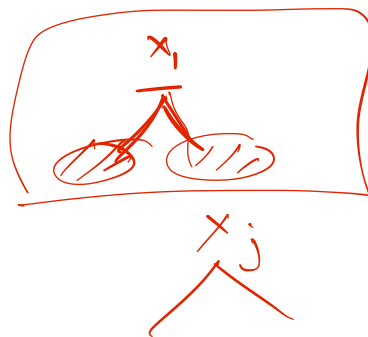
$$\sum_{j=1}^J \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

mean response for the training observations within the  $j^{\text{th}}$  box



# Constructing the Regions $R_j$

- Infeasible to consider every possible partition
- Instead take a top-down, greedy approach → recursive binary splitting



# Constructing the Regions $R_j$

'P'

- Top-down: starts at the top of the tree and recursively splits the predictor space
- Greedy: at each step, best split is made at that particular step
  - rather than looking ahead and picking a split that will lead to a better tree later

i from  
1 to j

$(x_i, s)$





# Constructing the Regions $R_j$

## Recursive Binary Splitting

- First select the predictor  $X_j$ , and then the cutpoint  $s \rightarrow$  leads to a greatest reduction in RSS



# Constructing the Regions $R_j$

## Recursive Binary Splitting

$$R_1(j, s) = \{X | X_j < s\} \text{ and } R_2(j, s) = \{X | X_j \geq s\}$$

seek the value of  $j$  and  $s$  that minimize the equation

$$\sum_{i: x_i \in R_1(j, s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i: x_i \in R_2(j, s)} (y_i - \hat{y}_{R_2})^2$$

285 = 200

179

$(x_1, s)$   
 $(x_2, s)$



# Constructing the Regions $R_j$

## Recursive Binary Splitting

- Next we repeat the process: look for the best predictor and best cutpoint that minimizes the RSS further
- But this time we split one of the two previously identified regions

# Constructing the Regions $R_j$

## Recursive Binary Splitting

- Continue until a stopping criterion is reached
  - E.g., until no region contains more than five observations

# Constructing the Regions $R_j$

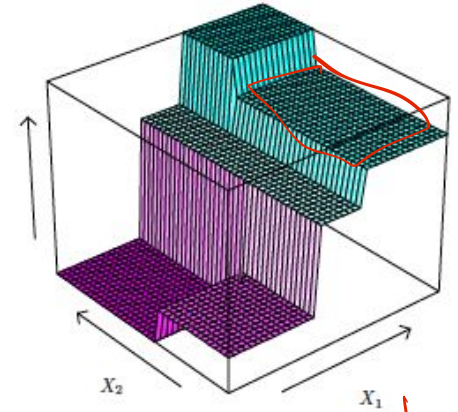
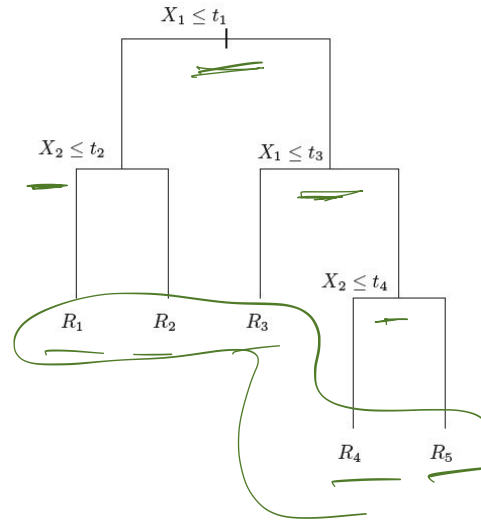
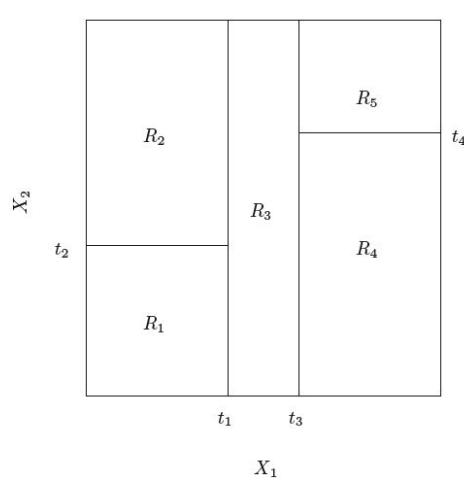
## Recursive Binary Splitting

- Once the regions are identified, prediction is the mean response of the training samples in that region

# Constructing the Regions $R_j$

Recursive Binary Splitting (a 5 region example)

$$(x_j, s_j)$$



piece-wise linear  
predictions



# Overfitting

- Above procedure may give good predictions on training data
  - But likely to overfit
- This is because the resulting tree may be too complex



# Overfitting

- Smaller tree with fewer splits might lead to lesser variance and better interpretation
  - At the cost of a little bias





# Overfitting

- One way to achieve this
  - build the tree only when the decrease in the RSS due to each split exceeds some (high) threshold
    - Results in smaller trees, but is short-sighted

# Tree Pruning

- Grow a large tree, then prune it back to obtain a subtree
- How to find the best subtree?
  - Intuitively, pick the one with min. test/validation error



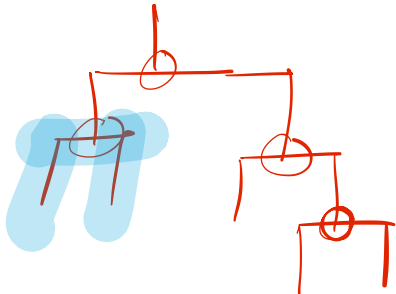
# Tree Pruning

- Estimating the cross-validation error for every possible subtree is cumbersome (large number of subtrees are possible)



# Tree Pruning - Cost Complexity Pruning

- Also known as weakest link pruning
- Rather than considering every possible subtree, consider a sequence of subtrees indexed by  $\alpha$
- For each value of  $\alpha$ ,  $\exists$  a subtree  $T \subset T_0$  s.t. the equation is minimum



=

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

$\alpha = 0.1$

$\alpha = 10$

$T_1 \subset T_0$   $\alpha_1$   
 $T_2 \subset T_0$   $\alpha_2$



# Tree Pruning - Cost Complexity Pruning

- As we increase  $\alpha$ , branches get pruned in a nested fashion
- We can select the value of  $\alpha$  from cross-validation

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

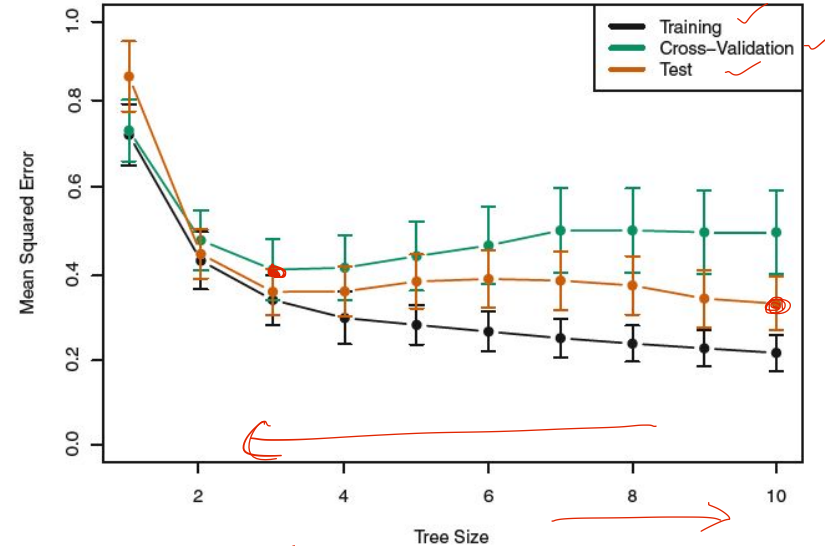
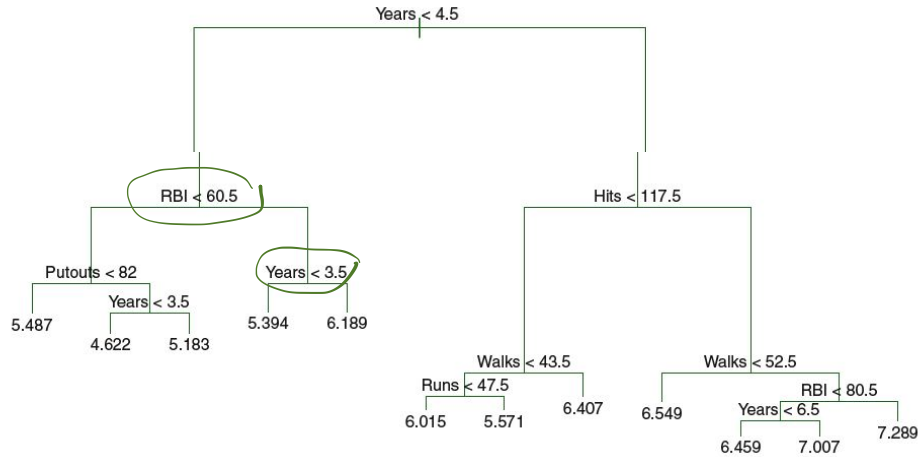


# Tree Pruning - Cost Complexity Pruning

1. Grow large tree using recursive binary splitting
2. Apply cost complexity pruning  $\rightarrow$  obtain a sequence of subtrees as a function of  $\alpha$
3. Compute the validation (or cross validation) performance and pick the best  $\alpha$  that minimizes the error
4. Return the subtree from step 2 that corresponds to the chosen  $\alpha$



# Baseball Salaries Example



# Classification Trees





# Trees for Classification

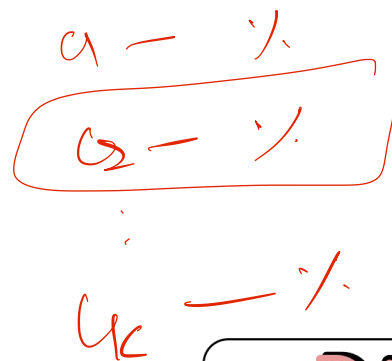
- Similar to the Regression Trees
- Except, predict a qualitative response



# Prediction in Classification Trees

- The most commonly occurring class of training observations in the region - Majority voting

Mean for Regression



# Prediction in Classification Trees

- Along with the class prediction for a terminal node
  - class proportions within the regions of terminal nodes

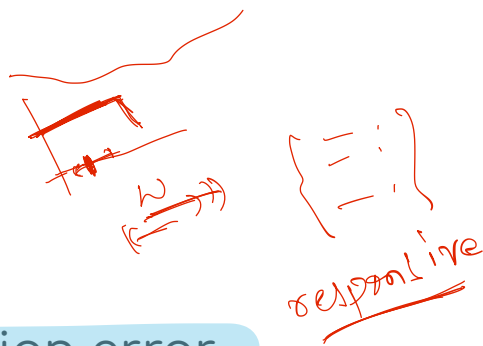


# Growing a Classification Tree

- Recursive binary splitting
- RSS will not do, a natural alternative is classification error
  - Fraction of the training observations in that region ( $R_m$ ) that do not belong to the most common class

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

0.15



$K = \# \text{ classes}$

$P_{m1} \rightarrow 10\%$

$P_{m2} \rightarrow 2\%$

$\vdots$   
 $P_{mk} \rightarrow 85\%$



# Growing a Classification Tree

- Classification error is not very sensitive for tree-growing
  - Two more metrics
- Gini Index and Entropy



# Growing a Classification Tree

- Gini Index → minimizes the total variance across the K classes
  - Referred to as a measure of node purity
    - Small value → node contains predominantly observations from a single class

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

ESS



# Growing a Classification Tree

- Entropy
  - Also serves as a measure of node purity
    - Small value → node contains predominantly observations from a single class

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



# Pruning a Classification Tree

- Any of the three metrics can be used
  - classification error might be preferred if prediction accuracy is the goal



# Heart Disease Example

- Binary outcome (Yes or No)
- 13 predictors: Age, Sex, Chol, heart and lung function measurements etc.



# Heart Disease Example

Thal:st tus  
a  
b  
c

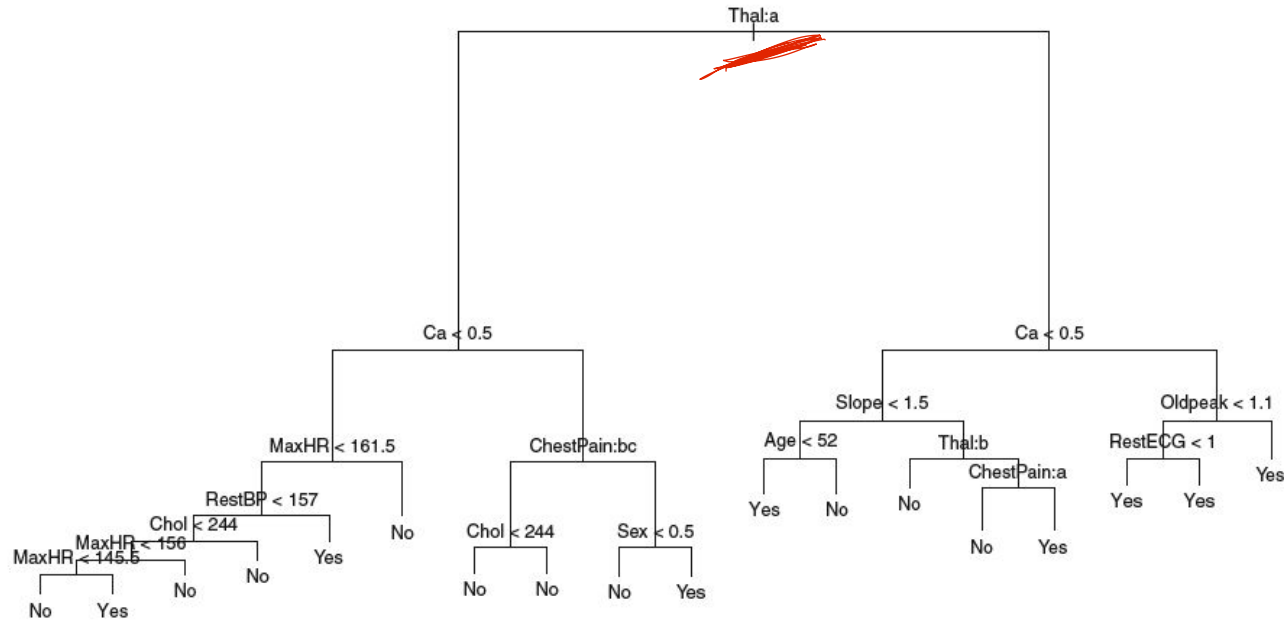
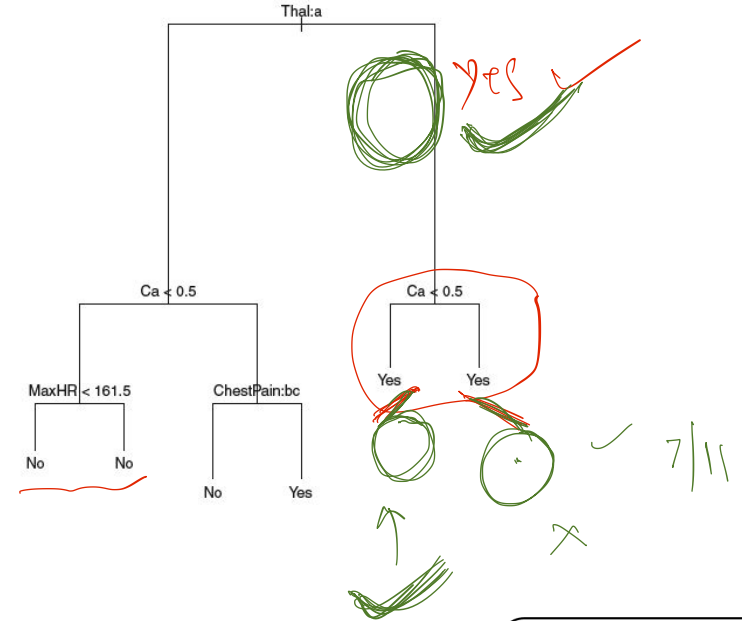
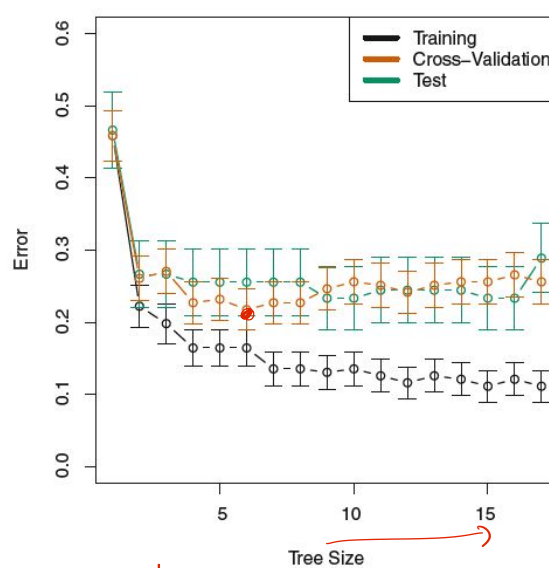


Figure credits: James et al. (ISLR)



# Heart Disease Example



# Some Notes

- Trees can be constructed in the presence of qualitative variable
  - E.g. Sex and Thal variables
- Some of the splits yield two terminal nodes that have the same predicted value
  - RestECG < 1
  - Why? → leads to increased node purity (all 9 of right split observations has a response of yes, whereas 7/11 of left split observations have Yes response)

# Trees vs. Linear Models



# Trees vs. Linear Regression

$$f(X) = \sum_{m=1}^M c_m \cdot \mathbf{1}_{(X \in R_m)}$$

$$f(X) = \beta_0 + \sum_{j=1}^p X_j \beta_j$$



# Trees vs. Linear Regression

- Which model is better? ✓
  - Depends on the problem at hand
- If the relationship between the features and response is well approximated by the linear model
  - LR is likely to work better (RT does not exploit the linear structure)
- If there is a highly nonlinear and complex relationship
  - Decision Trees may outperform the classical methods



# Trees vs. Linear Regression

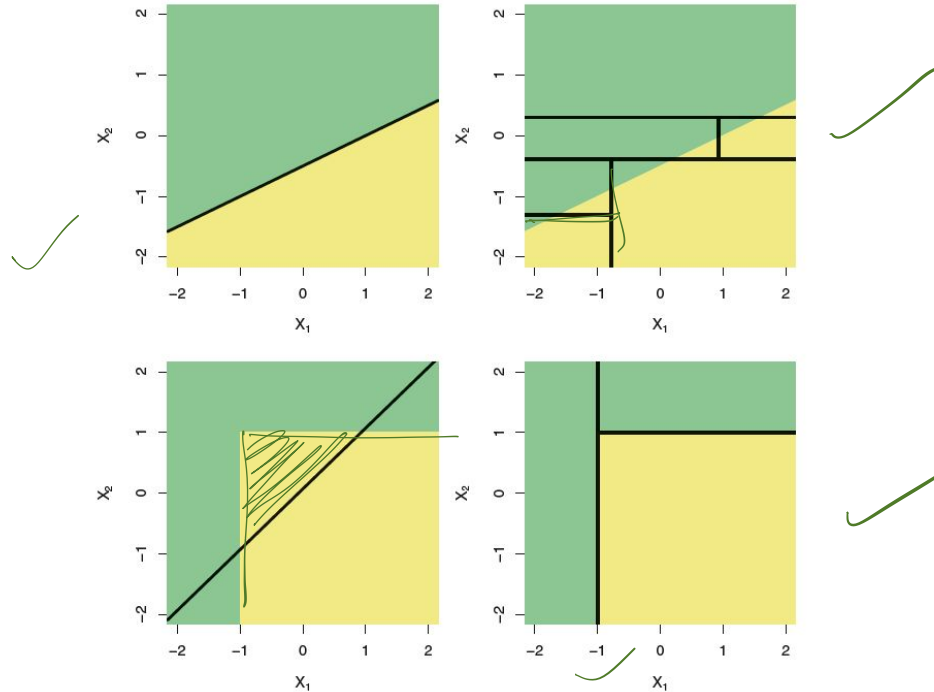


Figure credits: James et al. (ISLR)



# Advantages of Trees

- Very easy to explain to people
  - Some believe that they mirror human decision-making
- Can be displayed graphically (even to a non-expert)
- Can handle qualitative variables
  - Without the necessity of dummy variable

# Disadvantages of Trees

- Generally do not have the same level of predictive accuracy than some of the other techniques
- Can be non-robust
  - Small change in data may cause a large change in the final estimated tree

# Next: More powerful prediction models

- Model combination tools
  - Bagging and Boosting

# Thank You



భారతీయ సాంకేతిక విజ్ఞాన సంస్థ హైదరాబాద్  
भारतीय प्रौद्योगिकी संस्थान हैदराबाद  
Indian Institute of Technology Hyderabad

