KJ exs 8.1, 8.2

Daniel Moscoe

6/30/2021

## Exercise 8.1

#### Question

"Recreate the simulated data from Exercise 7.2:

library(mlbench)  
library(tidyverse)  
set.seed(200)  
simulated <- mlbench.friedman1(200, sd = 1)  
simulated <- cbind(simulated$x, simulated$y)  
simulated <- as.data.frame(simulated)  
colnames(simulated)[ncol(simulated)] <- "y"

"a. Fit a random forest model to all of the predictors, then estimate the variable importance scores:

library(randomForest)  
library(caret)  
model1 <- randomForest(y ~ ., data = simulated,  
 importance = TRUE,  
 ntree = 1000)  
rfImp1 <- varImp(model1, scale = FALSE)

“Did the random forest model significantly use the uninformative predictors (V6-V10)?”

#### Code

rfImp1

## Overall  
## V1 8.732235404  
## V2 6.415369387  
## V3 0.763591825  
## V4 7.615118809  
## V5 2.023524577  
## V6 0.165111172  
## V7 -0.005961659  
## V8 -0.166362581  
## V9 -0.095292651  
## V10 -0.074944788

#### Response

The code above shows the overall variable importance scores for the model. Variables V6-V10 have very low scores relative to the informative variables, V1-V5. The random forest model did not significantly use the uninformative predictors.

#### Question

“b. Now add an additional predictor that is highly correlated with one of the informative predictors…. Fit another random forest model to these data. Did the importance score for V1 change?”

#### Code

simulated2 <- mutate(simulated, "duplicate1" = simulated$V1 + rnorm(200) \* 0.1)  
model2 <- randomForest(y ~ ., data = simulated2,  
 importance = TRUE,  
 ntree = 1000)  
rfImp2 <- varImp(model2, scale = FALSE)  
rfImp2

## Overall  
## V1 5.69119973  
## V2 6.06896061  
## V3 0.62970218  
## V4 7.04752238  
## V5 1.87238438  
## V6 0.13569065  
## V7 -0.01345645  
## V8 -0.04370565  
## V9 0.00840438  
## V10 0.02894814  
## duplicate1 4.28331581

#### Response

The importance score for V1 is reduced when another highly correlated predictor variable is added to the model. This example serves as a warning about interpreting results of random forests that contain correlated variables. When several variables are highly correlated, it’s sometimes more informative to measure their importance together as a group. Or, a measure of variable importance such as Strobl et al.’s could be used instead (see p. 202).

#### Question

“What happens when you add another predictor that is also highly correlated with V1?”

#### Code

simulated3 <- mutate(simulated2, "duplicate2" = simulated$V1 + rnorm(200) \* 0.1)  
model3 <- randomForest(y ~ ., data = simulated3,  
 importance = TRUE,  
 ntree = 1000)  
rfImp3 <- varImp(model3, scale = FALSE)  
rfImp3

## Overall  
## V1 4.91687329  
## V2 6.52816504  
## V3 0.58711552  
## V4 7.04870917  
## V5 2.03115561  
## V6 0.14213148  
## V7 0.10991985  
## V8 -0.08405687  
## V9 -0.01075028  
## V10 0.09230576  
## duplicate1 3.80068234  
## duplicate2 1.87721959

#### Response

Because duplicate2 is highly correlated with both V1 and duplicate1, adding the second duplicate predictor reduces the variable importance scores of both V1 and the first duplicate, duplicate1.

#### Question

“c. Use the cforest function in the party package to fit a random forest model using conditional inference trees. The party package function varimp can calculate predictor importance. The conditional argument of that function toggles between the traditional importance measure and the modified version described in Strobl et al. (2007). Do these importances show the same pattern as the traditional random forest model?”

#### Code

library(partykit)  
  
#CI tree model for original data  
model4 <- cforest(y ~ ., data = simulated, ntree = 1000)  
  
#CI tree model with 1 additional uninformative variable  
model5 <- cforest(y ~ ., data = simulated2, ntree = 1000)  
  
#CI tree model with 2 additional uninformative variables  
model6 <- cforest(y ~ ., data = simulated3, ntree = 1000)  
  
#Variable importance scores with conditional = TRUE  
cfImp4.con <- varimp(model4, conditional = TRUE)  
cfImp5.con <- varimp(model5, conditional = TRUE)  
cfImp6.con <- varimp(model6, conditional = TRUE)  
  
#Variable importance scores with conditional = FALSE  
cfImp4.unc <- varimp(model4, conditional = FALSE)  
cfImp5.unc <- varimp(model5, conditional = FALSE)  
cfImp6.unc <- varimp(model6, conditional = FALSE)

## [1] "No add'l var's, conditional = TRUE"

## V1 V2 V3 V4 V5 V6   
## 6.43516035 5.09145324 0.13215164 5.96513447 1.34879237 -0.19143941   
## V7 V8 V9 V10   
## -0.07077046 -0.36577131 -0.32249962 -0.19569255

## [1] "1 add'l var's, conditional = TRUE"

## V1 V2 V3 V4 V5 V6   
## 3.14305172 4.96480823 0.07081723 5.72365409 1.31623662 -0.11512778   
## V7 V8 V9 V10 duplicate1   
## -0.14987141 -0.19642103 -0.21031583 -0.10791709 2.65165019

## [1] "2 add'l var's, conditional = TRUE"

## V1 V2 V3 V4 V5 V6   
## 2.327551509 4.844125813 -0.008135787 5.542279540 1.309393597 -0.146750782   
## V7 V8 V9 V10 duplicate1 duplicate2   
## -0.084207803 -0.258901852 -0.130949882 -0.154505095 2.182063275 0.826202282

## [1] "No add'l var's, conditional = FALSE"

## V1 V2 V3 V4 V5 V6   
## 8.366934415 5.900653256 0.204321211 7.225381451 2.119240114 -0.033185674   
## V7 V8 V9 V10   
## 0.139209377 -0.133312929 -0.002380075 -0.180685871

## [1] "1 add'l var's, conditional = FALSE"

## V1 V2 V3 V4 V5 V6   
## 6.12821975 5.83017936 0.18164133 6.56046590 1.98517223 -0.05859344   
## V7 V8 V9 V10 duplicate1   
## 0.06089181 -0.05375829 0.14633639 -0.10142300 5.76977589

## [1] "2 add'l var's, conditional = FALSE"

## V1 V2 V3 V4 V5 V6   
## 5.60814342 5.66703847 0.05579206 6.43689954 1.89592645 -0.08902450   
## V7 V8 V9 V10 duplicate1 duplicate2   
## 0.08690583 -0.06608416 0.12985785 -0.07452734 5.43495977 3.08408014

#### Response

As additional variables highly correlated to V1 were added to the model, the importance of V1 declined for both traditional and modified importance measures. For the traditional importance measure, the importance of V1 declined 51% and then 26% with one and two additional predictors, respectively. For the modified importance measure, the importance of V1 declined less: 27% with one additional variable, and an additional 8% when a second highly correlated variable was added to the model. Total decline in the importance of V1 under the traditional importance measure is 64%, and total decline in the importance of V1 under the modified importance measure is 33%. The modified importance measure mitigated the decline in variable importance associated with the addition of highly correlated predictors.

#### Question

“d. Repeat this process with different tree models, such as boosted trees and Cubist. Does the same pattern occur?”

#### Code

#Boosted tree model for original data  
library(gbm)  
model7 <- gbm(y ~ ., data = simulated, distribution = "gaussian")  
  
#Boosted tree model with 1 additional uninformative variable  
model8 <- gbm(y ~ ., data = simulated2, distribution = "gaussian")  
  
#Boosted tree model with 2 additional uninformative variables  
model9 <- gbm(y ~ ., data = simulated3, distribution = "gaussian")  
  
#Variable importance scores with method = relative.influence  
btImp7.ri <- summary.gbm(object = model7, method = relative.influence)

btImp8.ri <- summary.gbm(object = model8, method = relative.influence)

btImp9.ri <- summary.gbm(object = model9, method = relative.influence)

#Variable importance scores with conditional = FALSE  
btImp7.pt <- summary.gbm(object = model7, method = permutation.test.gbm)

btImp8.pt <- summary.gbm(object = model8, method = permutation.test.gbm)

btImp9.pt <- summary.gbm(object = model9, method = permutation.test.gbm)

## [1] "No add'l var's, method = relative.importance"

## var rel.inf  
## V4 V4 29.2480954  
## V1 V1 27.8308275  
## V2 V2 23.5365573  
## V5 V5 10.8579622  
## V3 V3 7.9915978  
## V7 V7 0.2147969  
## V9 V9 0.1726843  
## V6 V6 0.1474784  
## V8 V8 0.0000000  
## V10 V10 0.0000000

## [1] "1 add'l var's, method = relative.importance"

## var rel.inf  
## V4 V4 28.3422124  
## V2 V2 24.0528986  
## duplicate1 duplicate1 14.6551646  
## V1 V1 13.7272625  
## V5 V5 10.8736058  
## V3 V3 8.1951176  
## V6 V6 0.1537385  
## V7 V7 0.0000000  
## V8 V8 0.0000000  
## V9 V9 0.0000000  
## V10 V10 0.0000000

## [1] "2 add'l var's, method = relative.importance"

## var rel.inf  
## V4 V4 31.5714557  
## V2 V2 22.0166944  
## V1 V1 13.6734623  
## duplicate1 duplicate1 13.4204025  
## V5 V5 10.2834297  
## V3 V3 8.3565726  
## V6 V6 0.3935196  
## V7 V7 0.1815789  
## duplicate2 duplicate2 0.1028842  
## V8 V8 0.0000000  
## V9 V9 0.0000000  
## V10 V10 0.0000000

## [1] "No add'l var's, method = relative.permutation.test.gbm"

## var rel.inf  
## 1 V4 36.57125254  
## 2 V1 26.17956859  
## 3 V2 21.65925303  
## 4 V5 9.12945991  
## 5 V3 6.30170169  
## 6 V7 0.06471367  
## 7 V6 0.05471134  
## 8 V9 0.03933923  
## 9 V8 0.00000000  
## 10 V10 0.00000000

## [1] "1 add'l var's, method = relative.permutation.test.gbm"

## var rel.inf  
## 1 V4 37.08826085  
## 2 V2 24.74377153  
## 3 V5 11.53116460  
## 4 V1 9.92921593  
## 5 duplicate1 9.44644702  
## 6 V3 7.19150181  
## 7 V6 0.06963826  
## 8 V7 0.00000000  
## 9 V8 0.00000000  
## 10 V9 0.00000000  
## 11 V10 0.00000000

## [1] "2 add'l var's, method = relative.permutation.test.gbm"

## var rel.inf  
## 1 V4 37.67891247  
## 2 V2 26.31089783  
## 3 V5 11.11713185  
## 4 V1 10.44633418  
## 5 V3 7.22686977  
## 6 duplicate1 6.85659656  
## 7 V6 0.26612477  
## 8 duplicate2 0.07438784  
## 9 V7 0.02274474  
## 10 V8 0.00000000  
## 11 V9 0.00000000  
## 12 V10 0.00000000

#### Response

For boosted trees, there are two methods for assessing variable importance: relative.influence, which is a traditional procedure, and an experimental procedure called permutation.test.gbm. As additional variables highly correlated with V1 are added to the model, the relative importance of V1 declines more steeply under the permutation.test.gbm method than under the relative.influence method. The importance of V1 declines a total of 51% under the relative importance measure when two additional variables are added. Under the permutation test, the total decline in importance for V1 is 60%.

#### Code

#Cubist Model  
library(Cubist)  
  
model10 <- cubist(x = simulated[-11], y = simulated$y, committees = 100)  
model11 <- cubist(x = simulated2[-11], y = simulated2$y, committees = 100)  
model12 <- cubist(x = simulated3[-11], y = simulated3$y, committees = 100)

#Note: output is truncated to emphasize measures of variable importance.  
summary(model10)

## Evaluation on training data (200 cases):  
##   
## Average |error| 1.480315  
## Relative |error| 0.37  
## Correlation coefficient 0.93  
##   
##   
## Attribute usage:  
## Conds Model  
##   
## 47% 96% V1  
## 40% 54% V3  
## 17% 100% V2  
## 96% V4  
## 66% V5  
## 26% V6  
##   
##   
## Time: 0.3 secs

summary(model11)

## Evaluation on training data (200 cases):  
##   
## Average |error| 1.462939  
## Relative |error| 0.36  
## Correlation coefficient 0.93  
##   
##   
## Attribute usage:  
## Conds Model  
##   
## 39% 48% V3  
## 38% 67% V1  
## 27% 92% V2  
## 2% 53% duplicate1  
## 92% V4  
## 8% V8  
## 54% V5  
## 20% V6  
## 2% V10  
##   
##   
## Time: 0.3 secs

summary(model12)

## Evaluation on training data (200 cases):  
##   
## Average |error| 1.354942  
## Relative |error| 0.34  
## Correlation coefficient 0.94  
##   
##   
## Attribute usage:  
## Conds Model  
##   
## 43% 58% V1  
## 38% 63% V3  
## 31% 95% V2  
## 41% duplicate2  
## 92% V4  
## 57% duplicate1  
## 7% V8  
## 57% V5  
## 17% V6  
##   
##   
## Time: 0.4 secs

#### Response

While “There is no established technique for measuring predictor importance for Cubist models” (KJ 212), we can observe the frequency with which a variable appeared in a rule criterion (“conds”). This gives some indication of the importance of a variable during the Cubist modeling process.

In the first model, which does not include additional variables correlated with V1, V1 appeared in rule criteria 47% of the time. As duplicate1 and then duplicate2 were added to the model, V1 appeared in rule criteria 38% and 43% of the time, respectively. This indicates some decline in variable importance for V1 as highly correlated variables are added to the model. In terms of its appearance in final models, V1 underwent a more significant decline: from 96% with no correlated duplicate variables, to 67%, and then to 58%.

## Exercise 8.2

#### Question

“Use a simulation to show tree bias with different granularities.”

#### Code

#Create dataframe with some informative (x11, x12, x13) and some uninformative (x14, x15, x16) predictors.  
  
y <- rnorm(1000, 0, 1)  
x11 <- y + rnorm(1000, 1, 0.5)  
x12 <- y / 3 \* rnorm(1000, 0, 0.5)  
x13 <- (y + rnorm(1000, 0, 0.5)) ^ (1/3)   
x14 <- rnorm(1000, 0, 1)  
x15 <- runif(1000, 4, 6)  
x16 <- rexp(1000, 0.5)  
df\_1 <- data.frame(x11 = x11, x12 = x12, x13 = x13, x14 = x14, x15 = x15, x16 = x16, y = y)  
  
#Generate cutpoints for decreasing granularity of uninformative predictors (50, 20, and 10 distinct values).  
  
x14\_50 <- seq(from = min(df\_1$x14), to = max(df\_1$x14), length.out = 50)  
x14\_20 <- seq(from = min(df\_1$x14), to = max(df\_1$x14), length.out = 20)  
x14\_10 <- seq(from = min(df\_1$x14), to = max(df\_1$x14), length.out = 10)  
  
x15\_50 <- seq(from = min(df\_1$x15), to = max(df\_1$x15), length.out = 50)  
x15\_20 <- seq(from = min(df\_1$x15), to = max(df\_1$x15), length.out = 20)  
x15\_10 <- seq(from = min(df\_1$x15), to = max(df\_1$x15), length.out = 10)  
  
x16\_50 <- seq(from = min(df\_1$x16), to = max(df\_1$x16), length.out = 50)  
x16\_20 <- seq(from = min(df\_1$x16), to = max(df\_1$x16), length.out = 20)  
x16\_10 <- seq(from = min(df\_1$x16), to = max(df\_1$x16), length.out = 10)  
  
#Create low-granularity variables  
  
x24 <- as.numeric(as.vector(cut(df\_1$x14, breaks = 50, labels = x14\_50)))  
x34 <- as.numeric(as.vector(cut(df\_1$x14, breaks = 20, labels = x14\_20)))  
x44 <- as.numeric(as.vector(cut(df\_1$x14, breaks = 10, labels = x14\_10)))  
x25 <- as.numeric(as.vector(cut(df\_1$x15, breaks = 50, labels = x15\_50)))  
x35 <- as.numeric(as.vector(cut(df\_1$x15, breaks = 20, labels = x15\_20)))  
x45 <- as.numeric(as.vector(cut(df\_1$x15, breaks = 10, labels = x15\_10)))  
x26 <- as.numeric(as.vector(cut(df\_1$x16, breaks = 50, labels = x16\_50)))  
x36 <- as.numeric(as.vector(cut(df\_1$x16, breaks = 20, labels = x16\_20)))  
x46 <- as.numeric(as.vector(cut(df\_1$x16, breaks = 10, labels = x16\_10)))  
  
#Construct reduced granularity dataframes  
  
df\_2 <- data.frame(x11 = x11, x12 = x12, x13 = x13, x24 = x24, x25 = x25, x26 = x26, y = y)  
df\_3 <- data.frame(x11 = x11, x12 = x12, x13 = x13, x34 = x34, x35 = x35, x36 = x36, y = y)  
df\_4 <- data.frame(x11 = x11, x12 = x12, x13 = x13, x44 = x44, x45 = x45, x46 = x46, y = y)  
  
#Construct models  
  
library(rpart)  
df\_1.mod <- rpart(y ~ ., data = df\_1)  
df\_2.mod <- rpart(y ~ ., data = df\_2)  
df\_3.mod <- rpart(y ~ ., data = df\_3)  
df\_4.mod <- rpart(y ~ ., data = df\_4)

## [1] "Variable importance with high granularity uninformative vars"

## Overall  
## x11 2.55169623  
## x12 0.90264412  
## x13 1.24738927  
## x14 0.08509515  
## x15 0.13668230  
## x16 0.10924922

## [1] "Variable importance with uninformative vars (50 distinct vals)"

## Overall  
## x11 2.55169623  
## x12 0.90264412  
## x13 1.24738927  
## x24 0.06262627  
## x25 0.12793460  
## x26 0.07365423

## [1] "Variable importance with uninformative vars (20 distinct vals)"

## Overall  
## x11 2.55169623  
## x12 0.90264412  
## x13 1.24738927  
## x34 0.06487874  
## x35 0.10749465  
## x36 0.05263083

## [1] "Variable importance with uninformative vars (10 distinct vals"

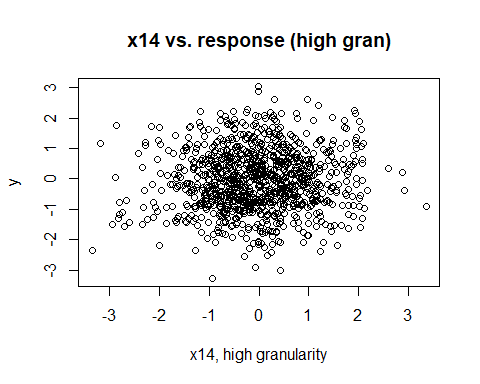
## Overall  
## x11 2.55169623  
## x12 0.90264412  
## x13 1.24738927  
## x44 0.03412582  
## x45 0.10152804  
## x46 0.04846781

#### Response

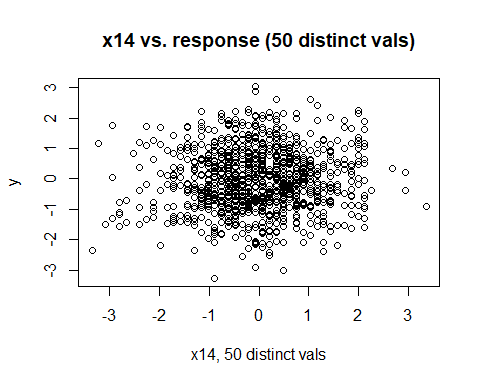
Tree bias due to granularity occurs when a variable’s importance in a tree model is inflated not due to its strong relationship to the response, but merely because of the large number of distinct values that variable takes on. Variables that take on large numbers of distinct values are said to be granular. To demonstrate tree bias due to granularity, we examined a single dataset comprised of some informative and some uninformative variables. By binning the uninformative variables (first to 50 distinct values, then 20, then 10) we were able to reduce their granularity. If bias due to granularity exists in the original high-granularity data, then the variable importance of the uninformative variables should decrease as they are binned to fewer and fewer distinct values.

The plots below show the relationship between X14, an uninformative variable, and the response variable, as the granularity of x14 is decreased.

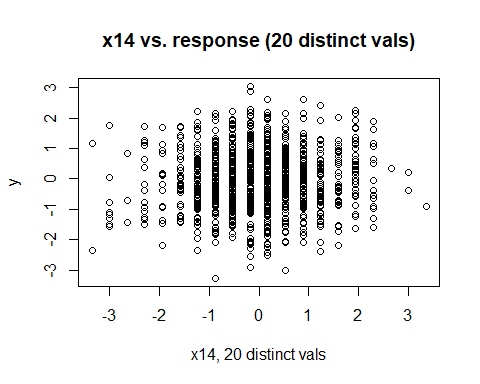
plot(df\_1$x14, df\_1$y, xlab = "x14, high granularity", ylab = "y", main = "x14 vs. response (high gran)")



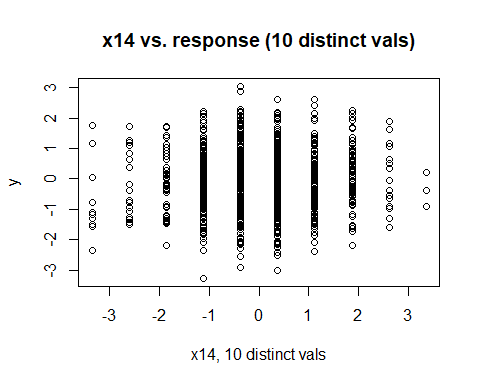
plot(df\_2$x24, df\_2$y, xlab = "x14, 50 distinct vals", ylab = "y", main = "x14 vs. response (50 distinct vals)")



plot(df\_3$x34, df\_3$y, xlab = "x14, 20 distinct vals", ylab = "y", main = "x14 vs. response (20 distinct vals)")



plot(df\_4$x44, df\_4$y, xlab = "x14, 10 distinct vals", ylab = "y", main = "x14 vs. response (10 distinct vals)")



The code in the **Code** section above shows the construction of the original dataset, along with the process for reducing the granularity of each uninformative variable through binning. This code is followed by variable importance tables for each model as granularity is varied.

As the granularity of the non-explanatory variables decreased, the overall importance of those variables decreased as well. This confirms that increased granularity can bias a tree model toward predictors with greater granularity, even when those predictors contain no information about the response variable.