Week 3, Quiz 3

Austin L. Bistline

September 8, 2018

1.

For this quiz we will be using several R packages. R package versions change over time, the right answers have been checked using the following versions of the packages.

AppliedPredictiveModeling: v1.1.6

caret: v6.0.47

ElemStatLearn: v2012.04-0

pgmm: v1.1

rpart: v4.1.8

If you aren't using these versions of the packages, your answers may not exactly match the right answer, but hopefully should be close.

Load the cell segmentation data from the AppliedPredictiveModeling package using the commands:

```
library(AppliedPredictiveModeling)
data(segmentationOriginal)
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

1. Subset the data to a training set and testing set based on the Case variable in the data set.

```
head(segmentationOriginal[, 1:7])
```

```
##        Cell  Case Class   AngleCh1 AngleStatusCh1 AreaCh1 AreaStatusCh1
## 1 207827637  Test    PS 143.247705              1     185             0
## 2 207932307 Train    PS 133.752037              0     819             1
## 3 207932463 Train    WS 106.646387              0     431             0
## 4 207932470 Train    PS  69.150325              0     298             0
## 5 207932455  Test    PS   2.887837              2     285             0
## 6 207827656  Test    WS  40.748298              2     172             0
```

```
segoTest = subset(segmentationOriginal, Case=="Test")
segoTrain = subset(segmentationOriginal, Case=="Train")
head(segoTest[, 1:7])
```

```
##        Cell Case Class   AngleCh1 AngleStatusCh1 AreaCh1 AreaStatusCh1
## 1 207827637 Test    PS 143.247705              1     185             0
## 5 207932455 Test    PS   2.887837              2     285             0
## 6 207827656 Test    WS  40.748298              2     172             0
## 7 207827659 Test    WS 173.957833              1     177             0
## 8 207827661 Test    PS 179.800467              1     251             0
## 9 207932479 Test    WS  18.936420              2     495             0
```

```r
table(segmentationOriginal$Class)
```

```
##
##   PS   WS
## 1300  719
```

2. Set the seed to 125 and fit a CART model with the rpart method using all predictor variables and default caret settings.

```r
set.seed(125)
modelFit = train(Class ~ ., method="rpart", data=segoTrain)
print(modelFit$finalModel)
```

```
## n= 1009
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 1009 373 PS (0.63032706 0.36967294)
##   2) TotalIntenCh2< 45323.5 454  34 PS (0.92511013 0.07488987) *
##   3) TotalIntenCh2>=45323.5 555 216 WS (0.38918919 0.61081081)
##     6) FiberWidthCh1< 9.673245 154  47 PS (0.69480519 0.30519481) *
##     7) FiberWidthCh1>=9.673245 401 109 WS (0.27182045 0.72817955) *
```

3. In the final model what would be the final model prediction for cases with the following variable values:

a. TotalIntench2 = 23,000; FiberWidthCh1 = 10; PerimStatusCh1=2

b. TotalIntench2 = 50,000; FiberWidthCh1 = 10;VarIntenCh4 = 100

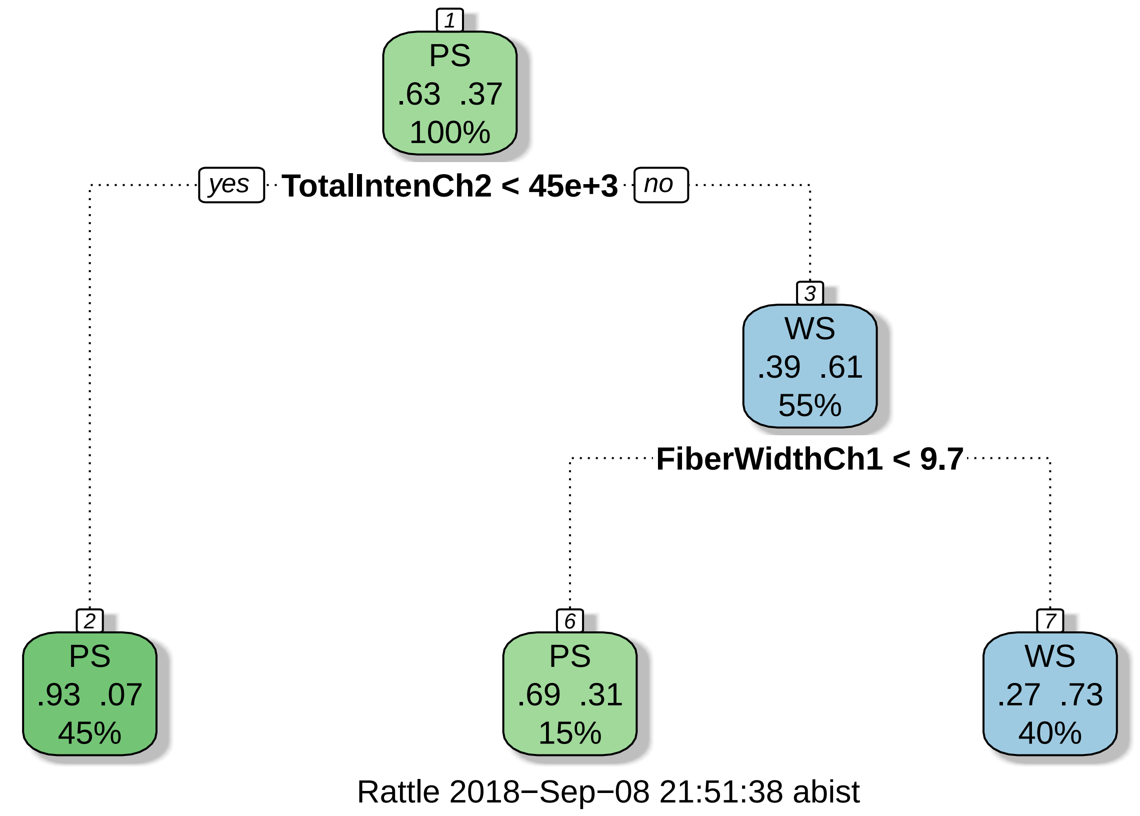
c. TotalIntench2 = 57,000; FiberWidthCh1 = 8;VarIntenCh4 = 100

d. FiberWidthCh1 = 8;VarIntenCh4 = 100; PerimStatusCh1=2

```r
library(rattle)
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```r
fancyRpartPlot(modelFit$finalModel)
```



a = PS because TotalIntenCh2 < 45,000 b = WS because TotalIntenCh2 > 45,000 and FiberWidthCh1 > 9.7 c = PS because TotalIntenCh2 > 45,000 and FiberWidthCh1 < 9.7 d is not possible to predict

2.

If K is small in a K-fold cross validation is the bias in the estimate of out-of-sample (test set) accuracy smaller or bigger? If K is small is the variance in the estimate of out-of-sample (test set) accuracy smaller or bigger. Is K large or small in leave one out cross validation?

The bias is bigger and the variance is smaller. With Leave One Out X-validation, K would be equal to the sample size.

3.

Load the olive oil data using the commands:

```r
library(pgmm)
data(olive)
olive = olive[,-1]
```

(NOTE: If you have trouble installing the pgmm package, you can download the -code-olive-/code- dataset here: olive_data.zip. After unzipping the archive, you can load the file using the -code-load()-/code- function in R.)

These data contain information on 572 different Italian olive oils from multiple regions in Italy. Fit a classification tree where Area is the outcome variable. Then predict the value of area for the following data frame using the tree command with all defaults

```
head(olive)
```

```
##   Area Palmitic Palmitoleic Stearic Oleic Linoleic Linolenic Arachidic
## 1    1     1075          75     226  7823      672        36        60
## 2    1     1088          73     224  7709      781        31        61
## 3    1      911          54     246  8113      549        31        63
## 4    1      966          57     240  7952      619        50        78
## 5    1     1051          67     259  7771      672        50        80
## 6    1      911          49     268  7924      678        51        70
##   Eicosenoic
## 1         29
## 2         29
## 3         29
## 4         35
## 5         46
## 6         44
```

```
inTrain = createDataPartition(y=olive$Area, p=0.7, list=FALSE)
oliveTrain = olive[inTrain, ]
oliveTest = olive[-inTrain, ]
dim(oliveTrain); dim(oliveTest)
```

```
## [1] 401   9
```

```
## [1] 171   9
```

```
library(caret)
areaFit = train(Area ~ ., method="rpart", data=oliveTrain)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info =
## trainInfo, : There were missing values in resampled performance measures.
```

```
print(areaFit$finalModel)
```

```
## n= 401
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
## 1) root 401 2273.28200 4.586035
##   2) Eicosenoic>=6.5 224  132.38840 2.727679 *
##   3) Eicosenoic< 6.5 177  388.31640 6.937853
##     6) Linoleic>=1053.5 69   14.60870 5.304348 *
##     7) Linoleic< 1053.5 108   71.96296 7.981481 *
```

```
predict(areaFit, newdata = as.data.frame(t(colMeans(olive))))
```

```
##        1
## 2.727679
```

This answer is strange because Area denotes a region of origin in Italy - but it is expected from the colMeans process.

4.

Load the South Africa Heart Disease Data and create training and test sets with the following code:

```
library(ElemStatLearn)
data(SAheart)
set.seed(8484)
train = sample(1:dim(SAheart)[1],size=dim(SAheart)[1]/2,replace=F)
trainSA = SAheart[train,]
testSA = SAheart[-train,]
```

Then set the seed to 13234 and fit a logistic regression model (method="glm", be sure to specify family="binomial") with Coronary Heart Disease (chd) as the outcome and age at onset, current alcohol consumption, obesity levels, cumulative tabacco, type-A behavior, and low density lipoprotein cholesterol as predictors. Calculate the misclassification rate for your model using this function and a prediction on the "response" scale:

```
names(SAheart)
```

```
##  [1] "sbp"       "tobacco"   "ldl"       "adiposity" "famhist"
##  [6] "typea"     "obesity"   "alcohol"   "age"       "chd"
```

```
set.seed(13234)
trainFit = glm(chd ~ age + alcohol + obesity + tobacco + typea + ldl, family="binomial", data=trainSA)
testFit = glm(chd ~ age + alcohol + obesity + tobacco + typea + ldl, family="binomial", data=testSA)

trainPrediction = predict(trainFit, newdata = SAheart)
testPrediction = predict(testFit, newdata = SAheart)

missClass = function(values, prediction) {
  sum(((prediction > 0.5)*1) != values)/length(values)
}
missClass(SAheart$chd, testPrediction)
```

```
## [1] 0.3051948
```

```
missClass(SAheart$chd, trainPrediction)
```

```
## [1] 0.2770563
```

Answers don't exactly match, but are close to 31 and 27.

5.

Load the vowel.train and vowel.test data sets:

```
library(ElemStatLearn)
data(vowel.train)
data(vowel.test)
```

Set the variable y to be a factor variable in both the training and test set. Then set the seed to 33833. Fit a random forest predictor relating the factor variable y to the remaining variables. Read about variable importance in random forests here: http://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#ooberr The caret package uses by default the Gini importance.

Calculate the variable importance using the varImp function in the caret package. What is the order of variable importance?

[NOTE: Use randomForest() specifically, not caret, as there's been some issues reported with that approach. 11/6/2016]

```
head(vowel.train)
```

```
##   y    x.1   x.2    x.3   x.4    x.5   x.6    x.7    x.8    x.9   x.10
## 1 1 -3.639 0.418 -0.670 1.779 -0.168 1.627 -0.388  0.529 -0.874 -0.814
## 2 2 -3.327 0.496 -0.694 1.365 -0.265 1.933 -0.363  0.510 -0.621 -0.488
## 3 3 -2.120 0.894 -1.576 0.147 -0.707 1.559 -0.579  0.676 -0.809 -0.049
## 4 4 -2.287 1.809 -1.498 1.012 -1.053 1.060 -0.567  0.235 -0.091 -0.795
## 5 5 -2.598 1.938 -0.846 1.062 -1.633 0.764  0.394 -0.150  0.277 -0.396
## 6 6 -2.852 1.914 -0.755 0.825 -1.588 0.855  0.217 -0.246  0.238 -0.365
```

```
vowel.train$y = as.factor(vowel.train$y)
vowel.test$y = as.factor(vowel.test$y)
```

```
set.seed(33833)
vFit = train(y ~ ., method="rf", data=vowel.train)
varImp(vFit)
```

```
## rf variable importance
##
##      Overall
## x.2  100.000
## x.1   97.511
## x.5   39.196
## x.6   25.532
## x.8   22.663
## x.4    9.684
## x.3    7.975
## x.9    5.655
## x.7    3.589
## x.10   0.000
```