Stat 154 HW9

Mokssh Surve

4/25/2020

```r
library('ISLR')
library('rpart')
library('rpart.plot')
library('stats')
library('factoextra')
```

```
## Loading required package: ggplot2

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```r
set.seed(123)
```

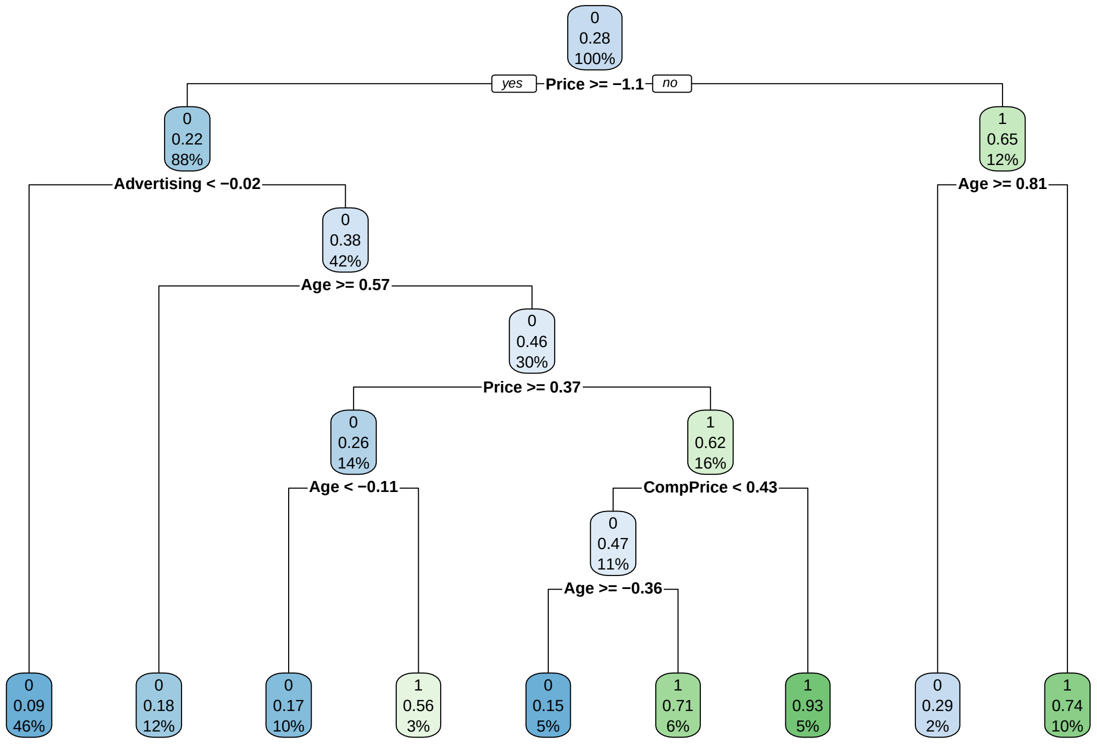
Problem 6

a)

```r
data("Carseats")

#trimming data set
data <-  Carseats[, -c(7,10,11)]
n <- nrow(data)

#creating variable High
data$High <- rep(9, n)
data$High[which(data$Sales >= 9)] <- 1
data$High[which(data$Sales < 9)] <- 0
data <- data[, -1]

#scaling
scaled.data <- scale(data[, 1:7])
data[, 1:7] <- scaled.data

#training/test split - choosing a 60-40 train-test split
train <- 0.7
train.indices <- sample(1:n, floor(train * n))
data.train <- as.data.frame(data[train.indices, ])
data.test <- as.data.frame(data[-train.indices, ])
```

b)

```r
tree <- rpart(High ~ CompPrice + Income + Advertising + Population + Price + Age + Education, data = da
#plot(tree, margin=0.15, compress = TRUE, uniform = TRUE)
#text(tree, fancy = TRUE,  use.n = TRUE, all = TRUE, cex=0.75)
rpart.plot(tree)
```



```r
confusion_mat <- table(predict(tree, data.test ,type = 'class'), data.test$High)
selectivity <- (confusion_mat[1,1]) / (confusion_mat[1,1] + confusion_mat[2,1])
specificity <- (confusion_mat[2,2]) / (confusion_mat[2,2] + confusion_mat[1,2])
accuracy <- (confusion_mat[1,1] + confusion_mat[2,2]) / sum(confusion_mat)

confusion_mat
```

```
##
##       0  1
##    0 67 18
##    1 16 19
```

```r
selectivity
```

```
## [1] 0.8072289
```

```r
specificity
```

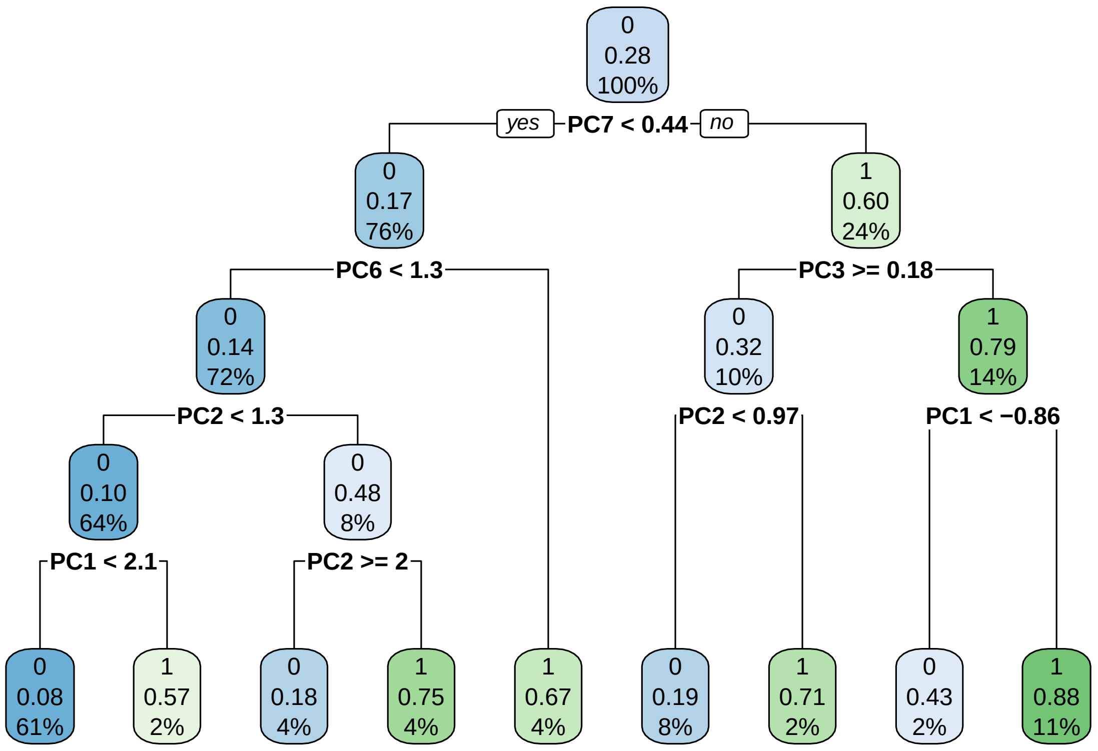
```
## [1] 0.5135135
```

```
accuracy
```

```
## [1] 0.7166667
```

c) PCA

```
pca.data <- Carseats[, -c(1,7,10,11)]

pca <- prcomp(pca.data, scale = TRUE)
pca.var <- get_pca_var(pca)

pca.train <- as.data.frame(pca$x[train.indices, ])
pca.train$High <- data$High[train.indices]
pca.test <- as.data.frame(pca$x[-train.indices, ])
pca.test$High <- data$High[-train.indices]

pca.tree <- rpart(High ~ PC1+PC2+PC3+PC4+PC5+PC6+PC7, data = pca.train, method = 'class', maxdepth=6)
#plot(pca.tree, margin=0.15, compress = TRUE, uniform = TRUE)
#text(pca.tree, fancy = TRUE,  use.n = TRUE, all = TRUE, cex=0.75)
rpart.plot(pca.tree)
```



```
confusion_mat <- table(predict(pca.tree, pca.test ,type = 'class'), pca.test$High)
selectivity <- (confusion_mat[1,1]) / (confusion_mat[1,1] + confusion_mat[2,1])
specificity <- (confusion_mat[2,2]) / (confusion_mat[2,2] + confusion_mat[1,2])
accuracy <- (confusion_mat[1,1] + confusion_mat[2,2]) / sum(confusion_mat)

confusion_mat
```

```
##
##      0  1
##   0 68 22
##   1 15 15
```

```
selectivity
```

```
## [1] 0.8192771
```

```
specificity
```

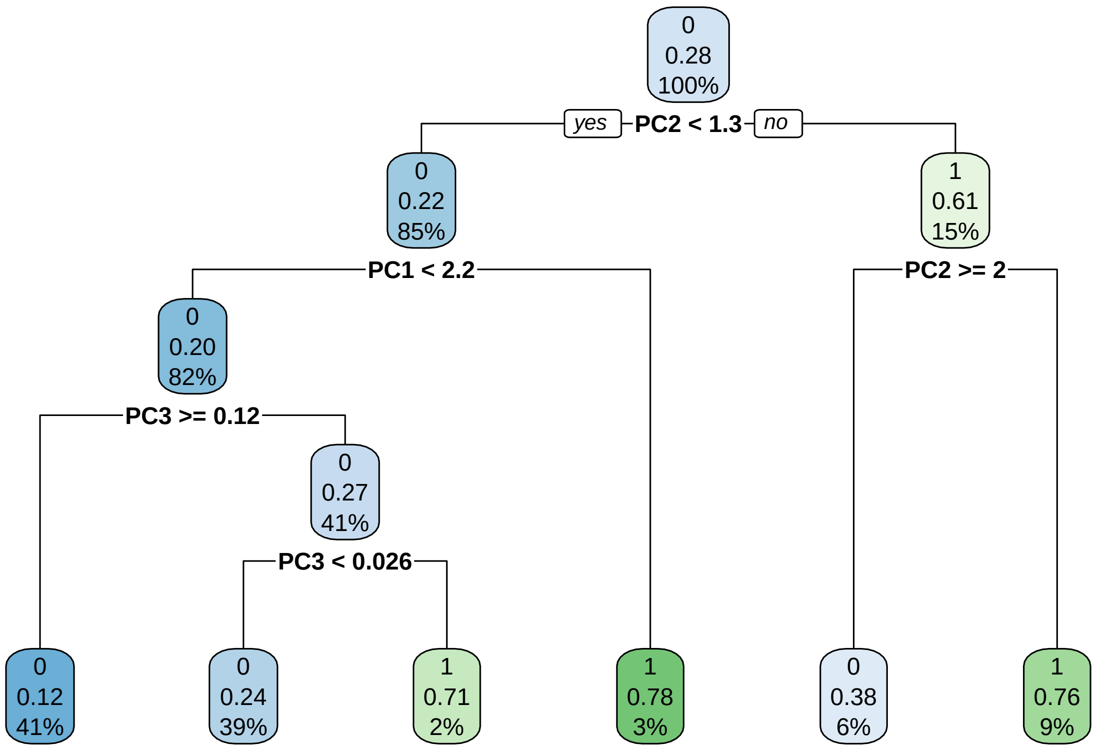
```
## [1] 0.4054054
```

```
accuracy
```

```
## [1] 0.6916667
```

d) Fist 3 components

```
pca.tree.trim <- rpart(High ~ PC1+PC2+PC3, data = pca.train, method = 'class', maxdepth=6)
#plot(pca.tree.trim, margin=0.15, compress = TRUE, uniform = TRUE)
#text(pca.tree.trim, fancy = TRUE,  use.n = TRUE, all = TRUE, cex=0.75)
rpart.plot(pca.tree.trim)
```



```
confusion_mat <- table(predict(pca.tree.trim, pca.test ,type = 'class'), pca.test$High)
selectivity <- (confusion_mat[1,1]) / (confusion_mat[1,1] + confusion_mat[2,1])
specificity <- (confusion_mat[2,2]) / (confusion_mat[2,2] + confusion_mat[1,2])
accuracy <- (confusion_mat[1,1] + confusion_mat[2,2]) / sum(confusion_mat)

confusion_mat
```

```
##
##      0  1
##   0 75 30
##   1  8  7
```

```
selectivity
```

```
## [1] 0.9036145
```

```
specificity
```

```
## [1] 0.1891892
```

```
accuracy
```

```
## [1] 0.6833333
```

Q6 e)

In this question, PCA is used to reduce dimensions before tree-based classification is carried out. Although dimension reduction can be thought to reduce consideration on "unimportant" variables that could skew the results from ideality, in this case, the effect is quite counterintuitive - the accuracy rate of the classification decreases after PCA is carried out on the data.
Note: Assume 0 to signify the True value for discussion purposes.

In this data set, the True (or low-sellers or 0) values were more in abundance in both the training set as well as the overall distribution. Thus, the PCA leads to the variables that play a greater role in the True values to be heavily weighted in the first 3 PCs. This translates into the Selectivity/TPR (True Positive Rate) being higher for the PCA'ed data set, and the Specificity/TNR (True Negative Rate) being much lower. The following are the summarised conclusions from this question:

PROS
• Dimension reduction **trims the dataset** and makes further analysis and classification less expensive a process.
• Dimension reduction makes **visualising the tree easier** a task.
In this setting:
Dimension reduction led to the **TPR/Selectivity rate to rise.**

CONS
• Dimension reduction could lead to a **lack of explanation** of individuals of a certain class that form the minority in the dataset - leading to a higher misclassification rate for the particular class. In this case, the TNR fell by a non-trivial amount post dimension reduction.

• Dimension reduction **reduces the interpretability** of the data.
• Dimension reduction makes making sense out of the thresholds on the tree branches non-interpretable - as is seen in the tree for part d), **PCA1 < 1.9** for the root to first node means not much in terms of the actual variables like *Price, Advertising, Age, etc.*
In this setting:
Dimension reduction led to the **TNR/Specificity rate to fall,** This fall was mucher larger than the rise in TPR - leading to the overall accuracy rate to fall as well. This meant the **Misclassification rate rose** after dimension reduction was carried out.