Exercise 10 - Decision Trees

O’Malley, Conie

2025-03-05

Table of Contents

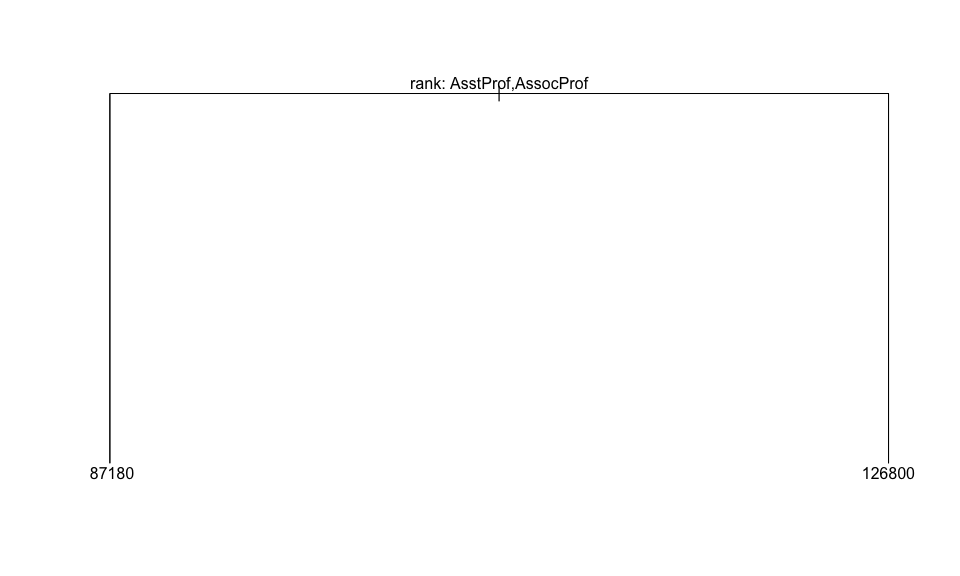
## Preparation

Download the **Ex10\_Trees\_YourLastName.Rmd** R Markdown file to your working directory, rename it to include your last name and follow the instructions below. When you finish, knit it into an HTML file and upload it (the zipped or PDF version of your knitted file) onto Blackboard.

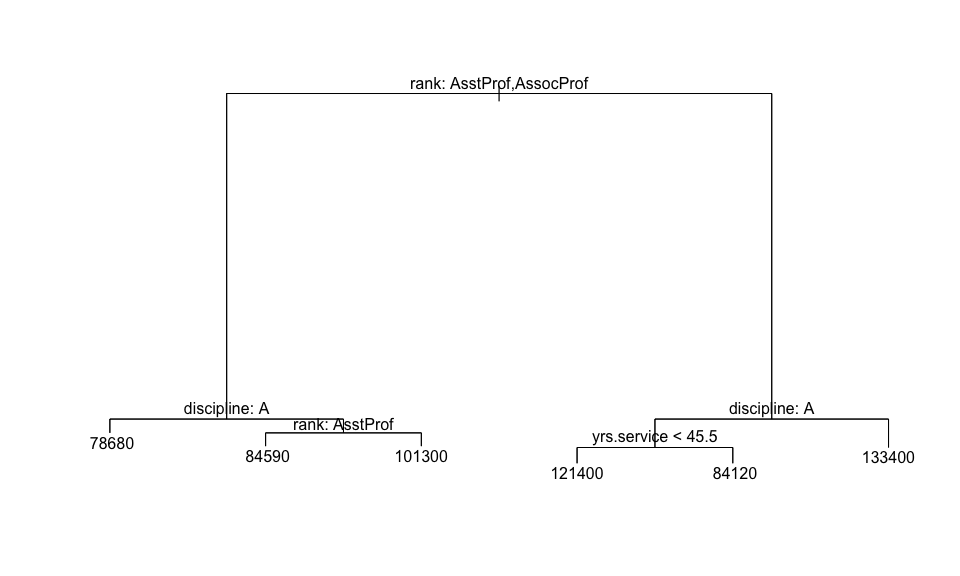
## 1. Regression Trees

1.1 Load the {car} library (contains the Salaries) data set and the {tree} library (contains the tree() function). Then fit a small tree to predict **salary** using rank + discipline + yrs.service + sex as predictors with mindev=0.1. Name this tree object **tree.sal.small**. Then use the plot() and text() functions to graph the resulting tree. Then fit a larger tree using mindev=0.005 and plot it the same way.

library(car)  
library(carData)  
attach(carData::Salaries)  
data(Salaries)  
tree.sal.small <- tree::tree(salary ~ rank + discipline + yrs.service + sex, mindev = 0.1, data = Salaries)  
graphics::plot(tree.sal.small)  
graphics::text(tree.sal.small, pretty = 0)



tree.sal.large <- tree::tree(salary ~ rank + discipline + yrs.service + sex, mindev = 0.005, data = Salaries)  
graphics::plot(tree.sal.large)  
graphics::text(tree.sal.large, pretty = 0)

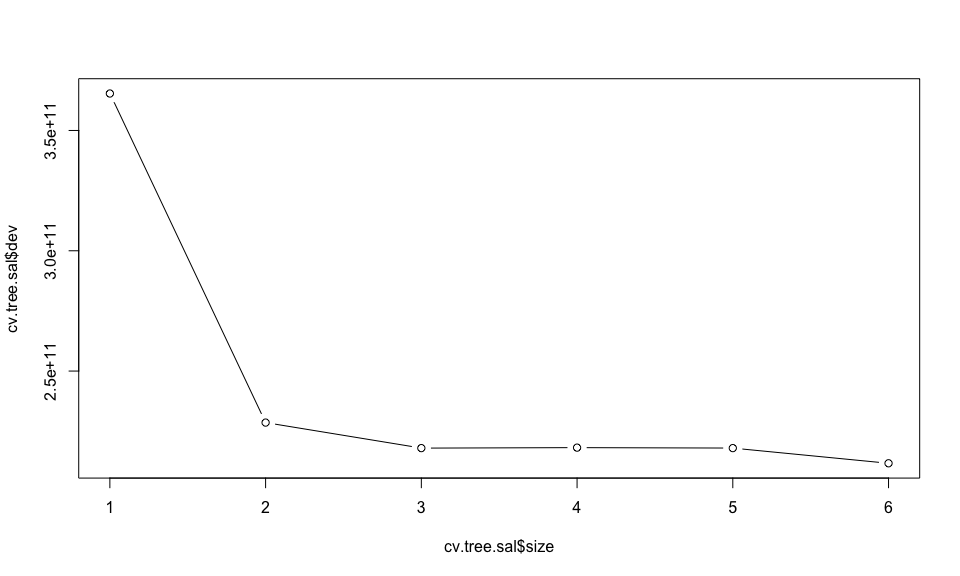


1.2 Now let’s find the optimal tree size with cross-validation. First, issue the command RNGkind(sample.kind="default") and then set the seed to 1. Then fit a tree, exactly as you did above, but do not specify the mindev=. This will fit a default tree size using mindev=0.01, so you don’t have to specify it. Name this tree **tree.sal**.

RNGkind(sample.kind="default")   
set.seed(1)  
tree.sal <- tree::tree(salary ~ rank + discipline + yrs.service + sex, data=Salaries)

Then use the cv.tree() function to do cross validation on **tree.sal**. Name the resulting object **cv.tree.sal**. Then use the cbind() function to display all tree sizes ($size) and their respective deviances ($dev). Label the columns as shown below.

cv.tree.sal <- tree::cv.tree(tree.sal)   
graphics::plot(cv.tree.sal$size, cv.tree.sal$dev, type='b')



cbind("Size"= cv.tree.sal$size, "Deviance"=cv.tree.sal$dev)

## Size Deviance  
## [1,] 6 211669434198  
## [2,] 5 217982645237  
## [3,] 4 218176280804  
## [4,] 3 217976604463  
## [5,] 2 228587185210  
## [6,] 1 365360263994

Then use the min() function to calculate the smallest deviance and name it **min.dev**. Then use the which() function to find the tree index number for the smallest deviance (Tip: use which(cv.tree.sal$dev == min.dev) and save the result in an object named **best.ind**.

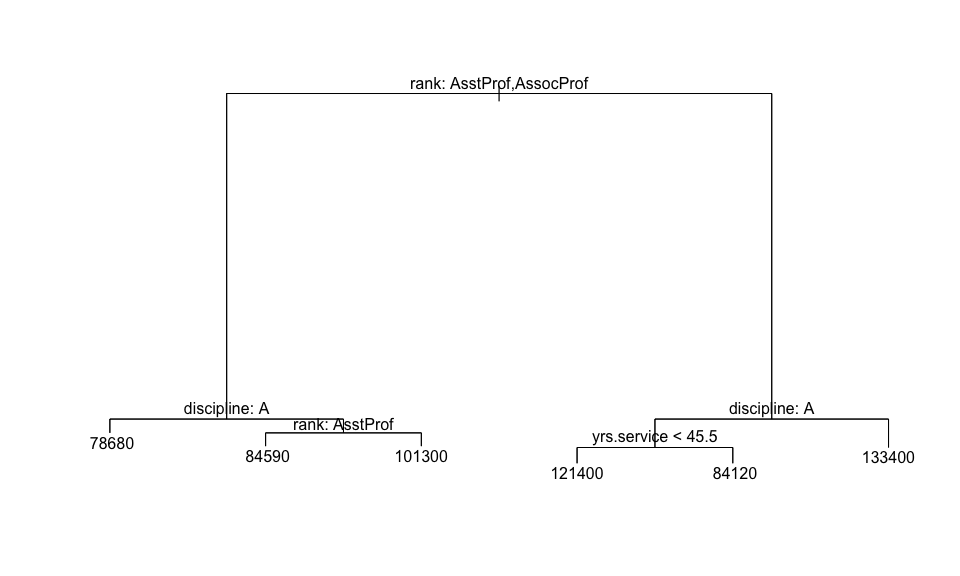
Note that this result is the tree number, but not the tree size. You can see in the display above that trees are listed in descending order of size, so tree #1 is the largest tree. So, use **best.ind** as an index to display the tree size for this tree (Tip: cv.tree.sal$size[best.ind]) and name the result **best.size**. Then use the c() function to display these tree results together, with the labels shown below.

Then use the prune.tree() function to prune tree to this optimal size, using best=best.size) and then display the tree with plot() and text().

min.dev <- min(cv.tree.sal$dev)   
best.ind <- which(cv.tree.sal$dev == min.dev)   
best.size <- cv.tree.sal$size[best.ind]   
c("Min Dev"=min.dev, "Tree No."=best.ind, "Tree Size"=best.size)

## Min Dev Tree No. Tree Size   
## 211669434198 1 6

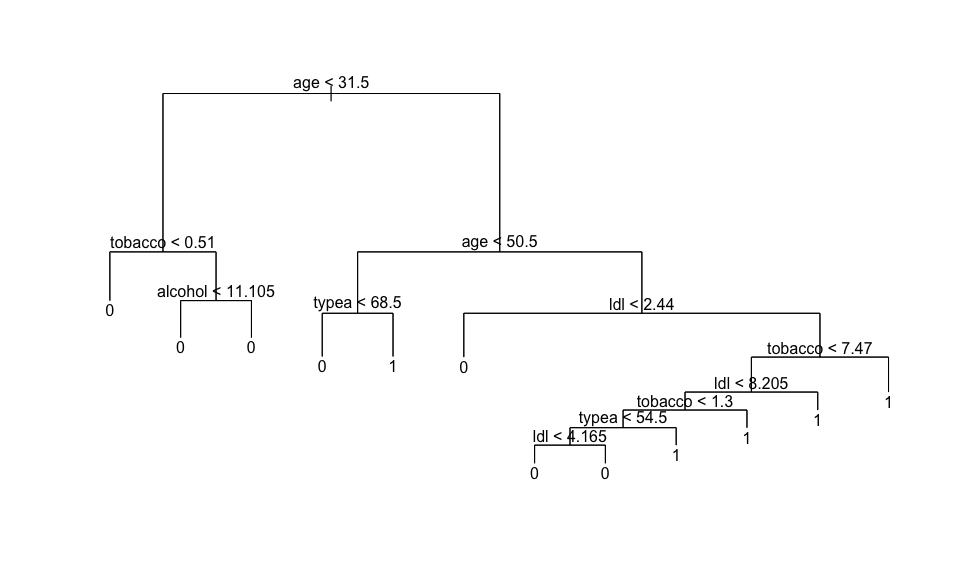
prune.sal <- tree::prune.tree(tree.sal, best=best.size)   
graphics::plot(prune.sal)   
graphics::text(prune.sal, pretty=0)



## 2. Classification Tree

Use the read.table() function to read the **Heart.csv** dataset with sep=",", header=T. Then enter heart$chd <- as.factor(heart$chd) to ensure that the response variable **chd** is read as a factor variable. Then fit a tree named **tree.heart** to predict **chd** using all available predictors and then plot the tree as you did above.

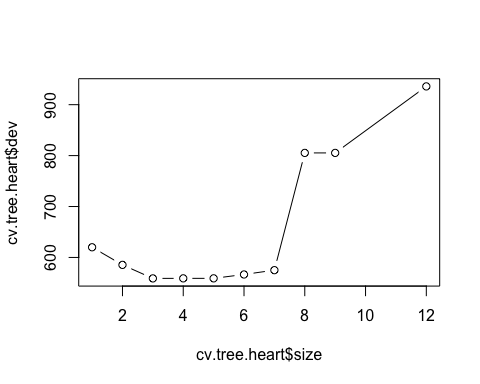
heart <- utils::read.table("/Users/coniecakes/Library/CloudStorage/OneDrive-Personal/001. Documents - Main/023. Programming Tools/R Studio/PredictiveAnalytics/R\_Exercises/data/Heart.csv", sep=",", header=T)   
heart$chd <- as.factor(heart$chd)  
tree.heart <- tree::tree(chd ~ ., data=heart)   
graphics::plot(tree.heart)   
graphics::text(tree.heart, pretty=0)



Then do cross-validation on this tree as you did above and name the resulting cross validation object **cv.tree.heart**. Display the various tree sizes with their respective deviances as you did above. Also, following the steps above, find the smallest deviance, its respective index number and the corresponding tree size and display the results as shown below.

Then prune this tree as you did above using the **best.size** tree and name the resulting tree **prune.heart**. Then plot is as shown below.

set.seed(1)   
cv.tree.heart <- tree::cv.tree(tree.heart)   
graphics::plot(cv.tree.heart$size, cv.tree.heart$dev, type='b')



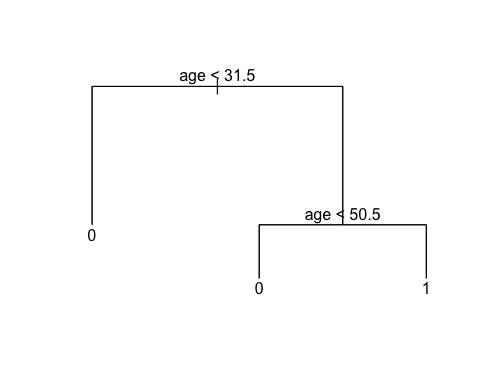
cbind("Size"= cv.tree.heart$size, "Deviance"=cv.tree.heart$dev)

## Size Deviance  
## [1,] 12 935.8905  
## [2,] 9 805.2711  
## [3,] 8 805.2711  
## [4,] 7 575.0283  
## [5,] 6 566.5570  
## [6,] 5 558.9486  
## [7,] 4 558.9486  
## [8,] 3 558.8491  
## [9,] 2 585.4142  
## [10,] 1 619.9996

min.dev <- min(cv.tree.heart$dev) # Find the tree size with smallest deviance   
best.ind <- which(cv.tree.heart$dev == min.dev) # Tree with best CV deviance   
best.size <- cv.tree.heart$size[best.ind] # Tree size with best CV deviance   
c("Min Dev"=min.dev, "Tree No."=best.ind, "Tree Size"=best.size)

## Min Dev Tree No. Tree Size   
## 558.8491 8.0000 3.0000

prune.heart <- tree::prune.tree(tree.heart, best=best.size)   
graphics::plot(prune.heart)   
graphics::text(prune.heart, pretty=0)



### Aside

In this section I’m replicating the code in the lecture slides and R scripts to build a cross-validation confusion matrix (with a lambda classification threshold of 0.5), missclassification statistics and ROC curve.

# Done for you  
  
RNGkind(sample.kind="default")  
set.seed(1)  
  
train <- sample(1:nrow(heart), 0.7\*nrow(heart)) # Train index  
heart.train <- heart[train,] # Train subsample  
heart.test = heart[-train,] # Test subsample  
  
tree.heart.train <- tree::tree(chd ~ ., data=heart.train) # Train the tree  
# We now use the trained model to predict classiifications with the test data  
heart.tree.pred.class <- stats::predict(tree.heart.train, heart.test, type="class") # prob > 0.5  
  
confmat <- table(heart.tree.pred.class, heart.test$chd)  
confmat # Confusion matrix

##   
## heart.tree.pred.class 0 1  
## 0 70 26  
## 1 20 23

TruN <- confmat[1,1]  
TruP <- confmat[2,2]  
FalN <- confmat[1,2]  
FalP <- confmat[2,1]  
TotN <- TruN + FalP  
TotP <- TruP + FalN  
Tot <- TotN + TotP  
  
Accuracy.Rate <- (TruN + TruP) / Tot  
Error.Rate <- (FalN + FalP) / Tot  
Sensitivity <- TruP / TotP  
Specificity <- TruN / TotN  
FalP.Rate <- 1 - Specificity  
  
tree.rates.0.50 <-   
 c("Accuracy Rate"=Accuracy.Rate, "Error Rate"=Error.Rate,   
 "Sensitivity"=Sensitivity, "Specificity"=Specificity,   
 "False Positives"=FalP.Rate)  
  
tree.rates.0.50

## Accuracy Rate Error Rate Sensitivity Specificity False Positives   
## 0.6690647 0.3309353 0.4693878 0.7777778 0.2222222

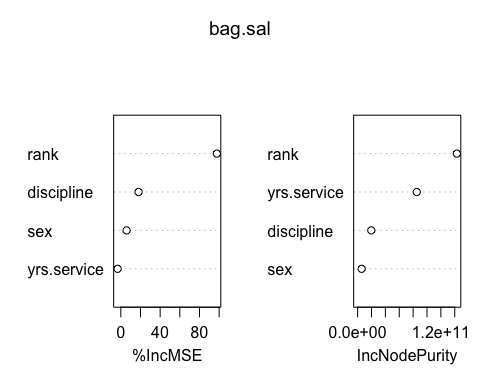
## 3. Bootstrap Aggregation (Bagging) and Random Forest (RF)

1.1 Bagging. First, load the {randomForest} library. Use the randomForest() function to fit the same tree specification we used in 1.2 above, but using **bagging**. To fit such tree, you need to specify mtry=4, which the total number of predictors available. Also use the parameter importance=T to obtain the variable importance statistics. Name this tree **bag.sal**, display it and then use the varImpPlot() and importance() functions to display the results.

library(randomForest)  
bag.sal <- randomForest::randomForest(salary ~ rank + discipline + yrs.service + sex, data=Salaries, mtry=4, importance=T)   
bag.sal

##   
## Call:  
## randomForest(formula = salary ~ rank + discipline + yrs.service + sex, data = Salaries, mtry = 4, importance = T)   
## Type of random forest: regression  
## Number of trees: 500  
## No. of variables tried at each split: 4  
##   
## Mean of squared residuals: 629508508  
## % Var explained: 31.21

randomForest::varImpPlot(bag.sal)



randomForest::importance(bag.sal)

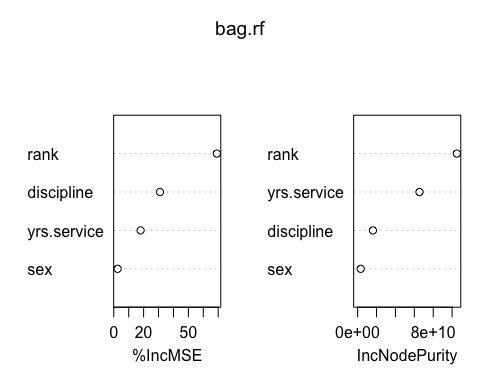
## %IncMSE IncNodePurity  
## rank 97.643757 143002249251  
## discipline 18.038955 19732307198  
## yrs.service -3.255820 85229295746  
## sex 5.841683 5783579092

1.2 Random Forest. Now do the same as you did above, but fit a **Random Forest** tree instead. All you need to do is write the same instructions as above, but name your tree object **bag.rf** and use an **mtry** that is less than the full number of predictors. For this example use mtry=2. Then display the results shown below.

bag.rf <- randomForest::randomForest(salary ~ rank + discipline + yrs.service + sex, data=Salaries, mtry=2, importance=T)   
bag.rf

##   
## Call:  
## randomForest(formula = salary ~ rank + discipline + yrs.service + sex, data = Salaries, mtry = 2, importance = T)   
## Type of random forest: regression  
## Number of trees: 500  
## No. of variables tried at each split: 2  
##   
## Mean of squared residuals: 531591228  
## % Var explained: 41.91

randomForest::varImpPlot(bag.rf)



randomForest::importance(bag.rf)

## %IncMSE IncNodePurity  
## rank 69.084916 104993118823  
## discipline 30.965057 16234719411  
## yrs.service 17.961544 65473631491  
## sex 2.572115 3238818965

## 4. Interpretation

4.1 According to your results in 2 above, which are the best predictors of heart disease?

#### Analysis

Based on the tree - tobacco, ldl, and typea are the best predictors of heart disease.

4.2 Describe briefly how you would evaluate the misclassification, sensitivity and specificity of this model, and how you would you evaluate the overall performance of the model.

#### Analsyis

After training and testing my model, I would build a confusion matrix and calculate Acccuracy Rate, Error Rate, Sensitivity, Specificity, and False Positive Rate for the model to evaluate the predictive accuracy. To further evaluate visually, I would plot the ROC curve and calculate AUC to visualize the model’s accuracy compared to its inaccuracies.

4.3 According to your results in 1 and 3 above, which are the best predictors of professor salaries?

#### Analysis

rank and discipline are the best predictors of professor salaries, followed by yrs.service in number 1 - based on the branches of the tree (which branch at these variable points). In number 3, the same variables are important based on reviewing the variable importance plots.

4.4 Briefly discuss the differences between the 3 tree results and how would you go about selecting the best tree modeling approach.

#### Analysis

The three trees have different structures and performance metrics. The first tree has a higher accuracy rate, while the second tree has a lower error rate but a lower sensitivity. The third tree has a higher specificity but a lower overall accuracy rate. To select the best tree modeling approach, I would consider the trade-off between accuracy, sensitivity, and specificity based on the specific requirements of the problem.