stats 406 hw8

Li Hsuan Lin

Lab section 002

UM_ID: 49109112

(1)

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ISLR)
library(pls)
```

```
##
## Attaching package: 'pls'
```

```
## The following object is masked from 'package:stats':
##
##     loadings
```

```
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

reuse the code in hw7.

```
set.seed(234)
college_raw = College

#create new response variable
college = college_raw %>% mutate(accept_rate = Accept / Apps)


#split train/test dataset
RNGkind(sample.kind = "Rejection")
nrows_college = nrow(college)
train_id = sample(nrows_college, floor(0.7*nrows_college))
college_train = college[train_id,]
college_test = college[-train_id,]
```

```
#predictors
X = model.matrix(accept_rate ~ ., college)[, -1]
#response
Y <- college$accept_rate
```

(a)

Based on the summary of college dataset, I chose to standardize the predictors first as they seem to measure on the different scales.

Based on the summary of PCA, we need at least 11 eigenvalues/components in order to explain 95% of the variance in this dataset.

It seems that the first PC mostly represents the average of variable Top10perc, Top25perc, Outstate, Room.Board, PhD, Terminal, Expand and Grad.Rate.
For the second PC, it seems that it represents the average of variable PrivateSchool, Apps, Accept, Enroll, and F.Undergrad.

```
summary(college)
```

```
##  Private        Apps           Accept          Enroll       Top10perc
##  No :212   Min.   :   81   Min.   :   72   Min.   :  35   Min.   : 1.00
##  Yes:565   1st Qu.:  776   1st Qu.:  604   1st Qu.: 242   1st Qu.:15.00
##            Median : 1558   Median : 1110   Median : 434   Median :23.00
##            Mean   : 3002   Mean   : 2019   Mean   : 780   Mean   :27.56
##            3rd Qu.: 3624   3rd Qu.: 2424   3rd Qu.: 902   3rd Qu.:35.00
##            Max.   :48094   Max.   :26330   Max.   :6392   Max.   :96.00
##    Top25perc      F.Undergrad     P.Undergrad         Outstate
##  Min.   :  9.0   Min.   :  139   Min.   :    1.0   Min.   : 2340
##  1st Qu.: 41.0   1st Qu.:  992   1st Qu.:   95.0   1st Qu.: 7320
##  Median : 54.0   Median : 1707   Median :  353.0   Median : 9990
##  Mean   : 55.8   Mean   : 3700   Mean   :  855.3   Mean   :10441
##  3rd Qu.: 69.0   3rd Qu.: 4005   3rd Qu.:  967.0   3rd Qu.:12925
##  Max.   :100.0   Max.   :31643   Max.   :21836.0   Max.   :21700
##    Room.Board       Books           Personal         PhD
##  Min.   :1780   Min.   :  96.0   Min.   : 250   Min.   :  8.00
##  1st Qu.:3597   1st Qu.: 470.0   1st Qu.: 850   1st Qu.: 62.00
##  Median :4200   Median : 500.0   Median :1200   Median : 75.00
##  Mean   :4358   Mean   : 549.4   Mean   :1341   Mean   : 72.66
##  3rd Qu.:5050   3rd Qu.: 600.0   3rd Qu.:1700   3rd Qu.: 85.00
##  Max.   :8124   Max.   :2340.0   Max.   :6800   Max.   :103.00
##     Terminal       S.F.Ratio      perc.alumni        Expend
##  Min.   : 24.0   Min.   : 2.50   Min.   : 0.00   Min.   : 3186
##  1st Qu.: 71.0   1st Qu.:11.50   1st Qu.:13.00   1st Qu.: 6751
##  Median : 82.0   Median :13.60   Median :21.00   Median : 8377
##  Mean   : 79.7   Mean   :14.09   Mean   :22.74   Mean   : 9660
##  3rd Qu.: 92.0   3rd Qu.:16.50   3rd Qu.:31.00   3rd Qu.:10830
##  Max.   :100.0   Max.   :39.80   Max.   :64.00   Max.   :56233
##    Grad.Rate       accept_rate
##  Min.   : 10.00   Min.   :0.1545
##  1st Qu.: 53.00   1st Qu.:0.6756
##  Median : 65.00   Median :0.7788
```

```
##  Mean   : 65.46   Mean   :0.7469
##  3rd Qu.: 78.00   3rd Qu.:0.8485
##  Max.   :118.00   Max.   :1.0000
```

```
college_PCA = prcomp(x = X[train_id,],center = TRUE,scale. = TRUE)
plot(college_PCA,main = "scree plot of the eigenvalues")
```


```
summary(college_PCA)
```

```
## Importance of components:
##                           PC1    PC2     PC3    PC4     PC5    PC6     PC7
## Standard deviation     2.3554 2.2464 1.08769 1.0147 0.97138 0.9477 0.79955
## Proportion of Variance 0.3082 0.2804 0.06573 0.0572 0.05242 0.0499 0.03552
## Cumulative Proportion  0.3082 0.5886 0.65430 0.7115 0.76393 0.8138 0.84934
##                           PC8     PC9    PC10    PC11    PC12    PC13
## Standard deviation     0.76844 0.69643 0.61998 0.59602 0.53974 0.43967
## Proportion of Variance 0.03281 0.02695 0.02135 0.01974 0.01618 0.01074
## Cumulative Proportion  0.88215 0.90909 0.93045 0.95018 0.96637 0.97711
##                           PC14    PC15    PC16    PC17    PC18
## Standard deviation     0.38491 0.32996 0.28547 0.21288 0.16805
## Proportion of Variance 0.00823 0.00605 0.00453 0.00252 0.00157
## Cumulative Proportion  0.98534 0.99139 0.99591 0.99843 1.00000
```

```
cumsum((college_PCA$sdev)^2)/sum((college_PCA$sdev)^2)
```

```
##  [1] 0.3082229 0.5885783 0.6543047 0.7115077 0.7639292 0.8138265 0.8493422
##  [8] 0.8821475 0.9090928 0.9304471 0.9501829 0.9663673 0.9771068 0.9853376
## [15] 0.9913861 0.9959136 0.9984311 1.0000000
```

Below is the ouput for the first and the second components.

```
college_PCA$rotation[,1:2]
```

```
##                       PC1         PC2
## PrivateYes  -0.062546079  0.35219990
## Apps        -0.238959092 -0.32381712
## Accept      -0.180543932 -0.36363068
```

```
## Enroll      -0.143951433 -0.39380001
## Top10perc   -0.356438459  0.05002215
## Top25perc   -0.339809756  0.01910038
## F.Undergrad -0.118244306 -0.40445653
## P.Undergrad -0.004632796 -0.29641340
## Outstate    -0.318273811  0.20409325
## Room.Board  -0.259916643  0.09907049
## Books       -0.066513963 -0.03568954
## Personal     0.055940733 -0.18346507
## PhD         -0.306400129 -0.09693786
## Terminal    -0.308558424 -0.08811304
## S.F.Ratio    0.204033322 -0.22085352
## perc.alumni -0.219895650  0.21913203
## Expend      -0.324959248  0.09340372
## Grad.Rate   -0.260895172  0.14281641
```

(b)

Based on the output below, I choose 18 components as it yields the lowest cv error.

The test error rate is 0.01334912.

```
college_PCR = pcr(accept_rate~.,data = college,subset = train_id,scale = TRUE,validation = "CV")
summary(college_PCR)
```

```
## Data:    X dimension: 543 18
##  Y dimension: 543 1
## Fit method: svdpc
## Number of components considered: 18
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV          0.1478   0.1277   0.1273   0.1272   0.1272   0.1269   0.1268
## adjCV       0.1478   0.1275   0.1273   0.1272   0.1271   0.1268   0.1267
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV      0.1269   0.1272   0.1264    0.1263    0.1178    0.1184    0.1182
## adjCV   0.1268   0.1272   0.1263    0.1265    0.1176    0.1182    0.1180
##        14 comps  15 comps  16 comps  17 comps  18 comps
## CV       0.1184    0.1130    0.1134   0.09982   0.09511
## adjCV    0.1182    0.1128    0.1132   0.09952   0.09485
##
## TRAINING: % variance explained
##              1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X              30.82    58.86    65.43    71.15    76.39    81.38    84.93
## accept_rate    25.66    26.13    26.44    27.03    28.08    28.34    28.46
##              8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## X              88.21    90.91     93.04     95.02     96.64     97.71
## accept_rate    28.53    29.99     30.55     39.13     39.24     39.90
##              14 comps  15 comps  16 comps  17 comps  18 comps
## X               98.53     99.14     99.59     99.84    100.00
## accept_rate     40.13     46.65     47.05     58.07     62.58
```

```r
pcr_test_predict = predict(college_PCR,college[-train_id,names(college)!="accept_rate"],ncomp = 18)
mean((pcr_test_predict - college[-train_id,"accept_rate"])^2)
```

```
## [1] 0.01334912
```

(c)

Based on the output below, I choose 14 components as it yields the lowest cv error.

The train error is 0.008144985. The test error is 0.01340262.

```r
college_PLS = plsr(accept_rate~.,data = college, subset = train_id,scale = TRUE,validation = "CV")
summary(college_PLS)
```

```
## Data:    X dimension: 543 18
##  Y dimension: 543 1
## Fit method: kernelpls
## Number of components considered: 18
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV          0.1478   0.1253   0.1157   0.1121   0.1071   0.1011  0.09816
## adjCV       0.1478   0.1253   0.1153   0.1122   0.1069   0.1008  0.09791
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV     0.09717  0.09612  0.09535   0.09514   0.09506   0.09503   0.09489
## adjCV  0.09687  0.09591  0.09508   0.09488   0.09480   0.09477   0.09463
##        14 comps  15 comps  16 comps  17 comps  18 comps
## CV      0.09485   0.09487   0.09486   0.09487   0.09486
## adjCV   0.09459   0.09462   0.09460   0.09461   0.09460
##
## TRAINING: % variance explained
##              1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X              30.65    35.36    58.74    66.32    69.38    73.22    76.26
## accept_rate    28.90    42.76    45.29    50.44    56.11    59.33    60.59
##              8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## X              79.75    82.24     85.28     87.95     91.25     92.53
## accept_rate    61.48    62.23     62.38     62.46     62.50     62.55
##              14 comps  15 comps  16 comps  17 comps  18 comps
## X               94.58     96.73     98.06     99.29    100.00
## accept_rate     62.57     62.57     62.58     62.58     62.58
```

```r
pls_train_predict = predict(college_PLS,college[train_id,names(college) != "accept_rate"],ncomp = 14)
pls_train_mse = mean((pls_train_predict - college[train_id,"accept_rate"])^2) %>% print()
```

```
## [1] 0.008144985
```

```r
pls_test_predict = predict(college_PLS,college[-train_id,names(college) != "accept_rate"],ncomp = 14)
pls_test_mse = mean((pls_test_predict - college[-train_id,"accept_rate"])^2) %>% print()
```

```
## [1] 0.01340262
```

(d)

The methods used in hw6 and hw7 yield testing error around 0.008 to 0.010, with the smallest testing error (0.0083) coming from the forward selection model. The testing error of PCR and PLS are both around 0.013.

Since the forward selection model has the best performance and taking into account the fact PCR and PLS models are relatively difficult to interpret, I recommend using forward selection model with this dataset.

(2)

```
crab = crabs
crab %>% head(3)
```

```
##   sp sex index  FL  RW   CL   CW  BD
## 1  B   M     1 8.1 6.7 16.1 19.0 7.0
## 2  B   M     2 8.8 7.7 18.1 20.8 7.4
## 3  B   M     3 9.2 7.8 19.0 22.4 7.7
```

```
str(crab)
```

```
## 'data.frame':    200 obs. of  8 variables:
##  $ sp   : Factor w/ 2 levels "B","O": 1 1 1 1 1 1 1 1 1 1 ...
##  $ sex  : Factor w/ 2 levels "F","M": 2 2 2 2 2 2 2 2 2 2 ...
##  $ index: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ FL   : num  8.1 8.8 9.2 9.6 9.8 10.8 11.1 11.6 11.8 11.8 ...
##  $ RW   : num  6.7 7.7 7.8 7.9 8 9 9.9 9.1 9.6 10.5 ...
##  $ CL   : num  16.1 18.1 19 20.1 20.3 23 23.8 24.5 24.2 25.2 ...
##  $ CW   : num  19 20.8 22.4 23.1 23 26.5 27.1 28.4 27.8 29.3 ...
##  $ BD   : num  7 7.4 7.7 8.2 8.2 9.8 9.8 10.4 9.7 10.3 ...
```

(a)

```
set.seed(6789)
RNGkind(sample.kind = "Rejection")

bm = which(crabs$sp == "B" & crabs$sex == "M") #blueMale
om = which(crabs$sp == "O" & crabs$sex == "M") #orangeMale
bf = which(crabs$sp == "B" & crabs$sex == "F") #blueFemale
of = which(crabs$sp == "O" & crabs$sex == "F") #orangeFemale

train_id = c(sample(bm, size = floor(0.80 * length(bm))),
sample(om, size = floor(0.80 * length(om))), sample(bf, size = floor(0.80 * length(bf))),
sample(of, size = floor(0.80 * length(of))))

crab_train = crab[train_id,]
crab_test = crab[-train_id,]
```

(b)

Based on the output from the cross-validation and the constraint of no more than 10 splits, the optimal size is 10

"FL","CW","BD","CL" are the variables used in the classification tree.

The train error is 0.05263158, and the test error is 0.2903226.

```
library(tree)
```

```
## Registered S3 method overwritten by 'tree':
```

```
##   method     from
##   print.tree cli
```

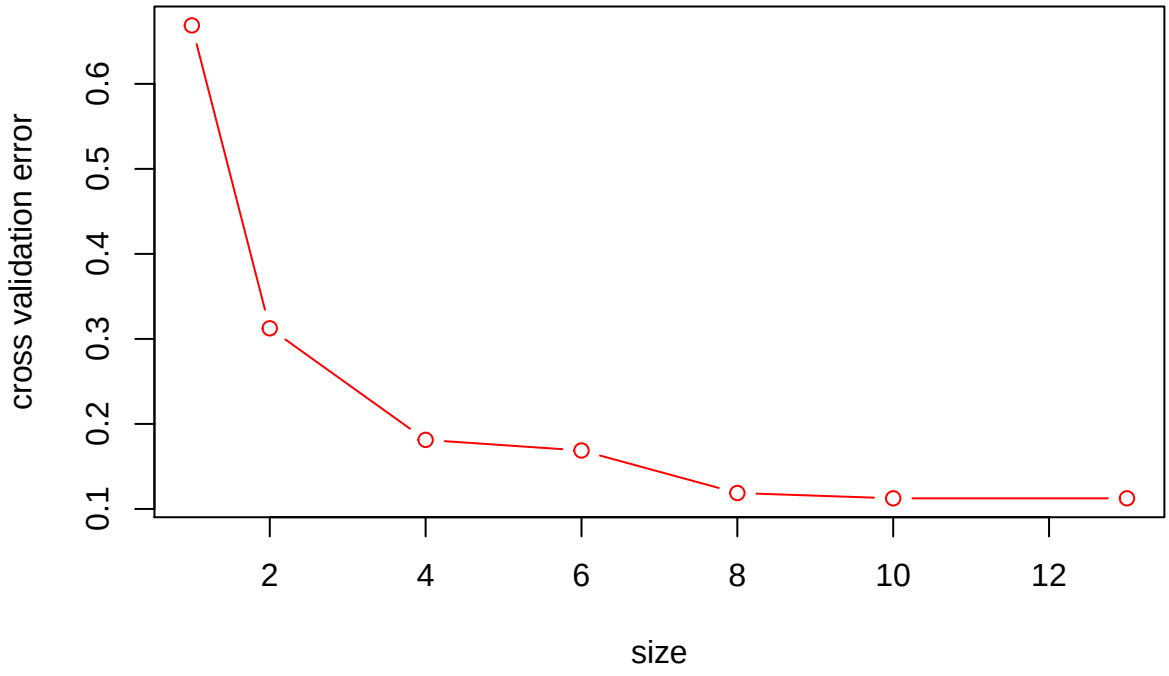
```
library(gbm)
```

```
## Loaded gbm 2.1.5
```

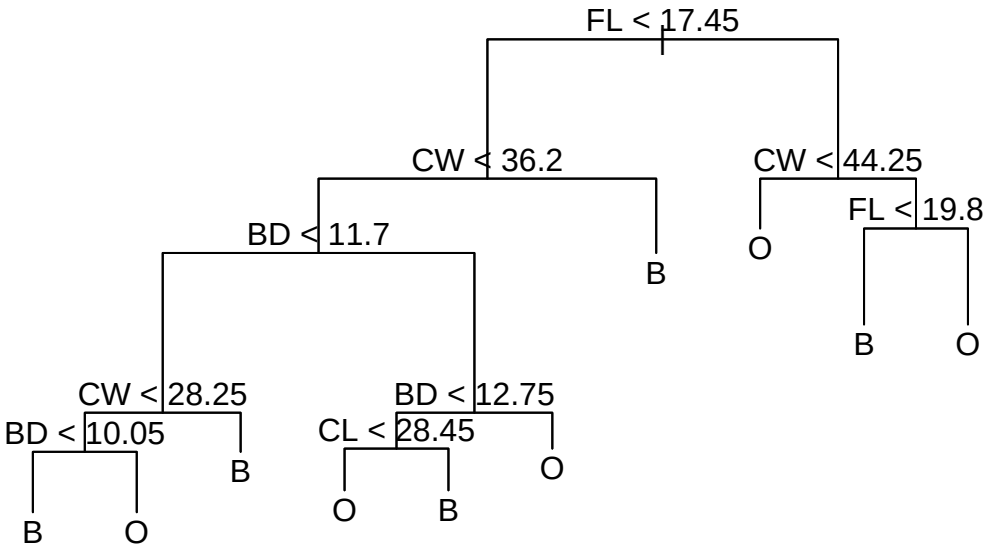
```
crab_tree = tree(sp ~ sex + FL + RW + CL + CW + BD,data = crab,subset = train_id)

cv_crab_tree = cv.tree(crab_tree,FUN = prune.misclass)

plot(cv_crab_tree$size,(cv_crab_tree$dev)/length(train_id),
    ylab = "cross validation error",xlab = "size",type = "b",col = "red")
```




```
pruned_crab = prune.misclass(crab_tree, best=10)
plot(pruned_crab)
text(pruned_crab,pretty = 0)
```




```
crab_train_pred = predict(pruned_crab, crab[train_id,],type="class")
table(crab_train_pred, crabs$sp[train_id])
```

```
##
## crab_train_pred  B  O
##               B 79  7
##               O  1 73
```

```
tree_train_error = 8/(73+79)
tree_train_error
```

```
## [1] 0.05263158
```

```
crab_train_pred = predict(pruned_crab, crab[-train_id,],type="class")
table(crab_train_pred, crabs$sp[-train_id])
```

```
##
## crab_train_pred  B  O
##               B 15  4
##               O  5 16
```

```
tree_test_error = 9/(31)
tree_test_error
```

```
## [1] 0.2903226
```