Applied Data Science II - Homework 3

Phileas Dazeley Gaist

22/01/2021

ISLR 6.6: Questions 9, 11

Libraries

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.0     v forcats 0.5.1

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

```
library(leaps)
library(glmnet)
```

```
## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 4.1-3
```

```r
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(pls)
```

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

## The following object is masked from 'package:caret':
##
##     R2

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

```r
library(ISLR2)
```

1. ISLR 6.6 - 9

a)

```r
# prepare data
head(College)
```

```
##                              Private Apps Accept Enroll Top10perc Top25perc
## Abilene Christian University     Yes 1660   1232    721        23        52
## Adelphi University               Yes 2186   1924    512        16        29
## Adrian College                   Yes 1428   1097    336        22        50
## Agnes Scott College              Yes  417    349    137        60        89
## Alaska Pacific University        Yes  193    146     55        16        44
## Albertson College                Yes  587    479    158        38        62
##                              F.Undergrad P.Undergrad Outstate Room.Board Books
## Abilene Christian University        2885         537     7440       3300   450
## Adelphi University                  2683        1227    12280       6450   750
```

```
## Adrian College                         1036          99     11250         3750    400
## Agnes Scott College                     510          63     12960         5450    450
## Alaska Pacific University               249         869      7560         4120    800
## Albertson College                       678          41     13500         3335    500
##                              Personal PhD Terminal S.F.Ratio perc.alumni Expend
## Abilene Christian University     2200  70       78      18.1          12   7041
## Adelphi University               1500  29       30      12.2          16  10527
## Adrian College                   1165  53       66      12.9          30   8735
## Agnes Scott College               875  92       97       7.7          37  19016
## Alaska Pacific University        1500  76       72      11.9           2  10922
## Albertson College                 675  67       73       9.4          11   9727
##                              Grad.Rate
## Abilene Christian University        60
## Adelphi University                  56
## Adrian College                      54
## Agnes Scott College                 59
## Alaska Pacific University           15
## Albertson College                   55
```

```r
College <- na.omit(College)

rownames(College) <- College$X
College = College[, -1]  # remove original College names column

set.seed(1)

# split data into testing and training sets:
train_size = dim(College)[1]/2  #  half the number of rows in the College set

train = sample(1:dim(College)[1], train_size)  # training sample indices
test = -train  # testing sample indices ('-' means 'exclude')

# create training set from subset of College data set at the training indices
College_train = College[train, ]
# create testing set from subset of College data set at the testing indices
College_test = College[test, ]
```

Generally speaking, how should I decide on the proportions the testing and training sets should represent from the original data set/the ratio of training to testing data? Also, shouldn't we cross validate for lots of different training and testing sets to make sure that our model is more generally applicable to the data? - Is there then a higher risk of overfitting? I imagine some balance must be struck, I would love to know more about how that balance should be reached.

b)

```r
lm_fit = lm(Apps ~ ., data = College_train)  # fit lm
lm_pred = predict(lm_fit, College_test)  # predict test using lm fit
```

```r
mean((College_test[, "Apps"] - lm_pred)^2)  # test mean squared error
```

```
## [1] 1162789
```

c)

```r
train_matrix <- model.matrix(Apps ~ ., data = College_train)  # generate training matrix
test_matrix <- model.matrix(Apps ~ ., data = College_test)  # generate testing matrix

grid <- 10^seq(10, -2, length = 100)  # values of lambda to try ridge regression with
```

```r
# fit ridge regression models using values of lambda specified in grid
ridge_fit <- glmnet(train_matrix, College_train$Apps, alpha = 0, lambda = grid, thresh = 1e-12)
# cross validate the regression models cv.glmnet
cv_ridge <- cv.glmnet(train_matrix, College_train$Apps, alpha = 0, lambda = grid,
    thresh = 1e-12)
# get the best value of lambda from the results of the cross validation
bestlam_ridge <- cv_ridge$lambda.min
bestlam_ridge
```

```
## [1] 0.01
```

```r
# using the best value of lambda, predict values in the test set using the
# corresponding model in ridge_fit note: this uses the predict function from
# the glmnet library, not the base r one
pred_ridge <- predict(ridge_fit, s = bestlam_ridge, newx = test_matrix)
mean((pred_ridge - College_test$Apps)^2)  # mean squared error
```

```
## [1] 1162744
```

```r
# get coefficient estimates
predict(ridge_fit, s = bestlam_ridge, type = "coefficients")
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept) -1.087080e+03
## (Intercept)  .
## Accept       1.778067e+00
## Enroll      -1.517155e+00
## Top10perc    6.683157e+01
## Top25perc   -2.211440e+01
## F.Undergrad  1.129735e-01
## P.Undergrad  1.014295e-02
## Outstate    -1.216374e-01
```

```
## Room.Board   2.053401e-01
## Books        2.560955e-01
## Personal    -2.498585e-04
## PhD         -1.457708e+01
## Terminal     7.590911e+00
## S.F.Ratio    2.866870e+01
## perc.alumni  5.068115e-02
## Expend       5.060304e-02
## Grad.Rate    7.176508e+00
```

```
# alternative using caret with 10-fold cross validation instructions found at
# http://www.sthda.com/english/articles/37-model-selection-essentials-in-r/153-penalized-regre

set.seed(1)
ridge <- train(Apps ~ ., data = College_train, method = "glmnet", trControl = trainControl(met
    number = 10, verboseIter = F), tuneGrid = expand.grid(alpha = 0, lambda = grid))

# Model coefficients
coef(ridge$finalModel, ridge$bestTune$lambda)
```

```
## 17 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept) -2.298365e+03
## Accept       1.130096e+00
## Enroll       3.870027e-01
## Top10perc    3.062361e+01
## Top25perc   -1.400602e-01
## F.Undergrad  6.369905e-02
## P.Undergrad  2.598572e-02
## Outstate    -4.925114e-02
## Room.Board   2.540393e-01
## Books        3.894309e-01
## Personal    -3.462457e-02
## PhD         -7.011688e+00
## Terminal    -1.375565e-02
## S.F.Ratio    3.294793e+01
## perc.alumni -6.481083e+00
## Expend       5.971763e-02
## Grad.Rate    8.196420e+00
```

```
# Make predictions
predictions <- ridge %>%
    predict(College_test)
# Model prediction performance

# return selected model metrics
```

```r
data.frame(RMSE = RMSE(predictions, College_test$Apps), Rsquare = caret::R2(predictions,
    College_test$Apps))
```

```
##       RMSE   Rsquare
## 1 1003.837 0.9158176
```

One thing I don't understand about the approach above using caret versus the other approach used in class is how the approach we used in class determines the type of cross validation. Where are the cross validation parameters specified in the non-caret approach?

I imagine the reason I have different results for the regressions using caret and not using caret is that my specified cross validation settings using caret are not the same as those using the other method, but I have no idea how to see what's going on and where those settings are. Could you help clarify this for me?

d)

```r
# same procedure as for ridge regression
lasso_fit <- glmnet(train_matrix, College_train$Apps, alpha = 1, lambda = grid, thresh = 1e-12)
cv_lasso <- cv.glmnet(train_matrix, College_train$Apps, alpha = 1, lambda = grid,
    thresh = 1e-12)

bestlam_lasso <- cv_lasso$lambda.min
bestlam_lasso
```

```
## [1] 0.01
```

```r
pred_lasso <- predict(lasso_fit, s = bestlam_lasso, newx = test_matrix)
mean((pred_lasso - College_test$Apps)^2)
```

```
## [1] 1162684
```

```r
# get coefficient estimates:
predict(lasso_fit, s = bestlam_lasso, type = "coefficients")
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept) -1.086983e+03
## (Intercept)  .
## Accept       1.778041e+00
## Enroll      -1.516760e+00
## Top10perc    6.682516e+01
## Top25perc   -2.210902e+01
## F.Undergrad  1.129087e-01
## P.Undergrad  1.014763e-02
## Outstate    -1.216233e-01
```

```
## Room.Board   2.053191e-01
## Books        2.560653e-01
## Personal    -2.257221e-04
## PhD         -1.457243e+01
## Terminal     7.586217e+00
## S.F.Ratio    2.866397e+01
## perc.alumni  4.826652e-02
## Expend       5.059989e-02
## Grad.Rate    7.174674e+00
```

```
# alternative using caret with 10-fold cross validation

set.seed(1)
lasso <- train(Apps ~ ., data = College_train, method = "glmnet", trControl = trainControl(meth
    number = 10, verboseIter = F), tuneGrid = expand.grid(alpha = 1, lambda = grid))

# Model coefficients
coef(lasso$finalModel, ridge$bestTune$lambda)
```

```
## 17 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept) -412.629614
## Accept         1.399193
## Enroll         .
## Top10perc     23.470319
## Top25perc      .
## F.Undergrad    .
## P.Undergrad    .
## Outstate       .
## Room.Board     .
## Books          .
## Personal       .
## PhD            .
## Terminal       .
## S.F.Ratio      .
## perc.alumni    .
## Expend         0.001724
## Grad.Rate      .
```

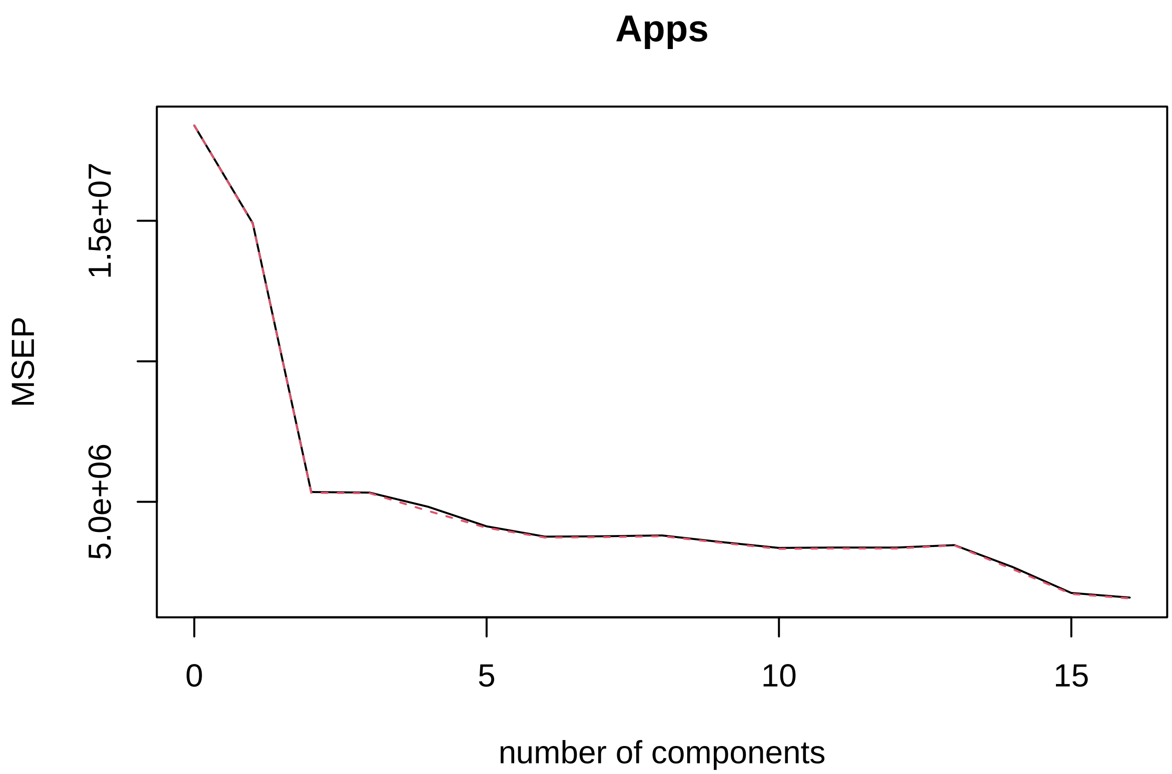
```
# Make predictions
predictions <- lasso %>%
    predict(College_test)
# Model prediction performance

# return selected model metrics
data.frame(RMSE = RMSE(predictions, College_test$Apps), Rsquare = caret::R2(predictions,
    College_test$Apps))
```

```
##       RMSE   Rsquare
## 1 1027.72 0.9132324
```

e)

```r
pcr_fit <- pcr(Apps ~ ., data = College_train, scale = TRUE, validation = "CV")
validationplot(pcr_fit, val.type = "MSEP")
```




```r
# summary(pcr_fit)

pcr_pred <- predict(pcr_fit, College_test, ncomp = 10)
# ncomp could also be lower if the number of components is a concern
mean((pcr_pred - College_test$Apps)^2)
```

```
## [1] 1723911
```

```r
pcr_fit <- pcr(Apps ~ ., data = College, scale = TRUE, ncomp = 10)
summary(pcr_fit)
```

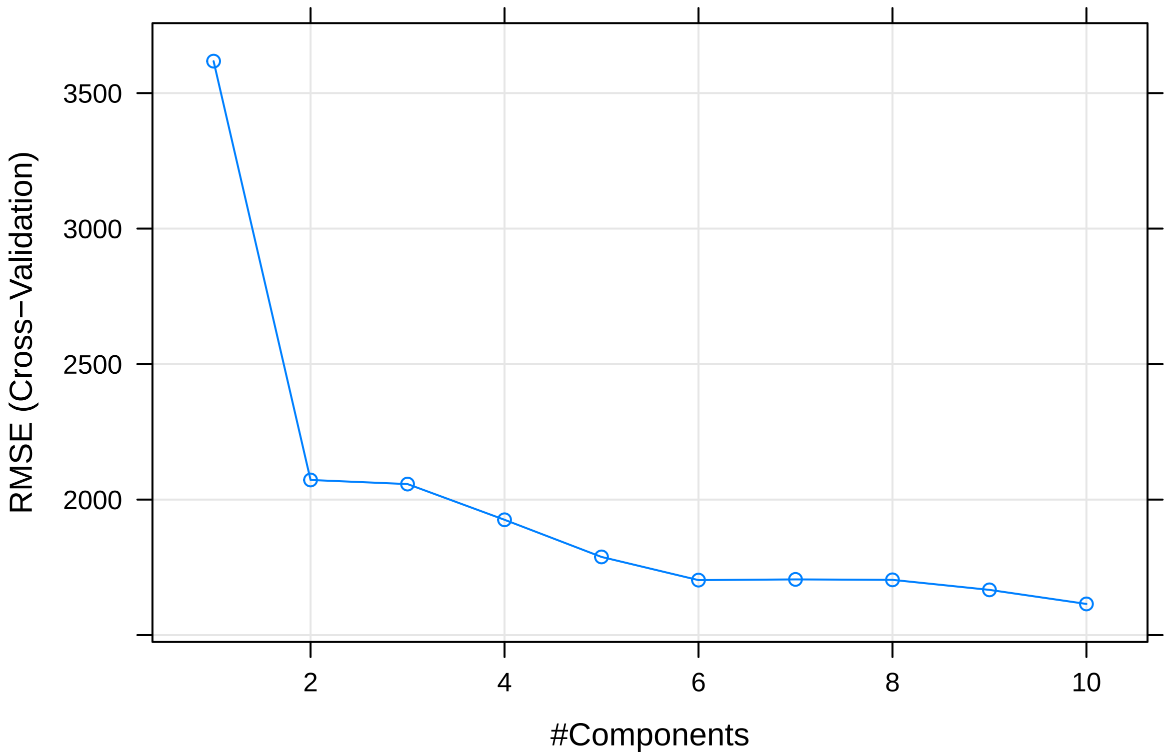
```
## Data:    X dimension: 777 16
##  Y dimension: 777 1
## Fit method: svdpc
## Number of components considered: 10
```

```
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       32.77    57.08    64.39    70.22    75.96    81.25    85.03    88.65
## Apps    10.83    74.28    74.53    74.62    83.94    84.06    84.13    85.00
##       9 comps  10 comps
## X       91.93     94.44
## Apps    85.80     85.93
```

```r
# alternative using caret with 10-fold cross validation solution found at
# http://www.sthda.com/english/articles/37-model-selection-essentials-in-r/152-principal-compo

# Build the model on training set
set.seed(1)
pcr <- train(Apps ~ ., data = College_train, method = "pcr", scale = TRUE, trControl = trainCon
    number = 10), tuneLength = 10)

# Plot model RMSE vs different values of components
plot(pcr)
```




```r
# Print the best tuning parameter ncomp that minimize the cross-validation
# error, RMSE
pcr$bestTune
```

```
##    ncomp
## 10    10
```

```r
# Summarize the final model
summary(pcr$finalModel)
```

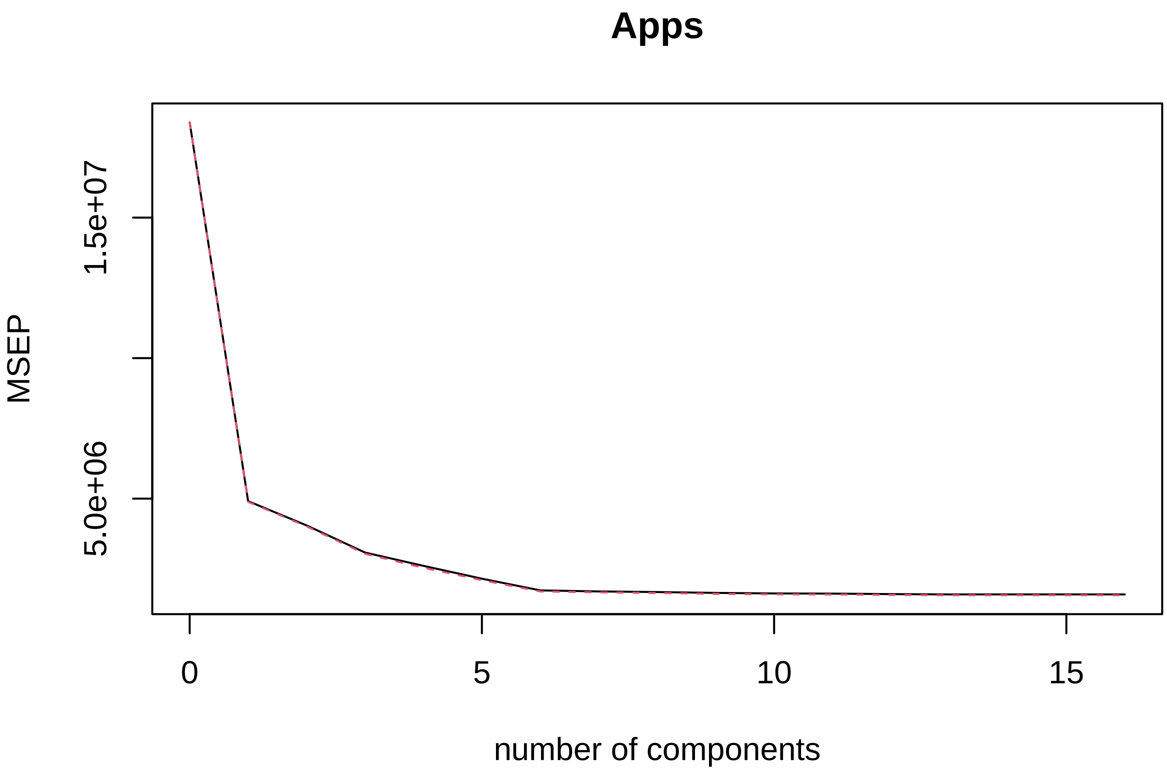
```
## Data:    X dimension: 388 16
##  Y dimension: 388 1
## Fit method: svdpc
## Number of components considered: 10
## TRAINING: % variance explained
##           1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X           33.86    57.78    65.67    71.39    77.06    82.24    85.90
## .outcome    21.01    72.78    73.53    77.64    80.62    82.30    82.32
##           8 comps  9 comps  10 comps
## X           89.14    92.07     94.46
## .outcome    82.35    83.54     84.77
```

```r
# Make predictions
predictions <- pcr %>%
    predict(College_test)
# Model performance metrics
data.frame(RMSE = caret::RMSE(predictions, College_test$Apps), Rsquare = caret::R2(predictions
    College_test$Apps))
```

```
##       RMSE   Rsquare
## 1 1312.978 0.8750703
```

f)

```r
pls_fit <- plsr(Apps ~ ., data = College_train, scale = TRUE, validation = "CV")
validationplot(pls_fit, val.type = "MSEP")
```

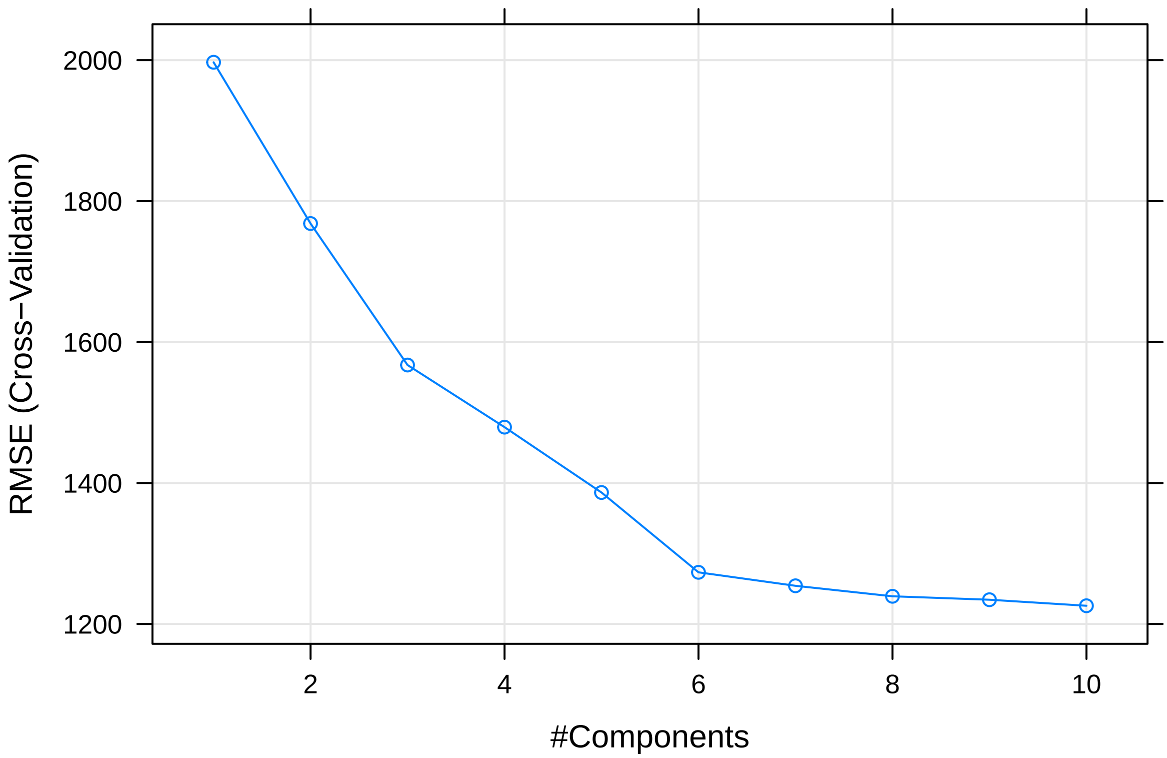



```
pls_pred <- predict(pls_fit, College_test, ncomp = 4)
mean((pls_pred - College_test$Apps)^2)
```

```
## [1] 1456872
```

```
# alternative using caret with 10-fold cross validation

# Build the model on training set
set.seed(1)
pls <- train(Apps ~ ., data = College_train, method = "pls", scale = TRUE, trControl = trainCor
    number = 10), tuneLength = 10)
# Plot model RMSE vs different values of components
plot(pls)
```




```
# Print the best tuning parameter ncomp that minimize the cross-validation
# error, RMSE
pls$bestTune
```

```
##    ncomp
## 10    10
```

```
# Summarize the final model
summary(pls$finalModel)
```

```
## Data:    X dimension: 388 16
##  Y dimension: 388 1
## Fit method: oscorespls
## Number of components considered: 10
## TRAINING: % variance explained
##           1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X           28.23    55.09    63.20    66.89    72.11    74.87    79.19
## .outcome    75.28    80.93    87.12    90.70    92.41    93.40    93.51
##           8 comps  9 comps  10 comps
## X           81.45    83.28     87.36
## .outcome    93.64    93.74     93.76
```

```
# Make predictions
predictions <- pls %>%
```

```r
    predict(College_test)
# Model performance metrics
data.frame(RMSE = caret::RMSE(predictions, College_test$Apps), Rsquare = caret::R2(predictions
    College_test$Apps))
```

```
##       RMSE   Rsquare
## 1 1077.215 0.9051903
```

g)

There isn't much difference between the test errors resulting from the five approaches, but the differences might still affect the fits of the models enough that we might want to choose one over the others. I generally prefer to visualise the linear regressions to make a judgement, but using just the MSE values, I can estimate that pcr and pls result in worse linear model fits than the other approaches.

Model	MSE values
lm	1162789
ridge	1162744
lasso	1162684
pcr	1723911
pls	1456872

2. ISLR 6.6 - 11

a)

```r
Boston <- na.omit(Boston)
head(Boston)
```

```
##      crim zn indus chas   nox    rm  age    dis rad tax ptratio lstat medv
## 1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3  4.98 24.0
## 2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8  9.14 21.6
## 3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8  4.03 34.7
## 4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7  2.94 33.4
## 5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7  5.33 36.2
## 6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7  5.21 28.7
```

```r
set.seed(1)

# split data into testing and training sets:
train_size = dim(Boston)[1]/2
train = sample(1:dim(Boston)[1], train_size)
test = -train
Boston_train = Boston[train, ]
Boston_test = Boston[test, ]
```

Accounting for adjusted R^2, best subset selection suggests using an 8 variable linear model (excluding cas, rm, age, and tax).

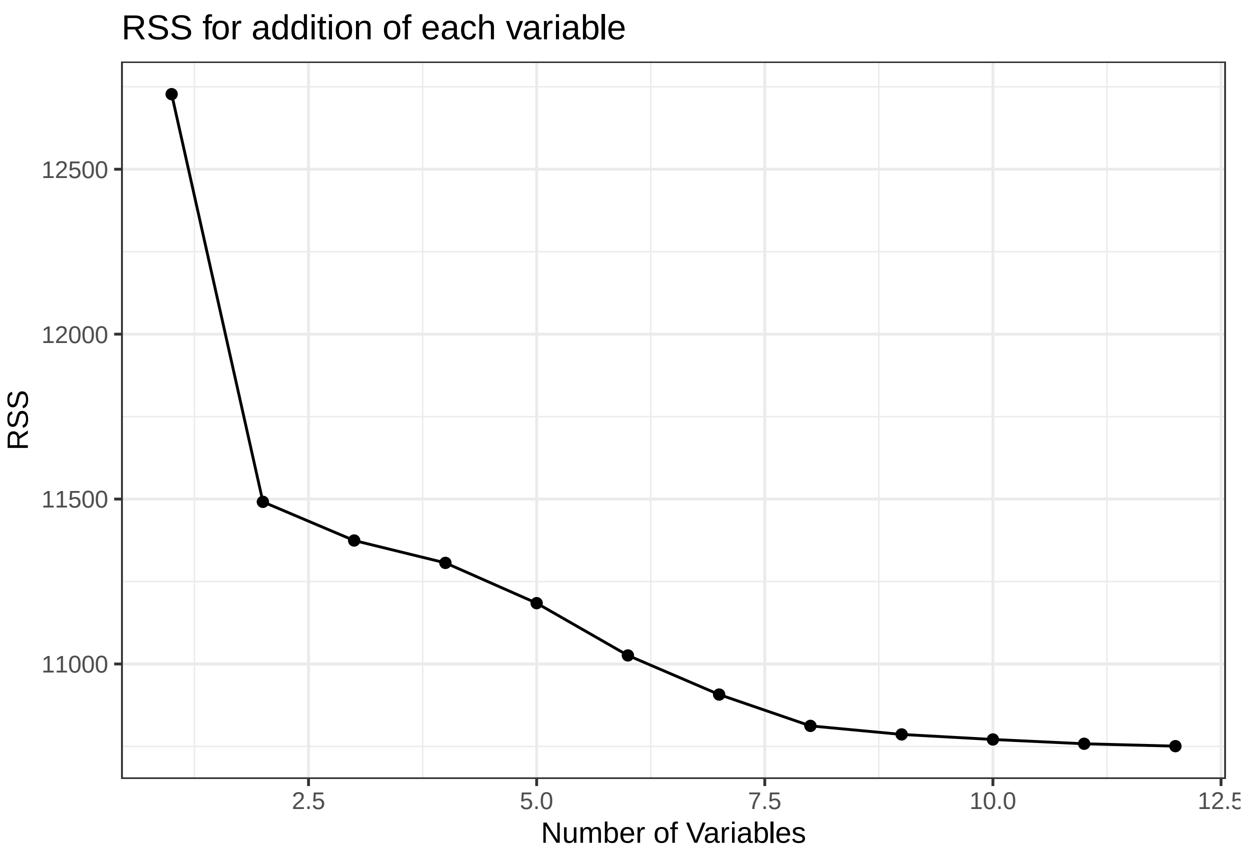
```r
# best subset selection (exhaustive)
regfit_full <- regsubsets(crim ~ ., data = Boston_train, nvmax = 12)
reg_summary <- summary(regfit_full)
reg_summary
```

```
## Subset selection object
## Call: regsubsets.formula(crim ~ ., data = Boston_train, nvmax = 12)
## 12 Variables  (and intercept)
##         Forced in Forced out
## zn          FALSE      FALSE
## indus       FALSE      FALSE
## chas        FALSE      FALSE
## nox         FALSE      FALSE
## rm          FALSE      FALSE
## age         FALSE      FALSE
## dis         FALSE      FALSE
## rad         FALSE      FALSE
## tax         FALSE      FALSE
## ptratio     FALSE      FALSE
## lstat       FALSE      FALSE
## medv        FALSE      FALSE
## 1 subsets of each size up to 12
## Selection Algorithm: exhaustive
##           zn  indus chas nox rm  age dis rad tax ptratio lstat medv
## 1  ( 1 ) " " " "   " "  " " " " " " " " "*" " " " "     " "   " " 
## 2  ( 1 ) " " " "   " "  " " " " " " " " "*" " " " "     "*"   " " 
## 3  ( 1 ) "*" " "   " "  " " " " " " " " "*" " " " "     "*"   " " 
## 4  ( 1 ) " " " "   " "  " " " " " " " " "*" " " "*"     "*"   "*" 
## 5  ( 1 ) "*" "*"   " "  " " " " " " "*" "*" " " " "     "*"   " " 
## 6  ( 1 ) "*" "*"   " "  " " " " " " "*" "*" " " " "     "*"   "*" 
## 7  ( 1 ) "*" " "   " "  "*" " " " " "*" "*" " " "*"     "*"   "*" 
## 8  ( 1 ) "*" "*"   " "  "*" " " " " "*" "*" " " "*"     "*"   "*" 
## 9  ( 1 ) "*" "*"   " "  "*" "*" " " "*" "*" " " "*"     "*"   "*" 
## 10  ( 1 ) "*" "*"   " "  "*" "*" " " "*" "*" "*" "*"     "*"   "*" 
## 11  ( 1 ) "*" "*"   "*"  "*" "*" " " "*" "*" "*" "*"     "*"   "*" 
## 12  ( 1 ) "*" "*"   "*"  "*" "*" "*" "*" "*" "*" "*"     "*"   "*"
```

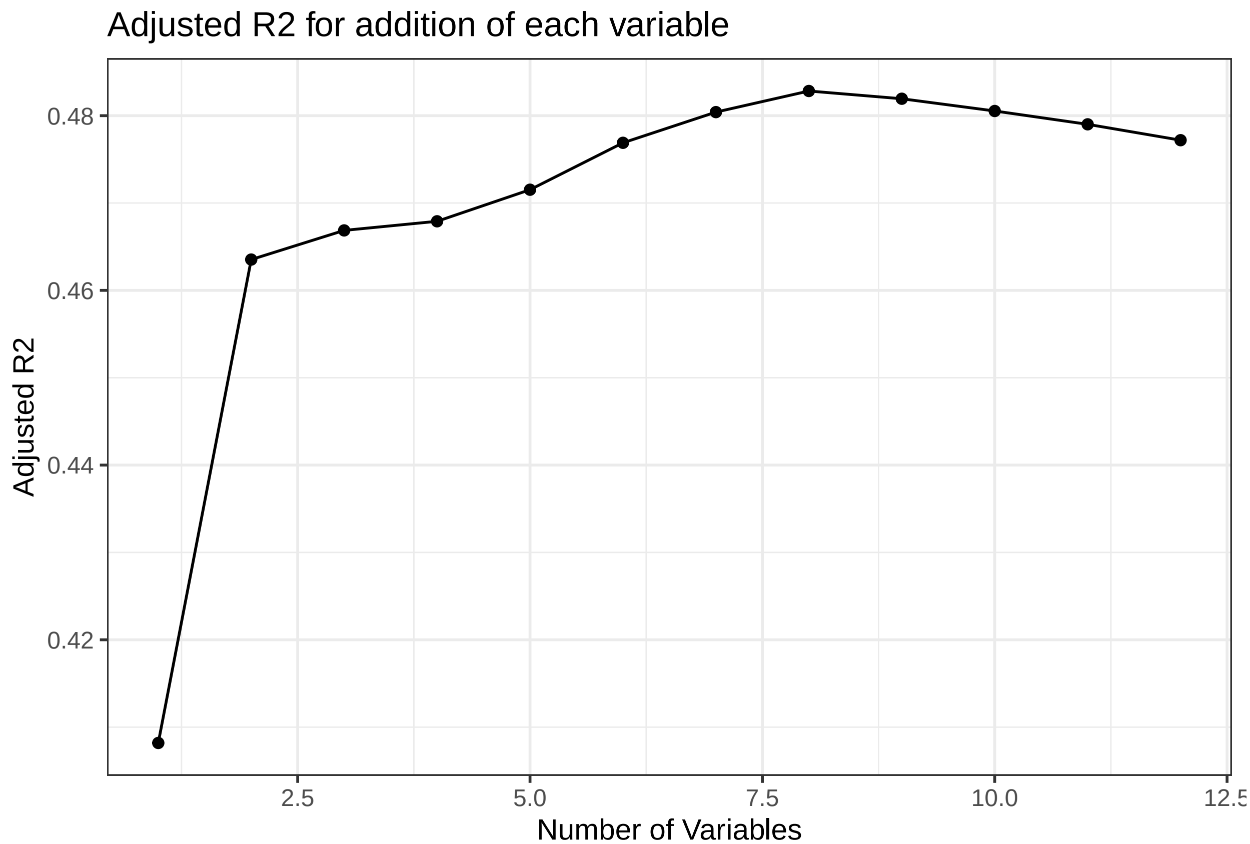
```r
plot_metrics <- data.frame(rss = reg_summary$rss, adjr2 = reg_summary$adjr2, numvar = 1:12)
plot_metrics %>%
    filter(adjr2 == max(adjr2))
```

```
##        rss     adjr2 numvar
## 1 10812.29 0.4828232      8
```

```r
plot_metrics %>%
    ggplot(aes(y = rss, x = numvar)) + geom_point() + geom_line() + xlab("Number of Variables")
    ylab("RSS") + theme_bw() + ggtitle("RSS for addition of each variable")
```




```r
plot_metrics %>%
    ggplot(aes(y = adjr2, x = numvar)) + geom_point() + geom_line() + xlab("Number of Variables
    ylab("Adjusted R2") + theme_bw() + ggtitle("Adjusted R2 for addition of each variable")
```



Ridge regression with 10-fold cross validation suggests a model using $\lambda = 0.869749$.

```
grid <- 10^seq(10, -2, length = 100)  # values of lambda to try

# ridge regression
ridge <- train(crim ~ ., data = Boston_train, method = "glmnet", trControl = trainControl(meth
    number = 10, verboseIter = F), tuneGrid = expand.grid(alpha = 0, lambda = grid))

# suggested lambda
ridge$bestTune$lambda
```

```
## [1] 1.149757
```

```
# Model coefficients
coef(ridge$finalModel, ridge$bestTune$lambda)
```

```
## 13 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept)  4.564514524
## zn           0.034439893
## indus       -0.100221883
## chas        -0.850908924
## nox         -3.323901120
```

```
## rm           0.362575226
## age         -0.003296697
## dis         -0.664878246
## rad          0.403651433
## tax          0.006115416
## ptratio     -0.221670255
## lstat        0.273832643
## medv        -0.147537499
```

```
# Make predictions
predictions <- ridge %>%
    predict(Boston_test)
# Model prediction performance

# return selected model metrics
data.frame(RMSE = RMSE(predictions, Boston_test$crim), Rsquare = caret::R2(predictions,
    Boston_test$crim))
```

```
##       RMSE   Rsquare
## 1 6.314929 0.3786872
```

Ridge regression with 10-fold cross validation suggests a model using $\lambda = 0.09326033$.

```
# lasso regression
lasso <- train(crim ~ ., data = Boston_train, method = "glmnet", trControl = trainControl(metho
    number = 10, verboseIter = F), tuneGrid = expand.grid(alpha = 1, lambda = grid))

# suggested lambda
lasso$bestTune$lambda
```

```
## [1] 0.09326033
```

```
# Model coefficients
coef(lasso$finalModel, ridge$bestTune$lambda)
```

```
## 13 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept) -3.310333783
## zn           .
## indus        .
## chas         .
## nox          .
## rm           .
## age          .
## dis          .
## rad          0.446691263
```

```
## tax          .
## ptratio      .
## lstat        0.240705800
## medv        -0.007101696
```

```r
# Make predictions
predictions <- lasso %>%
    predict(Boston_test)
# Model prediction performance

# return selected model metrics
data.frame(RMSE = RMSE(predictions, Boston_test$crim), Rsquare = caret::R2(predictions,
    Boston_test$crim))
```

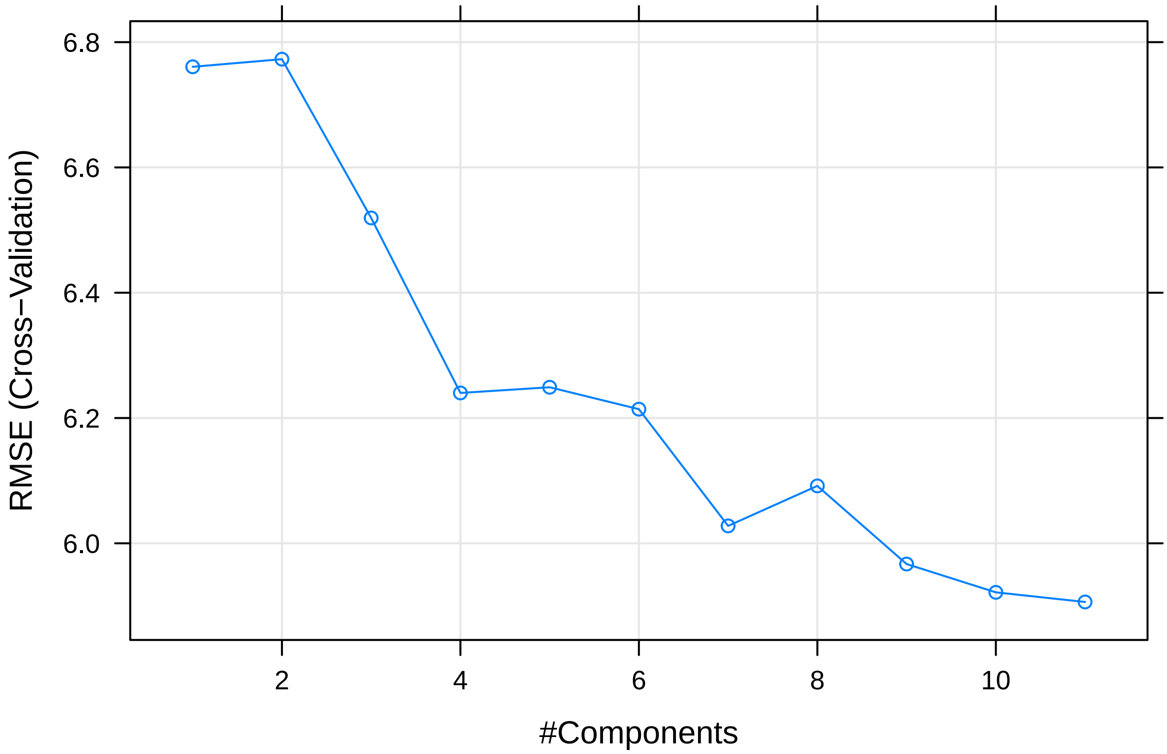
```
##       RMSE   Rsquare
## 1 6.339238 0.3842331
```

The principal component regression with 10-fold cross validation suggests that there is no advantage to dimension reduction in this case.

```r
# principal component regression

# Build the model on training set
set.seed(1)
pcr <- train(crim ~ ., data = Boston_train, method = "pcr", scale = TRUE, trControl = trainCon
    number = 10), tuneLength = 13)

# Plot model RMSE vs different values of components
plot(pcr)
```




```r
# Print the best tuning parameter ncomp that minimize the cross-validation
# error, RMSE
pcr$bestTune
```

```
##    ncomp
## 11    11
```

```r
# Summarize the final model
summary(pcr$finalModel)
```

```
## Data:    X dimension: 253 12
##  Y dimension: 253 1
## Fit method: svdpc
## Number of components considered: 11
## TRAINING: % variance explained
##           1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X           50.61    63.48    72.71    80.29    86.43    90.37    92.94
## .outcome    32.91    33.06    37.89    42.35    42.62    43.12    45.76
##           8 comps  9 comps  10 comps  11 comps
## X           95.04    96.80     98.34     99.50
## .outcome    45.79    47.16     47.87     48.83
```

```r
# Make predictions
predictions <- pcr %>%
```

```r
    predict(Boston_test)
# Model performance metrics
data.frame(RMSE = caret::RMSE(predictions, Boston_test$crim), Rsquare = caret::R2(predictions,
    Boston_test$crim))
```

```
##       RMSE   Rsquare
## 1 6.501073 0.3661381
```

b)

The best performing model we have completed so far is the eight predictor linear model suggested by best subset selection. Its adjusted test R^2 is higher than the test R^2 values of all the other models tested in this exercise.

c)

No, the model suggested by best subset selection has only eight predictors. I chose it due to its test adjusted R^2, which is higher than the test R^2 values of the other proposed models.

Session Info

```r
sessionInfo()
```

```
## R version 4.1.2 (2021-11-01)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur 10.16
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] ISLR2_1.3-1     pls_2.8-0       caret_6.0-90    lattice_0.20-45
##  [5] glmnet_4.1-3    Matrix_1.3-4    leaps_3.1       forcats_0.5.1
##  [9] stringr_1.4.0   dplyr_1.0.7     purrr_0.3.4     readr_2.1.0
## [13] tidyr_1.1.4     tibble_3.1.6    ggplot2_3.3.5   tidyverse_1.3.1
##
## loaded via a namespace (and not attached):
##  [1] nlme_3.1-153         fs_1.5.0             lubridate_1.8.0
##  [4] httr_1.4.2           tools_4.1.2          backports_1.4.1
```

```
##  [7] utf8_1.2.2          R6_2.5.1            rpart_4.1-15
## [10] DBI_1.1.1           colorspace_2.0-2    nnet_7.3-16
## [13] withr_2.4.3         tidyselect_1.1.1    compiler_4.1.2
## [16] cli_3.1.0           rvest_1.0.2         formatR_1.11
## [19] xml2_1.3.2          labeling_0.4.2      scales_1.1.1
## [22] digest_0.6.29       rmarkdown_2.11      pkgconfig_2.0.3
## [25] htmltools_0.5.2     parallelly_1.30.0   highr_0.9
## [28] dbplyr_2.1.1        fastmap_1.1.0       rlang_0.4.12
## [31] readxl_1.3.1        rstudioapi_0.13     farver_2.1.0
## [34] shape_1.4.6         generics_0.1.1      jsonlite_1.7.2
## [37] ModelMetrics_1.2.2.2 magrittr_2.0.1     Rcpp_1.0.8
## [40] munsell_0.5.0       fansi_1.0.0         lifecycle_1.0.1
## [43] pROC_1.18.0         stringi_1.7.6       yaml_2.2.1
## [46] MASS_7.3-54         plyr_1.8.6          recipes_0.1.17
## [49] grid_4.1.2          parallel_4.1.2      listenv_0.8.0
## [52] crayon_1.4.2        haven_2.4.3         splines_4.1.2
## [55] hms_1.1.1           knitr_1.37          pillar_1.6.4
## [58] stats4_4.1.2        future.apply_1.8.1  reshape2_1.4.4
## [61] codetools_0.2-18    reprex_2.0.1        glue_1.6.0
## [64] evaluate_0.14       data.table_1.14.2   modelr_0.1.8
## [67] vctrs_0.3.8         tzdb_0.2.0          foreach_1.5.1
## [70] cellranger_1.1.0    gtable_0.3.0        future_1.23.0
## [73] assertthat_0.2.1    xfun_0.29           gower_0.2.2
## [76] prodlim_2019.11.13  broom_0.7.11        class_7.3-19
## [79] survival_3.2-13     timeDate_3043.102   iterators_1.0.13
## [82] lava_1.6.10         globals_0.14.0      ellipsis_0.3.2
## [85] ipred_0.9-12
```