CIS 490 Machine Learning

Lecture 10

Instructor: (Julia) Hua Fang

1

Last time

2

- Supervised Learning
  - Regression
    - Linear regression
    - Regularized linear regression: Ridge, Lasso, Cross-validation
  - Classification
    - Logistic regression: Probability, Odds, Log Odds, Logit, logistic function.
    - Classification evaluation in general: Confusion table; classification accuracy (Sensitivity, Specificity, etc), and ROC/AUC
    - R: running logistic regression in R using UCLA admission data

Adapted from Jeff Howbert, Greg Shakhnarovich, Patrick Breheny , M. Magdon-Ismail, Patrick Breheny, Jeff Schneider

2

3

Quiz:

1. If Y is a binary variable, what classification method would you consider?
2. If Y has more than 2 categories, e.g, Y0= no risk; Y1= mild risk; Y3 = high risk, what classification method would you consider?
3. Logistic regression models the logit-transformed probability as a linear relationship with the predictors (T/F)
4. For the S-curve used in logistic regression model (associated with one attribute), the intercept controls slope of rise (T/F)
5. The steeper the S-curve, the better the classifier (T/F)

3

4

Outline

- Supervised Learning
  - **CART**: Classification **a**nd **R**egression Trees (Decision Trees)

Adapted from Jeff Howbert, Greg Shakhnarovich, Patrick Breheny , M. Magdon-Ismail, Patrick Breheny, Jeff Schneider

4

CART

5

CART is **Non-linear**

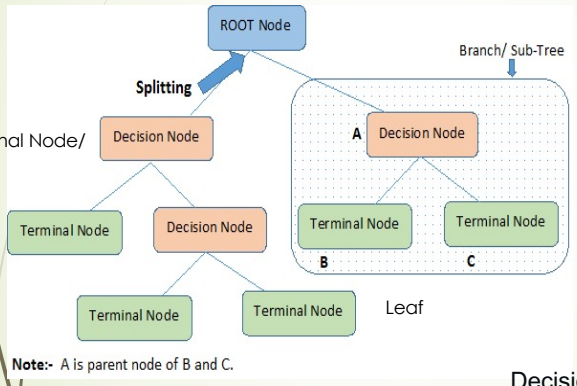
- Regression Trees: You are expected to know
  - When to use
  - Tree building process: Top-down and greedy approach (or recursive binary splitting)
  - Pruning: Algorithm, **Cross validation** for tuning parameter

  Note: for regression, check SSR/SSE/RSS, MSE or RMSE
- Classification trees: You are expected to know
  - When to use
  - Tree building process: Top-down and greedy approach (or recursive binary splitting)
  - Classification error, Gini, Deviance.

5

Decision Trees Terminology

6



Decision trees are typically drawn upside down such that leaves are the the bottom & roots are the tops

6

7

CART I : Regression Trees

7

8

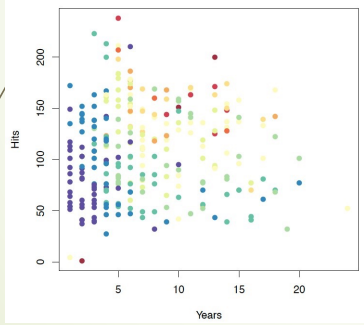
CART I: Regression Tree Example-Baseball Player salary data

- Simpler scenario from this entire data:

  Predicting Y, the **log** salary of a baseball player, based on

  X1: the number of years that he has played in the major leagues

  X2: the number of hits that he made in the previous year.



Y: **Log** Salary is color-coded from low (purple, blue, green) to high (yellow, red)

https://cran.r-project.org/web/packages/ISLR/ISLR.pdf

8

Suppl.: Description of Baseball Player Salary Data

9

A data frame with 263 observations of major league players on the following 20 variables.

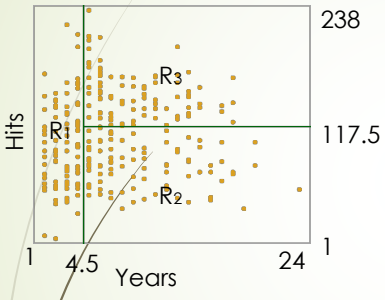
1. AtBat Number of times at bat in 1986
2. Hits Number of hits in 1986
3. HmRun Number of home runs in 1986
4. Runs Number of runs in 1986
5. RBI Number of runs batted in 1986
6. Walks Number of walks in 1986
7. Years Number of years in the major leagues
8. CAtBat Number of times at bat during his career
9. CHits Number of hits during his career
10. CHmRun Number of home runs during his career
11. CRuns Number of runs during his career
12. CRBI Number of runs batted in during his career
13. CWalks Number of walks during his career
14. League A factor with levels A and N indicating player's league at the end of 1986
15. Division A factor with levels E and W indicating player's division at the end of 1986
16. PutOuts Number of put outs in 1986
17. Assists Number of assists in 1986
18. Errors Number of errors in 1986
19. Salary 1987 annual salary on opening day in thousands of dollars (LogSalary: Log salary)
20. NewLeague A factor with levels A and N indicating player's league at the beginning of 1987

https://cran.r-project.org/web/packages/ISLR/ISLR.pdf

9

CART I: Regression Tree -- Goal

10



- The Goal is to find boxes or regions, $R_1,\ldots,R_J$ that minimize the **SSR/SSE/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 j^{th} box or region.

Note: In theory, the regions (R) 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.

10

11

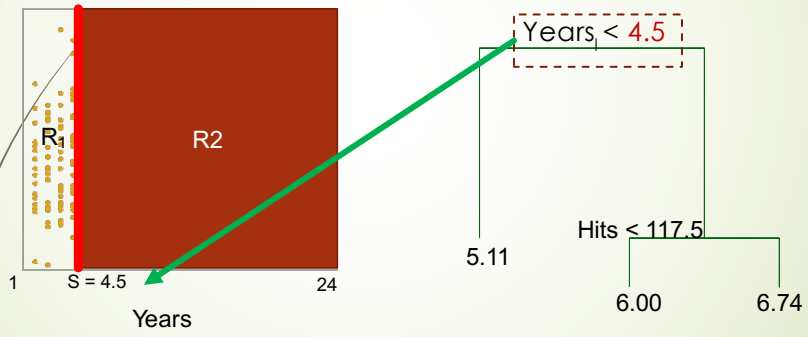
CART I : Regression Trees Building Process

11

12

Regression Tree building process: Baseball Player Salary Example continued

- First, select the predictor X_j and its cut point (splitter or differentiator) ***s,*** such that splitting the predictor space into two boxes or regions $\{X \mid X_j < \mathbf{s}\}$ and $\{X \mid X_j \geq \mathbf{s}\}$ leads to the greatest possible reduction in SSR/SSE/RSS.



Baseball Player Salary Example

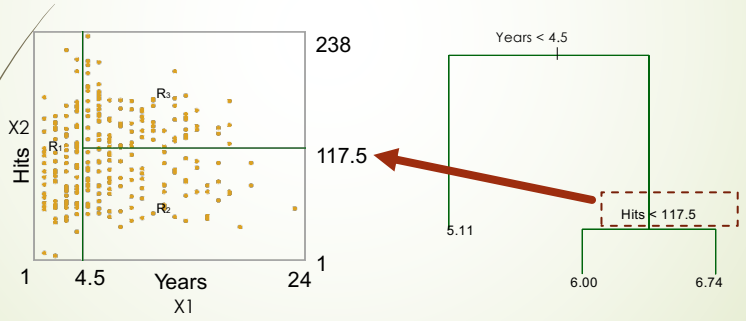
12

Regression Tree building process: Baseball Player Salary Example continued

13

- Next, repeat the process, looking for the next best predictor and best cutpoint in order to split the data further so as to minimize the SSR/SSE/RSS within each of the resulting regions.

-- this time, instead of splitting the entire predictor space (i.e., not from the root node again), we split one of the two previously identified regions. We now have three regions. e.g.,



Baseball Player Salary Example

13

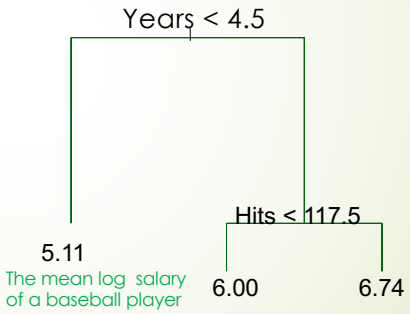
Regression Tree building process: Baseball Player Salary Example continued

14

- Finally, we look to split one of these three regions further, so as to minimize the SSR/SSE/RSS. The process continues until a stopping criterion is reached,

e.g, we may continue until no region contains more than five observations.

Note: We predict the response for a given test observation using the mean of the training observations in the region to which that test observation belongs.



Recall: Baseball Player Salary Example

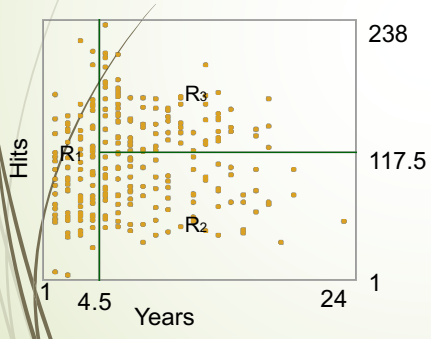
14

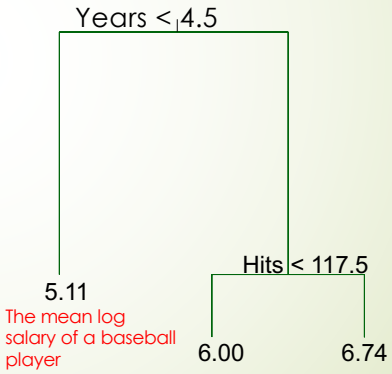
15

Summary: Regression Decision tree for Baseball Player Salary Example

Overall, the tree stratifies or segments the players into three regions of predictor space:

$R_1 = \{X \mid \text{Years} < 4.5\}$,
$R_2 = \{X \mid \text{Years} >= 4.5, \text{Hits} < 117.5\}$, and
$R_3 = \{X \mid \text{Years} >= 4.5, \text{Hits} >= 117.5\}$.

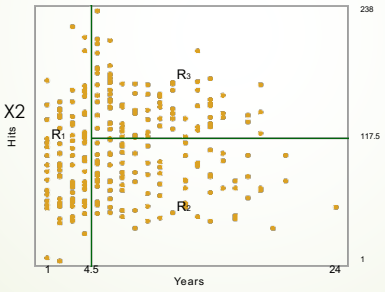




15

16

Question: Is it computationally feasible to consider every possible partition of the feature/predictor space into J boxes?



16

17

Answer:

- Computationally infeasible to consider every possible partition of the feature/predictor space into ***J*** boxes.
  - For this reason, we take a top-down, greedy approach that is known as recursive binary splitting.

17

18

CART 1 Regression Trees:
A top-down and greedy approach
(also known as recursive binary splitting)

18

19

CART 1 Regression Tree: Top-down and greedy approach

- CART takes a top-down and greedy approach that is known as recursive binary splitting.
  - The approach is *top-down* because it begins at the top of the tree and then successively splits the predictor space; each split is indicated via two new branches further down on the tree.

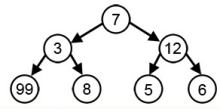
Downloaded from https://j.gifs.com/KzQbjl.gif

19

20

CART 1 Regression Tree: Top-down and greedy approach

- CART takes a top-down and greedy approach that is known as recursive binary splitting.
  - It is *greedy* because at each step of the tree-building process, the *best* split is made at that particular step, rather than looking ahead and picking a split that will lead to a better tree in some future step



Downloaded from https://en.wikipedia.org/wiki/Greedy_algorithm

20

21

CART I Regression Trees: Pruning, algorithm, cross-validation

21

22

CART 1 Regression Tree: Pruning a tree

The tree building process described above may produce good predictions on the training set, but is likely to overfit the data, leading to poor test set performance.

A better strategy is to grow a very large tree and then prune it back in order to obtain a subtree

- Technically, it is called **cost complexity pruning**, also known as **weakest link Pruning**
- Simply, pruning is the process where we remove sub-nodes of a decision/internal node. You can say pruning is the opposite process of splitting.

22

23

CART 1 Regression Tree: Pruning a tree

We consider a sequence of trees indexed by a nonnegative **tuning parameter** α

For each value of α , there corresponds a subtree $T \subset T_0$ such that

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

is as small as possible.

Where ***T*** indicates the number of terminal nodes of the tree T_0; R_m is the rectangle (i.e., box or region, the subset of predictor space) corresponding to the ***m*th** terminal node (leave), and $\hat{y}_{R_m}$ is the mean of the training observations in R_m.

23

24

Pruning a tree: Choosing the best subtree

- The tuning/penalty parameter α controls a trade-off between the subtree's complexity and its fit to the training data. (*Recall variance-bias trade-off*)
- How could we select an optimal value of α

  (hint: recall lasso & ridge; how do we find the optimal tuning parameter)
- We then return to the full data set and obtain the subtree corresponding to the optimal α.

24

CART 1 Regression tree Pruning algorithm

25

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 (stopping criterion).

2. Apply **cost complexity pruning** to the large tree in order to obtain a sequence of best subtrees, as a function of $\boldsymbol{\alpha}$.

3. Use K-fold cross-validation to choose $\boldsymbol{\alpha}$. For each $k = 1, \ldots, K$:
   - 3.1 Repeat Steps 1 and 2 on the (K-1)/K th fraction of the training data, excluding the kth fold.
   - 3.2 Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of $\boldsymbol{\alpha}$. Average the results, and pick $\boldsymbol{\alpha}$ to minimize the average error.

4. Return the subtree from Step 2 that corresponds to the chosen value of $\boldsymbol{\alpha}$ *(ie. The optimal* $\boldsymbol{\alpha}$*)*.

25

CART 1 Regression tree Pruning algorithm: Full Baseball player salary data

26

- First, randomly divide the data set in half, yielding 132 observations in the training set and 131 observations in the test set.
- Then build a large regression tree on the training data and varied $\boldsymbol{\alpha}$ in order to create subtrees with different numbers of terminal nodes.
- Finally, perform six-fold cross-validation in order to estimate the cross-validated MSE of the trees as a function of $\boldsymbol{\alpha}$.

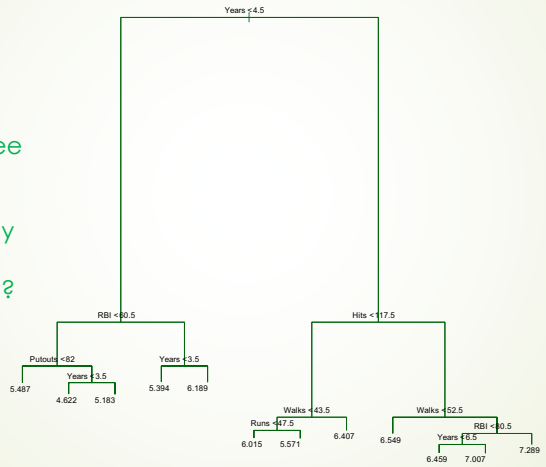
26

27

CART 1 Regression tree Pruning algorithm: Full Baseball player salary data

What's the tree size?

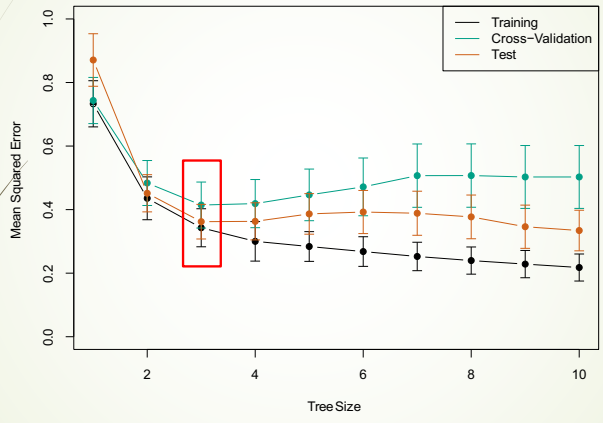
i.e., how many terminal nodes/leaves?



27

28

CART 1 Regression tree Pruning algorithm: Full Baseball player salary data

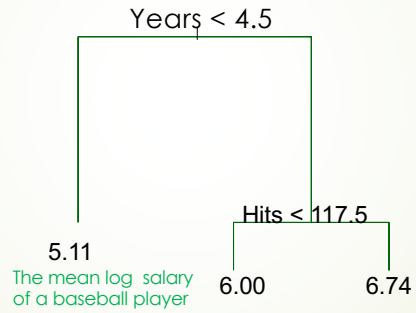


28

29

Baseball player salary example continued

What's the tree size?



29

30

CART 2: Classification Trees

30

31

Classification Trees

- Very similar to a regression tree, except that it is used to predict a discrete/categorical/qualitative response rather than a continuous/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.

31

32

CART 2 Building classification trees

- Just as in the regression trees setting, we use recursive binary splitting to grow a classification tree. (No repeat here)

Note: In the classification setting, SSR/SSE/RSS cannot be used as a criterion for making the binary splits

32

33

CART 2 Building Classification trees

- A natural alternative to SSR/SSE/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 m^{th} region that are from the k^{th} class.

Note: classification error is not sufficiently sensitive for tree-growing, and in practice two other measures are preferable. (see next slides)

33

34

CART 2 Building classification trees: Gini index and Deviance

- The *Gini index (G)* is defined by

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

Looks familiar? Hint: Variance of which discrete probability distribution?

It is a measure of total variance across the K classes; referred to as a measure of node *purity:*

That is, a small value indicates that a node contains predominantly observations from a single class.

we select the split with the smallest Gini index

34

35

CART 2 Building classification trees: Gini index and Deviance

- An alternative to the Gini index is deviance (D) (or Information gain or cross-entropy), given by

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

The best first split is the one that provides the most information gain. This process is repeated for each impure node until the tree is complete.

Note: Like the Gini index, the deviance will take on a small value if the m^{th} node is pure. (the smaller the better)

35

36

CART 2 **Classification trees**: Heart disease data example

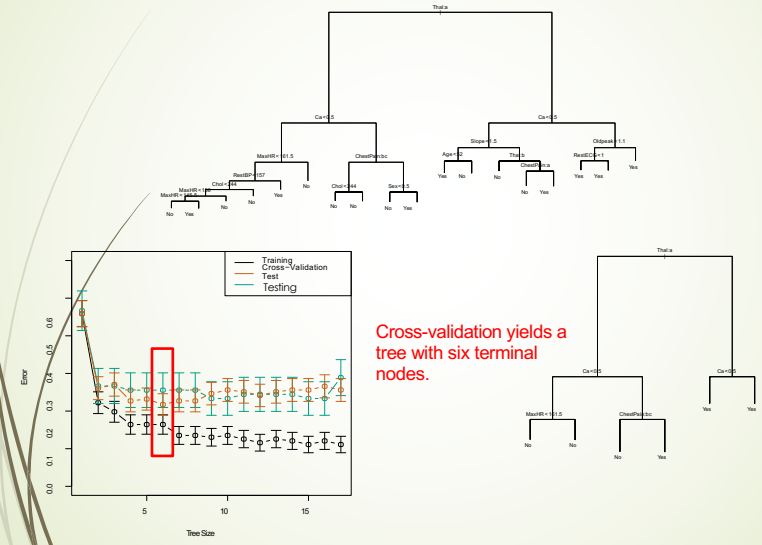
- A binary outcome HD for 303 patients who presented with chest pain.
- An outcome value of "Yes" indicates the presence of heart disease based on an angiographic test, while "No" means no heart disease.
- 13 predictors including Age, Sex, Chol (a cholesterol measurement), and other heart and lung function measurements.

See next figure.
https://www.statlearning.com/s/Heart.csv

36

37

CART 2 Classification Tree: Heart Disease Data



Cross-validation yields a tree with six terminal nodes.

37

38

Summary: Why using CART or Decision Trees?

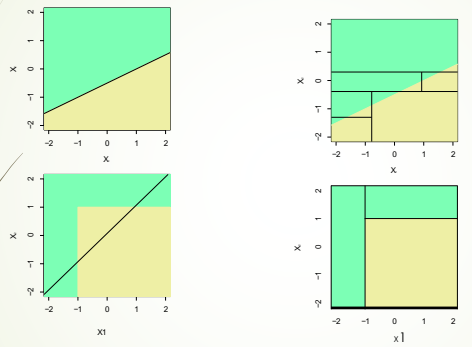
"If I can use logistic regression for classification problems and linear regression for regression problems, why is there a need to use trees"?

- If the relationship between dependent & independent variable is well approximated by a linear model, linear regression will outperform tree based model.
- If there is a high non-linearity & complex relationship between dependent & independent variables, a tree model will outperform a classical regression method.
- If you need to build a model which is easy to explain to people, a decision tree model will always do better than a linear model. Decision tree models are even simpler to interpret than linear regression!

38

39

CART Trees vs. Linear Models



Top Row: True linear boundary; Bottom row: true non-linear boundary.
Left column: linear model; Right column: tree-based model

39

40

Summary: Advantages and Disadvantages of CART or Decision Trees

- ▲ Trees are very easy to explain to people. Displayed graphically, and are easily interpreted even by a non-expert (especially if they are small).
- ▲ Some people believe that decision trees more closely mirror human decision-making
- ▲ Easily handle discrete predictors without the need to create dummy variables.
- ▼ Unfortunately, trees generally do not have the same level of predictive accuracy as some of the other regression and classification approaches.

40

41

Suppls: Remedies (Reading assignments: Slides and instructional file posted)

However, by aggregating many decision trees, the predictive performance of trees can be substantially improved. These approaches are:

-- Bagging
-- Random Forest
-- Boosting

41

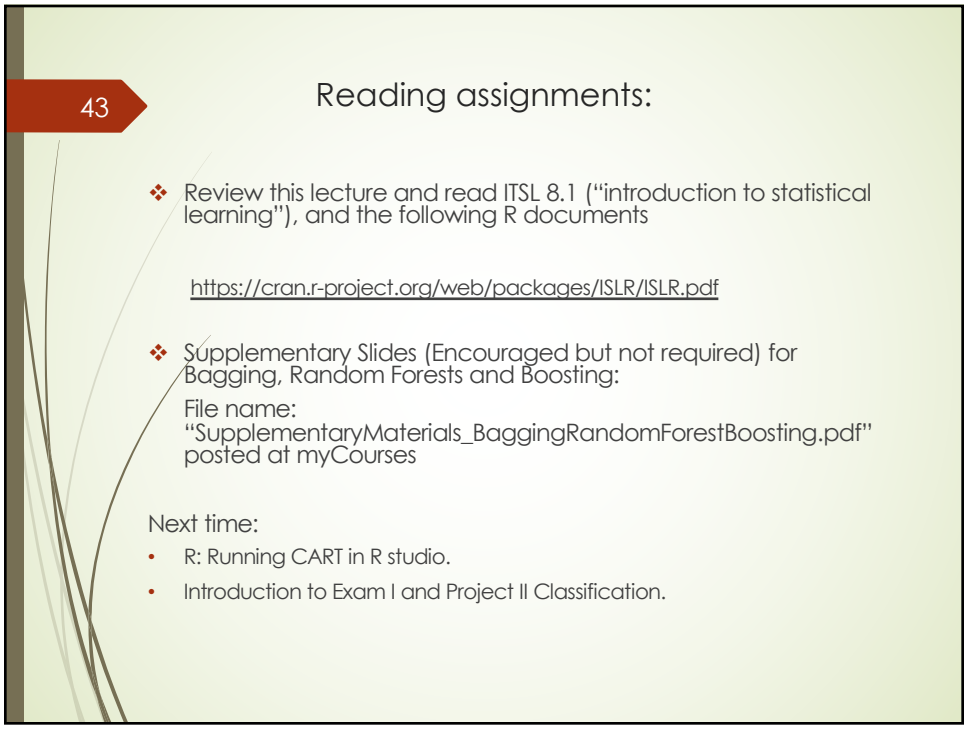
42

Suppls: CART--Bagging, Random Forests and Boosting

Bagging, random forests (RF) and boosting are good methods for improving the prediction accuracy of trees. They work by growing many trees on the training data and then combining the predictions of the resulting ensemble of trees.

- Bagging and RF use bootstrap sampling while boosting does not.
- Bagging and RF use out of bag (OOB) error estimation to estimate test error and does not need cross-validation or validation set approach, while Boosting uses cross-validation to select the number of trees.
- Bagging uses all predictors to estimate a split while random forests use the square root of the number of predictors to estimate a split; unlike bagging and RF, Boosting grows trees sequentially.
- RF provides an improvement over bagged trees by decorrelating the trees. This reduces the variance when we average the trees.
- random forests and boosting, are among the state-of-the-art methods for supervised learning. However their results can be difficult to interpret.

42


43
Reading assignments:
Review this lecture and read ITSL 8.1 ("introduction to statistical learning"), and the following R documents
https://cran.r-project.org/web/packages/ISLR/ISLR.pdf
Supplementary Slides (Encouraged but not required) for Bagging, Random Forests and Boosting:
File name:
"SupplementaryMaterials_BaggingRandomForestBoosting.pdf"
posted at myCourses
Next time:
R: Running CART in R studio.
Introduction to Exam I and Project II Classification.

43