MATH 4322 - Lecture 17

Tree Based Methods: Classification Tree Example and Boosting

Dr. Cathy Poliak, cpoliak@uh.edu

University of Houston

Tree Based Models

So far we have covered such parametric models as:

- Linear Regression
- Logistic Regression
- Linear Discriminant Analysis

and such re-sampling techniques as

- Cross-Validation
- Bootstrap

we are ready for another model class:

- **Tree-based models**.

Tree-based models can be applied to **both** regression and classification problems.

Recall Building a Tree

Below are the steps of building a decision tree (ISLR, page 309):

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.
2. Apply cost complexity pruning to the large tree in order to obtain a sequence of best sub-trees, as a function of α.
3. Use K-fold cross validation to choose α. That is, divide the training observations into K folds. For each $k = 1, \ldots, K$;
   (a) Repeat steps 1 and 2 on all but the kth fold of the training data.
   (b) Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of α.
4. Average the results for each value of α, and pick α to minimize the average error.
5. Return the sub-tree from Step 2 that corresponds to the chosen value of α.

Growing a Classification Tree

- Task of growing a classification tree is similar to the regression trees, we use recursive binary splitting to grow a classification tree.
- Recall criterion for regression tree is **RSS**.

$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \qquad MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Since the response is categorical we cannot calculate the residual standard error. Thus we use other criterion to grow a classification tree.

- Classification error rate - The fraction of the training observations in that region that do not belong to the most common class:

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

- Gini index - A measure of total variance across the K classes.

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

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

What Method To Use

- When building a classification tree, either the Gini index or the entropy are typically used to evaluate the quality of a particular split. The leads to increased **node purity**.
- Any of the three approaches might be used when **pruning** the tree.
- The classification error rate is preferable if prediction accuracy of the final pruned tree is the goal.
- The `tree` function uses the classification error rate as the default but can also use the Gini Index.

Example

- We will use the *Heart* data. See Canvas to get the data.
- This data contains info on patients with chest pains, and we'd like to classify if a patient has a heart disease (*AHD*) depending on multiple factors.
- Import the data into R type and run the following:

```
library(tree)
set.seed(100)
train = sample(1:nrow(Heart),nrow(Heart)/2+0.5)
Heart$AHD = as.factor(Heart$AHD)
Heart$ChestPain = as.factor(Heart$ChestPain)
Heart$Thal = as.factor(Heart$Thal)
Heart$Sex = as.factor(Heart$Sex)
tree.heart = tree(AHD ~ . -X, Heart,subset =  train)
```

Not including

Lab Questions

1. Type and run summary(tree.heart). What are the number of terminal nodes?

   a) 9 b) 10 c) 11 d) 13

2. What is the training error rate?

   a) 38.5% b) 10.96% c) 14% d) 51.6%

```
summary(tree.heart)

Classification tree:
tree(formula = AHD ~ . - X, data = Heart, subset = train)
Variables actually used in tree construction:
[1] "Ca"       "Thal"     "Chol"     "RestBP"   "ChestPain" "Oldpeak"
[7] "Slope"    "Sex"      "Age"
Number of terminal nodes:  11
Residual mean deviance:  0.4439 = 59.92 / 135
Misclassification error rate: 0.1096 = 16 / 146
```

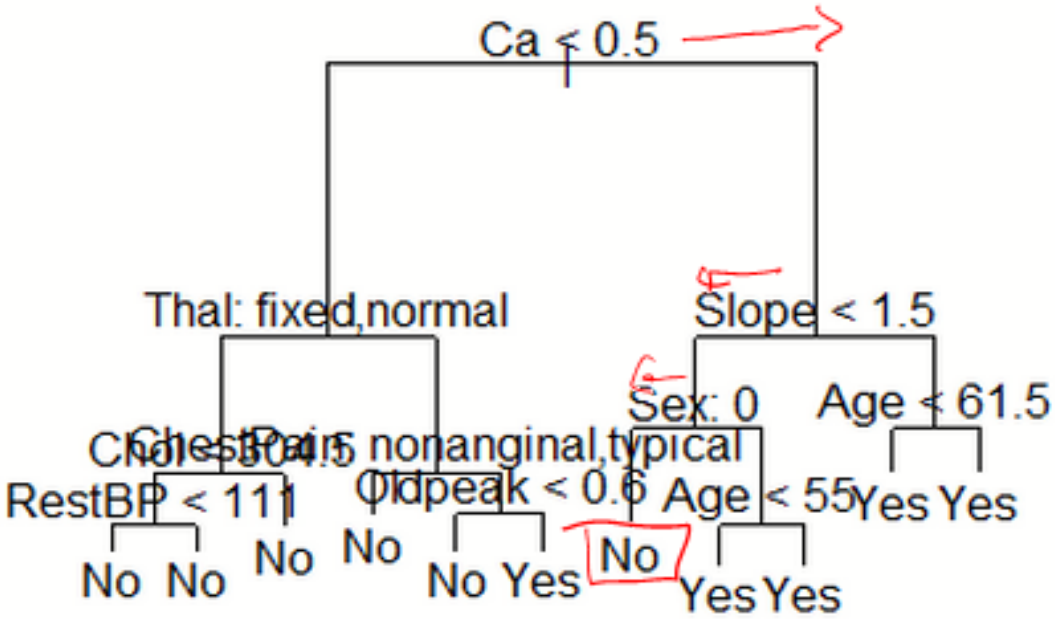
3. Type and run the following in R

```
plot(tree.heart)
text(tree.heart)
```

Is a person predicted to have heart disease if the Ca > 0.5, slope < 1.5 and sex $= 0$

a) Yes

b) No



Test Error Rate

In order to properly evaluate the performance of a classification tree on these data, we must estimate the test error rather than simply computing the training error. Type and run the following in

```
Heart.test = Heart[-train,]
tree.pred = predict(tree.heart,Heart.test,type = "class")
table(tree.pred,Heart.test$AHD)
```

4. What is the test error rate?

a) 18.34%

c) 21%

b) 81.66%

d) 17.5%

$$\frac{(13+18)}{60+18+60+13}$$

tree.pred	No	Yes
No	60	18
Yes	13	60

Pruning

- Next, we consider whether pruning the tree might lead to improved results.
- We use the argument `FUN=prune.misclass` in the `cv.tree()` function in order to indicate that we want the classification error rate to guide the cross-validation and pruning process, rather than the default for the `cv.tree()` function, which is deviance.
- The `cv.tree()` function reports the number of terminal nodes of each tree considered (`size`) as well as the corresponding error rate and the value of the cost-complexity parameter used (`k`, which corresponds to α).

Type and run the following in R

```
set.seed(3)
cv.heart = cv.tree(tree.heart,FUN = prune.misclass)
par(mfrow = c(1,2))
plot(cv.heart$size,cv.heart$dev,type = "b")
plot(cv.heart$k,cv.heart$dev,type = "b")
```

```
par(mfrow = c(1,1))
```

5. How many nodes do we really desire?

a) 14

b) 10

c) 6

d) 2

Pruned Tree

Type and run the following:

```
prune.heart = prune.misclass(tree.heart,best = 6)
tree.pred = predict(prune.heart,Heart.test,type = "class")
table(tree.pred,Heart.test$AHD)
```

What is the test error rate?

tree.pred	No	Yes
No	57	16
Yes	16	62

$$\frac{(16+16)}{(57+16+16+62)} = 21\%$$

Trees vs Linear Models

Linear regression assumes a model of the form

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

Regression trees assume a model of the form

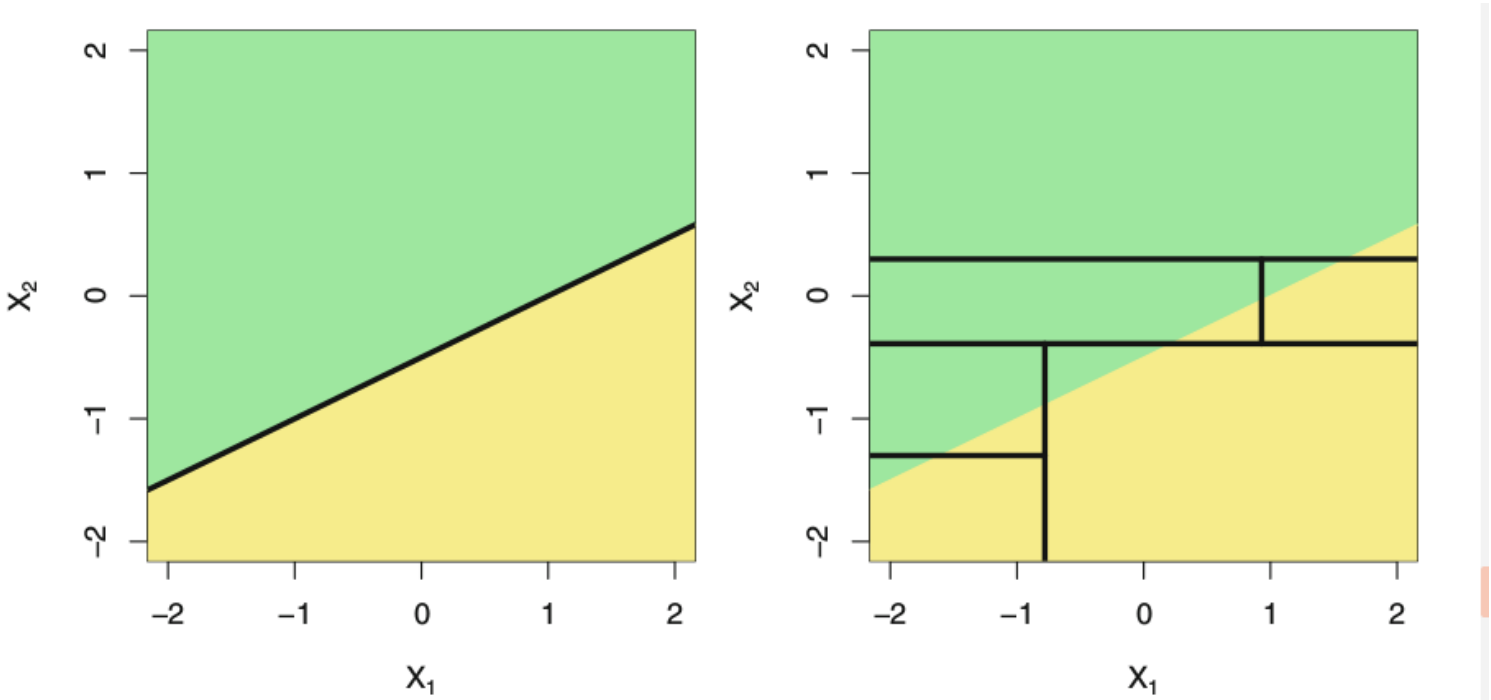
$$f(X) = \sum_{m=1}^{M} c_m \mathbf{1}_{(X \in R_m)}$$

Which is better? Data Scientist answer: it depends on the problem and data at hand,

- If relationship between predictors $X_1, \ldots, X_p$ and Y is approximately linear $\implies$ linear model it is.
- Otherwise, if that relationship is highly non-linear and complex $\implies$ decision trees may have an edge.

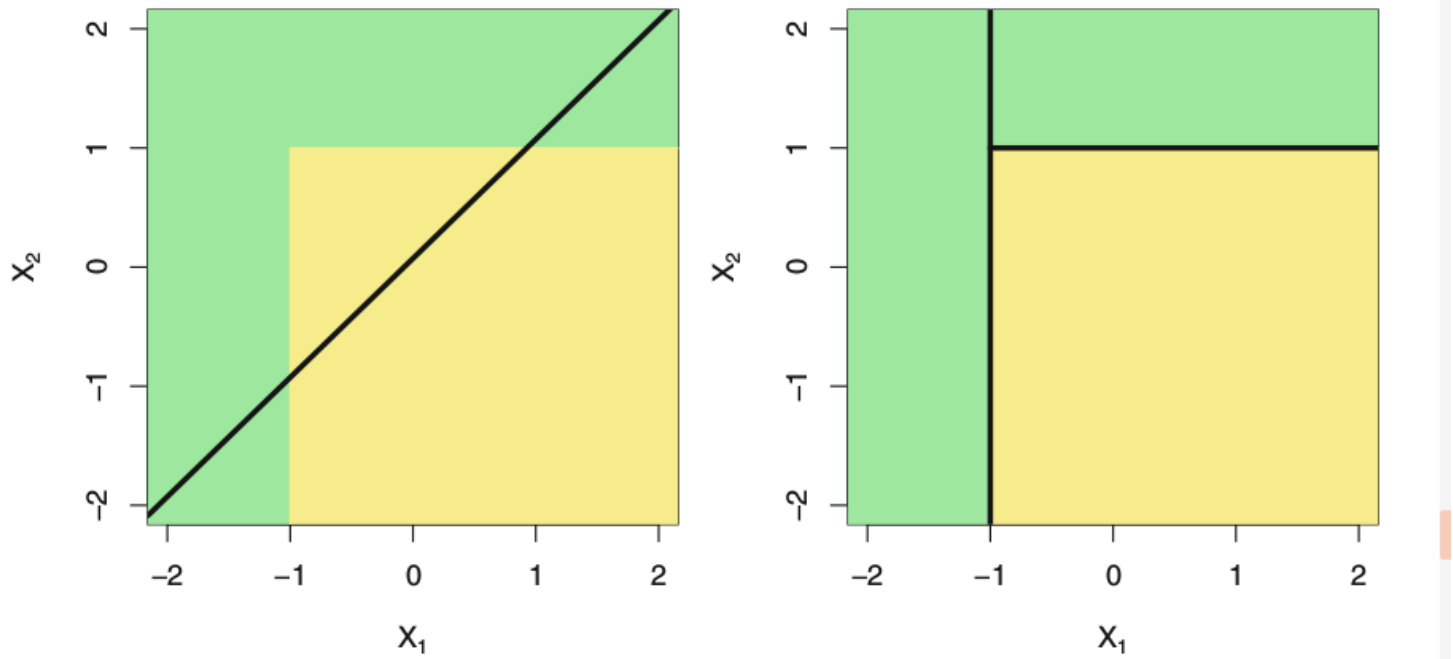
Classification Example: Linear preferred over Trees

Example. In the case of a classification problem with a linear boundary - linear approach is equipped to perform better. Below you see results of fitting a linear model (left) and a decision tree (right).



Classification Example: Trees preferred or Linear

Example. In classification problem with a more complex, non-linear boundary - decision trees have better chances. Below you see results of fitting a linear model (left) and a decision tree (right).



Decision Trees: Advantages and Disadvantages

Several **advantages** of decision trees as a model:

1. Easy to explain, visualize and interpret.
2. More closely mirror human decision-making than regression and certain other methods.
3. Easily handle both quantitative and qualitative predictors (and responses).

The biggest downside:

1. Trees generally do not have the same level of predictive accuracy as some other regression and classification approaches.

However, by aggregating many decision trees (e.g. bagging, random forests), the predictive performance of trees can be vastly improved.

Decision Trees: Issues

Decision trees have the following disadvantages:

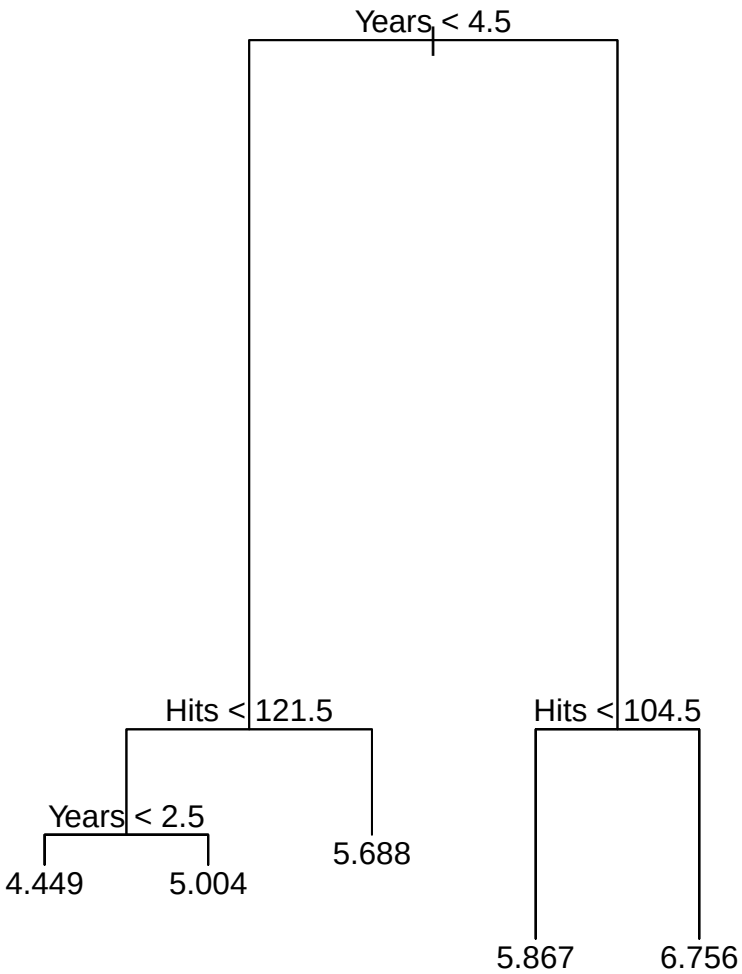
- They struggle with prediction accuracy.
- They suffer from high variance - different subsets of the same data could yield drastically different results.

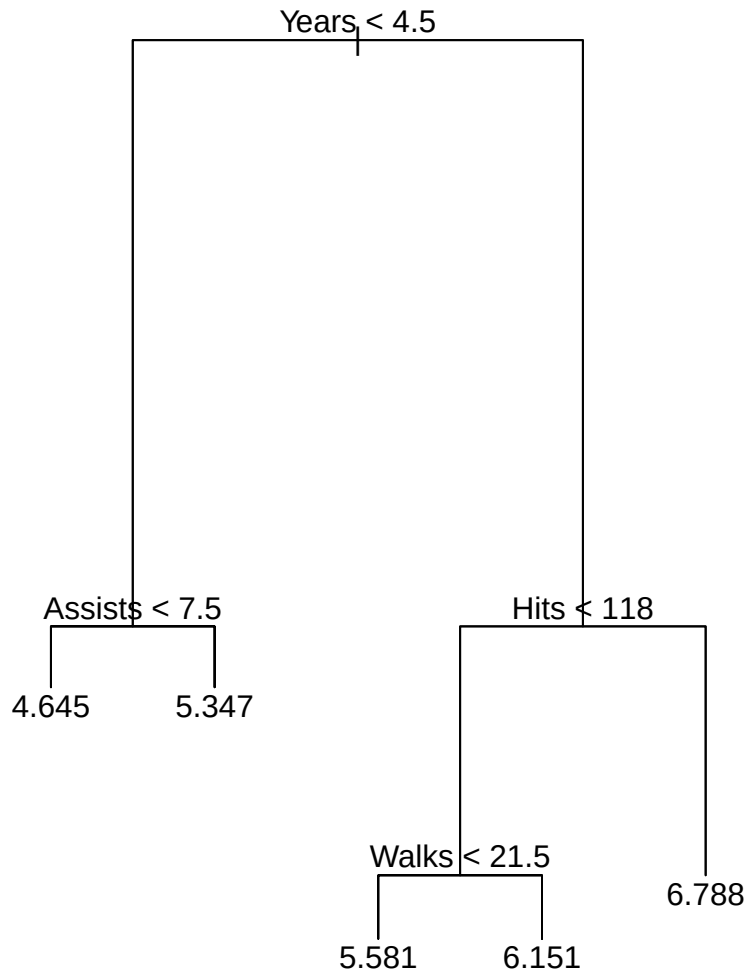
Example Recall the *Hitters* data set. Below we produced pruned regression trees that result from two different training subsets.

```
library(ISLR)
n = nrow(Hitters)
Hitters2 = na.omit(Hitters)
included.vars = names(Hitters[-c(3,8:15,20)])
par(mfrow = c(1,2))
for (i in 1:2) {
  train = sample(1:n, 0.5*n)
  tree.model = tree(log(Salary) ~.,
                    data = Hitters2[,included.vars],
                      subset = train)
      prune.model = prune.tree(tree.model,best = 5)
      plot(prune.model); text(prune.model,pretty = T)
    }
```

```
par(mfrow = c(1,1))
```

Difference in Trees





Compared to Linear Models

For comparison, linear models have **lower variability**

```
for (i in 1:2) {
  train = sample(1:n, .5*n)
    lm.model = lm(log(Salary) ~.,
                  data = Hitters2[,included.vars],
                  subset = train)
    hitters.coef = coef(summary(lm.model))
    print(knitr::kable(hitters.coef[,c(1,4)], escape = FALSE))
    }
```

	Estimate	Pr(>|t|)
(Intercept)	4.3430970	0.0000000
AtBat	-0.0013883	0.4158784
Hits	0.0080924	0.1392025
Runs	-0.0023133	0.6913220
RBI	0.0073918	0.0530580
Walks	0.0080909	0.0333524
Years	0.0935299	0.0000000
PutOuts	0.0001983	0.3390135
Assists	0.0003475	0.5061716
Errors	-0.0155108	0.1205428

	Estimate	Pr(>|t|)
(Intercept)	4.1611344	0.0000000
AtBat	-0.0018668	0.1906423
Hits	0.0112242	0.0162368
Runs	0.0011802	0.8427595
RBI	0.0048541	0.2285454
Walks	0.0060655	0.0958048
Years	0.0965246	0.0000000
PutOuts	-0.0000158	0.9324015
Assists	-0.0002110	0.6902496
Errors	-0.0020626	0.8619627

Recall Bootsrap Method

- Re-sample from the original data - either directly or via a fitted model - to create data sets, from which the variability of the quantities of interest can be assessed with out long-winded and error-prone analytically calculations.
- This approach involves repeating the original data analysis procedure with many replicate sets of data.
- The central goal is to obtain reliable standard errors, confidence intervals, and other measures of uncertainty for a wide range of problems.
- This approach can be applied in simple problems to check the adequacy of standard measures of uncertainty, to relax assumptions, and to give quick approximate solutions.
- The basic idea of bootstrap is to make inference about an estimate (such as the sample mean or sample coefficients $\hat{\beta}_j$) for a population parameter θ (such as the population mean or coefficients β_j) on sample data.

Recall General Approach to Statistical Learning

- Let Y be the response (dependent variable).
- Let $X = (X_1, X_2, \ldots, X_p)$ be p different predictors (independent) variables.
- We assume there is some sort of relationship between X and Y, which can be written in the general form

$$Y = f(X) + \epsilon$$

- Statistical leaning refers to a set of approaches for estimating f.
- A way to reduce the variance and increase the prediction accuracy of a statistical learning method is to take many training sets from the population, build a separate prediction model using each training set and average the resulting predictions.
- That is we could calculate $\hat{f}^1(x), \hat{f}^2(x), \ldots, \hat{f}^B(x)$ using B separate training sets and average them in order to obtain a single low-variance statistical learning model, given by

$$\hat{f}_{avg}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^b(x)$$

Bagging

Since we do not have access to multiple training sets, we can bootstrap, by taking repeated samples from the (single) training data set.

1. We generate B different bootstrapped training data sets.
2. We train our method on the bth bootstrapped training set in order to get $\hat{f}^{*b}(x)$.
3. Average all the predictions to obtain

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

This is called **bagging** - Bootstrap AGGregatING.

Bagging for Decision Trees

- Bagging can improve predictions for many regression methods, it is particularly useful for decision trees.
- To apply bagging to regression trees:
  1. Construct B regression trees using B bootstrapped training sets.
  2. Average the resulting predictions.
- These trees are grown deep, and are not pruned. Thus each individual tree has high variance but low bias.
- Averaging these B trees reduces the variance.
- To apply bagging to classification:
  1. We record the class predicted by each of the B trees
  2. Take a *majority vote*: the overall prediction is the most commonly occurring class among the B predictions.

Bagging Example: Regression Problem

We use a different package called randomForest. Install that package first in R.

```
library(randomForest)
set.seed(100)
train = sample(1:nrow(Hitters2),nrow(Hitters2)/2+0.5)
p = 9 #No. of predictors
B = 100 #No. of bootstrap trees
bag.model.p = randomForest(log(Salary) ~.,
                           Hitters2[,included.vars],
                          subset = train, mtry = p,
                          ntree = B, keep.forest = TRUE)
head(bag.model.p$predicted)
```

```
     -Ron Kittle        -Jose Cruz  -Rance Mulliniks -Andres Galarraga
        5.775968          6.616676          6.220998          4.968826
   -Glenn Wilson  -Argenis Salazar
        6.534321          4.664713
```

Results Checking Mean Square Error (MSE)

```
bag.model.p #To get an outline of bagging results.
```

```
Call:
 randomForest(formula = log(Salary) ~ ., data = Hitters2[, included.vars],      mtry
               Type of random forest: regression
                     Number of trees: 100
No. of variables tried at each split: 9

          Mean of squared residuals: 0.2537966
                    % Var explained: 65.66
```

```
hitters.test = Hitters2[-train,"Salary"]
yhat.bag = predict(bag.model.p,
                   newdata=Hitters2[-train,included.vars])
mean((yhat.bag - log(hitters.test))^2) #Test MSE
```

```
[1] 0.3052927
```

Test MSE from a pruned tree

```
[1] 0.4216292
```

Classification Problem

We will use the *Heart* data. Recall trees.

```
tree.pred No Yes
      No  65  21
      Yes 15  50
```

```
[1] 0.2384106
```

$$> \frac{21+15}{65+21+15+50} \Rightarrow \text{error "MSE"}$$

What is the name of this result?

Bagging for Classification Problem

```
Call:
 randomForest(formula = AHD ~ ., data = Heart, ntree = B, mtry = p,       im
               Type of random forest: classification
                     Number of trees: 1000
No. of variables tried at each split: 13

        OOB estimate of  error rate: 20.2%
Confusion matrix:
     No Yes class.error
No  132  28   0.1750000
Yes  32 105   0.2335766
```

Estimating the Test Error

- On average, each bagged tree makes use of around two-thirds of the B observations. $\lim_{B \to \infty} 1 - \frac{1}{e^{1}} \approx 0.632$
- The remaining one-third of the observations not used to fit a given bagged tree are referred to as the **out-of-bag** (OOB) observations.
- We can predict the response for the ith observation using each of the trees in which that observation was OOB.
- We predict the response for the ith observation using each of the trees in which that observation was OOB. This yields around $B/3$ predictions for the ith observation.
- We can then
  - Average these predicted responses if regression is the goal
  - Take a majority vote if classification is the goal
- Then the overall OOB test error can be computed.

Variable Importance Measure

- Recall advantage of decision trees is the attractive and easily interpreted diagram that results.
- When we bag a large number of trees, the resulting statistical learning procedure is no longer using a single tree and no long clear which variable are most important to the procedure.
- We use **relative variable importance measures**
  - If variable is important, the tree split over that variable causes the **RSS** (or **Gini index** for classification trees) to decrease the most.
  - The measure of variable's importance is the total amount by which RSS (Gini) decreased after each split over that variable.
  - The larger the value the more importance is that variable.

Variable Importance Measures in R

In R we used int the `randomForest()` function importance = TRUE.

The following is the plot of the variable importance for *Heart* data. The variables with the largest mean decrease in Gini index are Thal, Ca, and ChestPain.