ISL Lab-4 & 5

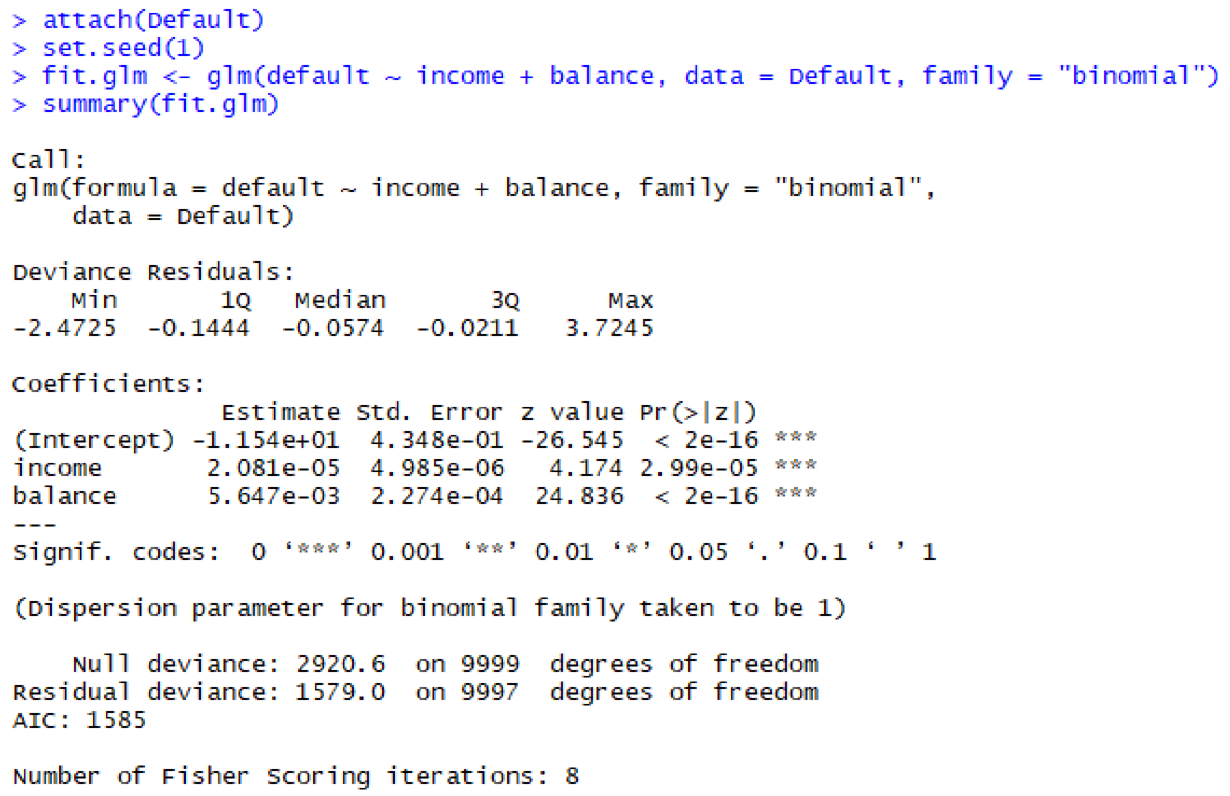
Linga Sai Yuvesh Venketa Kotiala

16242113

2. In Chapter 4, we used logistic regression to predict the probability of default using income and balance on the Default data set. We will now estimate the test error of this logistic regression model using the validation set approach. Do not forget to set a random seed before beginning your analysis.

(a) (10 points) Fit a logistic regression model that uses income and balance to predict default.

Ans.

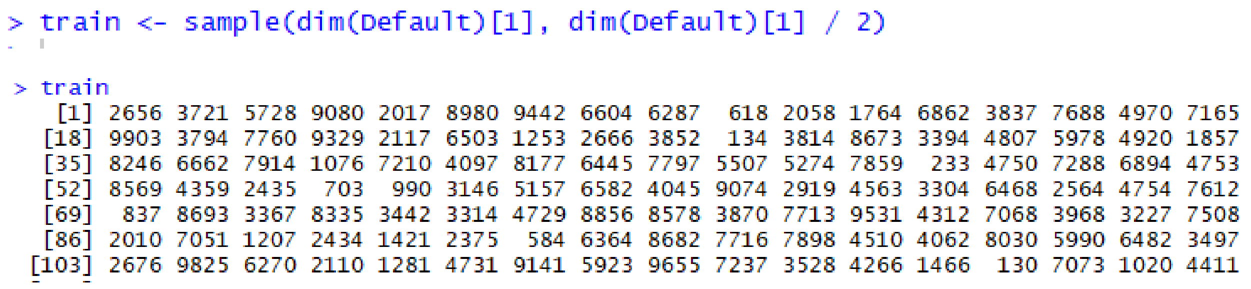


```
> attach(Default)
> set.seed(1)
> fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial")
> summary(fit.glm)

Call:
glm(formula = default ~ income + balance, family = "binomial",
    data = Default)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4725  -0.1444  -0.0574  -0.0211   3.7245

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1579.0  on 9997  degrees of freedom
AIC: 1585

Number of Fisher Scoring iterations: 8
```

(b) (10 points total) Using the validation set approach, estimate the test error of this model. In order to do this, you must perform the following steps:

i. (2.5 points) Split the sample set into a training set and a validation set.

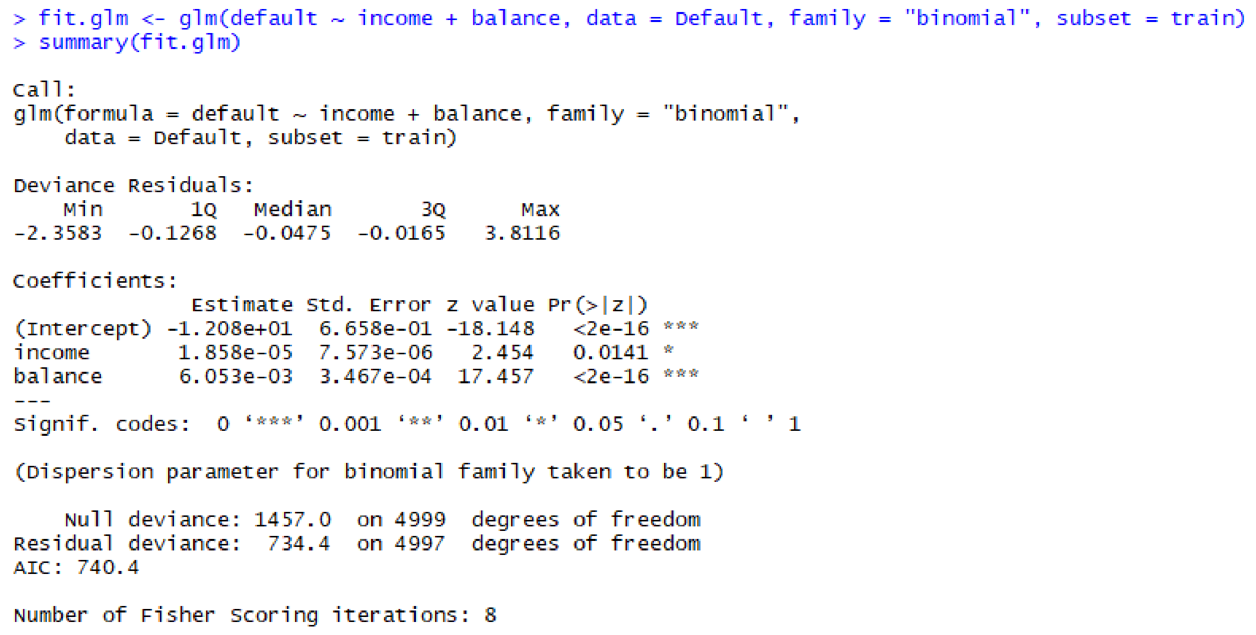
Ans.



```
> train <- sample(dim(Default)[1], dim(Default)[1] / 2)

> train
   [1] 2656 3721 5728 9080 2017 8980 9442 6604 6287  618 2058 1764 6862 3837 7688 4970 7165
  [18] 9903 3794 7760 9329 2117 6503 1253 2666 3852  134 3814 8673 3394 4807 5978 4920 1857
  [35] 8246 6662 7914 1076 7210 4097 8177 6445 7797 5507 5274 7859  233 4750 7288 6894 4753
  [52] 8569 4359 2435  703  990 3146 5157 6582 4045 9074 2919 4563 3304 6468 2564 4754 7612
  [69]  837 8693 3367 8335 3442 3314 4729 8856 8578 3870 7713 9531 4312 7068 3968 3227 7508
  [86] 2010 7051 1207 2434 1421 2375  584 6364 8682 7716 7898 4510 4062 8030 5990 6482 3497
 [103] 2676 9825 6270 2110 1281 4731 9141 5923 9655 7237 3528 4266 1466  130 7073 1020 4411
```

ii. (2.5 points) Fit a multiple logistic regression model using only the training observations.

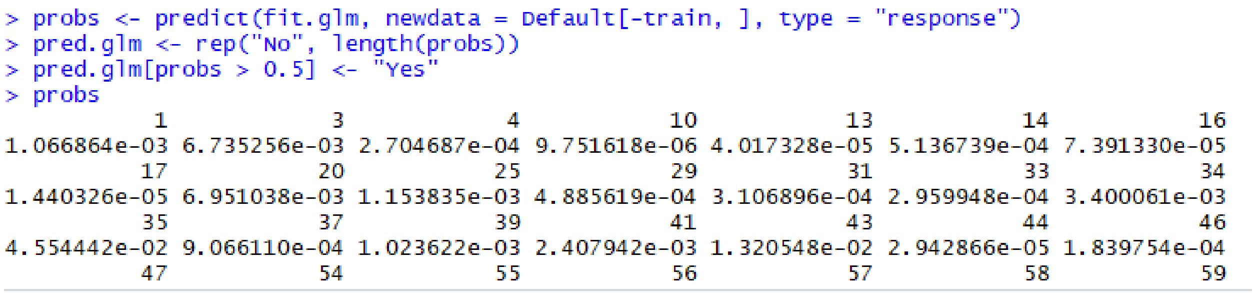
Ans.



```
> fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
> summary(fit.glm)

Call:
glm(formula = default ~ income + balance, family = "binomial",
    data = Default, subset = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3583  -0.1268  -0.0475  -0.0165   3.8116

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.208e+01  6.658e-01 -18.148   <2e-16 ***
income       1.858e-05  7.573e-06   2.454   0.0141 *
balance      6.053e-03  3.467e-04  17.457   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1457.0  on 4999  degrees of freedom
Residual deviance:  734.4  on 4997  degrees of freedom
AIC: 740.4

Number of Fisher Scoring iterations: 8
```

iii. (2.5 points) Obtain a prediction of default status for each individual in the validation set by computing the posterior probability of default for that individual, and classifying the individual to the default category if the posterior probability is greater than 0.5.

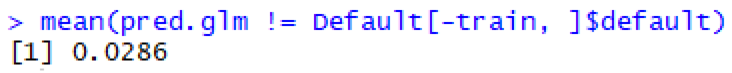
Ans.



```
> probs <- predict(fit.glm, newdata = Default[-train, ], type = "response")
> pred.glm <- rep("No", length(probs))
> pred.glm[probs > 0.5] <- "Yes"
> probs
           1            3            4           10           13           14           16
1.066864e-03 6.735256e-03 2.704687e-04 9.751618e-06 4.017328e-05 5.136739e-04 7.391330e-05
          17           20           25           29           31           33           34
1.440326e-05 6.951038e-03 1.153835e-03 4.885619e-04 3.106896e-04 2.959948e-04 3.400061e-03
          35           37           39           41           43           44           46
4.554442e-02 9.066110e-04 1.023622e-03 2.407942e-03 1.320548e-02 2.942866e-05 1.839754e-04
          47           54           55           56           57           58           59
```

iv. (2.5 points) Compute the validation set error, which is the fraction of the observations in the validation set that are misclassified.

Ans.

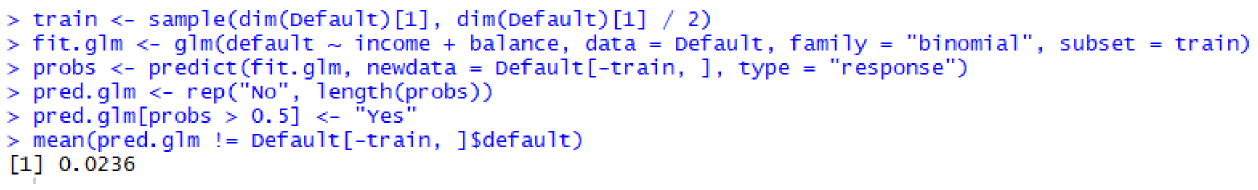


```
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0286
```

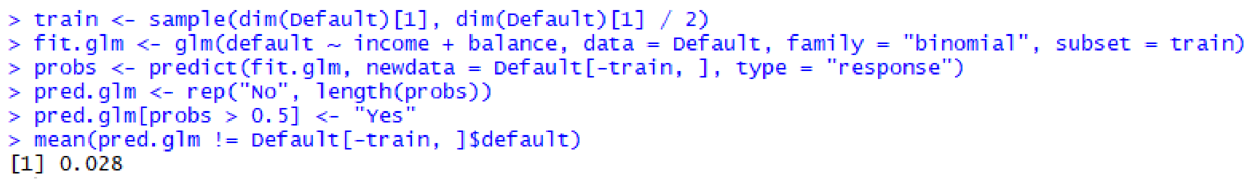
We have a 2.86% test error rate with the validation set approach.

(c) Repeat the process in (b) three times, using three different splits of the observations into a training set and a validation set. Comment on the results obtained.

Ans.



```
> train <- sample(dim(Default)[1], dim(Default)[1] / 2)
> fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
> probs <- predict(fit.glm, newdata = Default[-train, ], type = "response")
> pred.glm <- rep("No", length(probs))
> pred.glm[probs > 0.5] <- "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0236
```

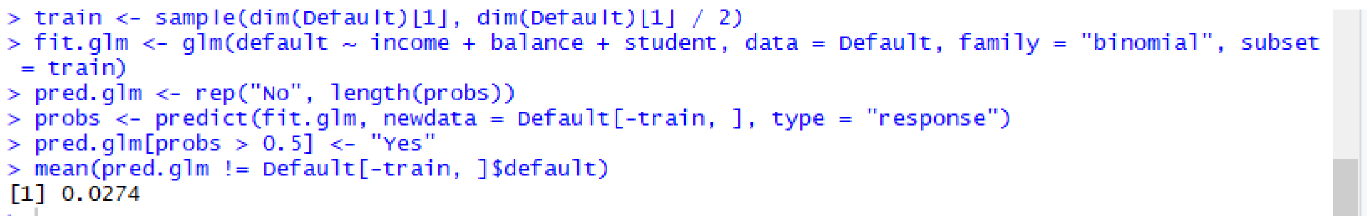


```
> train <- sample(dim(Default)[1], dim(Default)[1] / 2)
> fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
> probs <- predict(fit.glm, newdata = Default[-train, ], type = "response")
> pred.glm <- rep("No", length(probs))
> pred.glm[probs > 0.5] <- "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.028
```

We see that the validation estimate of the test error rate can be variable, depending on precisely which observations are included in the training set and which observations are included in the validation set.

(d) Now consider a logistic regression model that predicts the probability of “default” using “income”, “balance”, and a dummy variable for “student”. Estimate the test error for this model using the validation set approach. Comment on whether including a dummy variable for “student” leads to a reduction in the test error rate.

Ans.



```
> train <- sample(dim(Default)[1], dim(Default)[1] / 2)
> fit.glm <- glm(default ~ income + balance + student, data = Default, family = "binomial", subset
 = train)
> pred.glm <- rep("No", length(probs))
> probs <- predict(fit.glm, newdata = Default[-train, ], type = "response")
> pred.glm[probs > 0.5] <- "Yes"
> mean(pred.glm != Default[-train, ]$default)
[1] 0.0274
```

3. (40 points) We continue to consider the use of a logistic regression model to predict the probability of default using income and balance on the Default data set. In particular, we will now compute estimates for the standard errors of the income and balance logistic regression coefficients in two different ways: (1) using the bootstrap, and (2) using the standard formula for computing the standard errors in the glm() function. Do not forget to set a random seed before beginning your analysis.

(a) (10 points) Using the summary () and glm() functions, determine the estimated standard errors for the coefficients associated with income and balance in a multiple logistic regression model that uses both predictors.

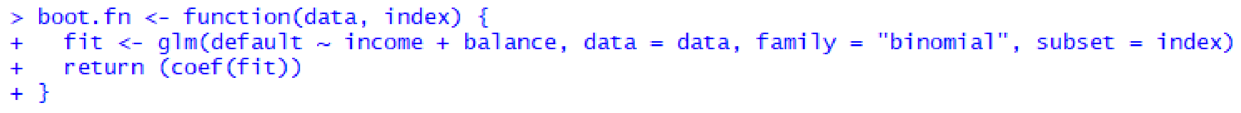
Ans.

```
> set.seed(1)
> attach(Default)
The following objects are masked from Default (pos = 3):

    balance, default, income, student




> fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial")
> summary(fit.glm)

Call:
glm(formula = default ~ income + balance, family = "binomial",
    data = Default)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4725  -0.1444  -0.0574  -0.0211   3.7245

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1579.0  on 9997  degrees of freedom
AIC: 1585

Number of Fisher Scoring iterations: 8
```

The glm() estimates of the standard errors for the coefficients β0, β1 and β2 are respectively 0.4347564, 4.985167210^{-6} and 2.273731410^{-4}.

(b) (10 points) Write a function, boot.fn(), that takes as input the Default data set as well as an index of the observations, and that outputs the coefficient estimates for income and balance in the multiple logistic regression model.

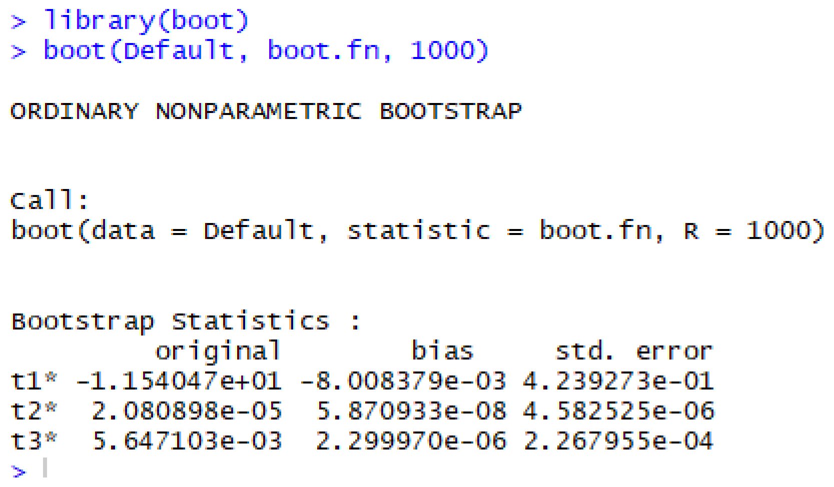
Ans.



```
> boot.fn <- function(data, index) {
+   fit <- glm(default ~ income + balance, data = data, family = "binomial", subset = index)
+   return (coef(fit))
+ }
```

(c) Use the boot() function together with your boot.fn() function to estimate the standard errors of the logistic regression coefficients for "income" and "balance".

Ans.



```
> library(boot)
> boot(Default, boot.fn, 1000)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Default, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
         original        bias     std. error
t1* -1.154047e+01 -8.008379e-03 4.239273e-01
t2*  2.080898e-05  5.870933e-08 4.582525e-06
t3*  5.647103e-03  2.299970e-06 2.267955e-04
> 
```

(d) Comment on the estimated standard errors obtained using the glm() function and using your bootstrap function.

Ans.

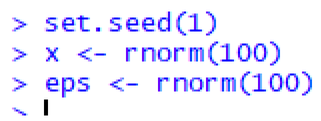
The estimated standard errors obtained by the two methods are close.

5. (30 points total) In this exercise, we will generate simulated data, and will then use this data to perform

best subset selection.

(a) (5 points) Use the rnorm() function to generate a predictor X of length n = 100, as well as a

noise vector ϵ of length n = 100



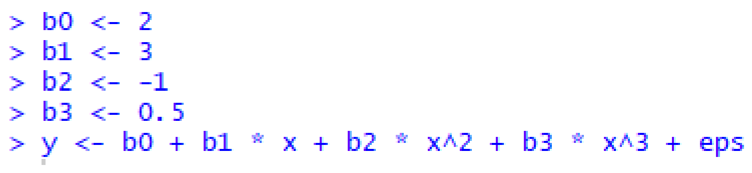
```
> set.seed(1)
> x <- rnorm(100)
> eps <- rnorm(100)
>
```

(b) (5 points) Generate a response vector Y of length n = 100 according to the model

$$Y = \beta 0 + \beta 1X + \beta 2X2 + \beta 3X3 + \epsilon,$$

where β0, β1, β2, and β3 are constants of your choice.

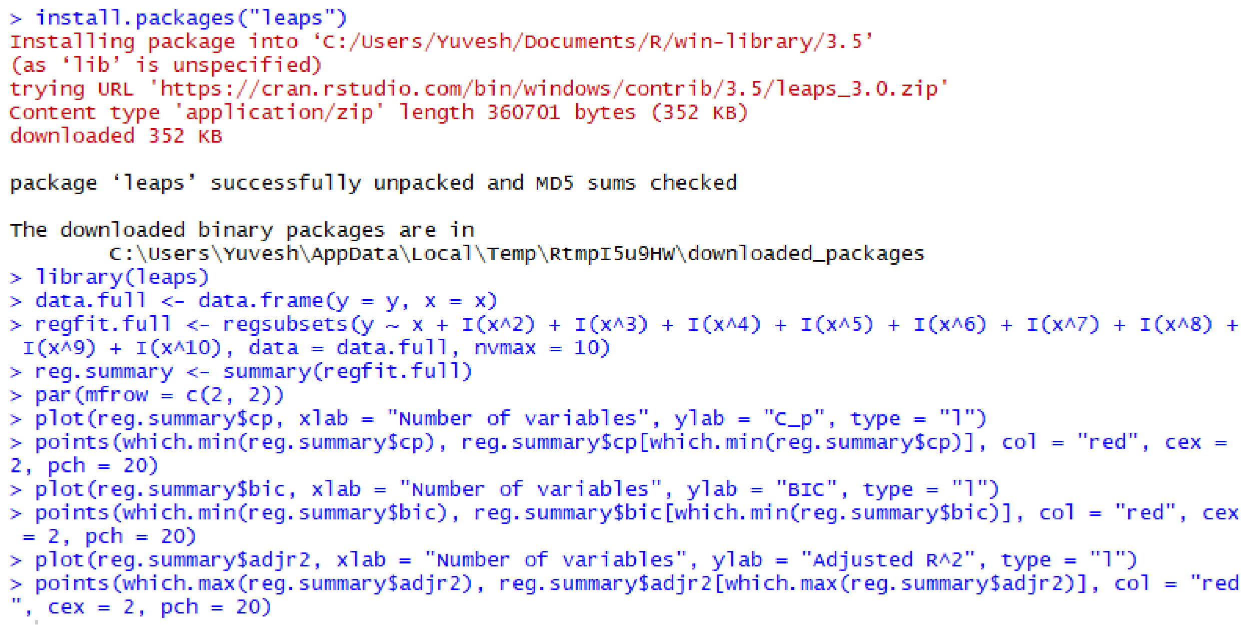
Ans.



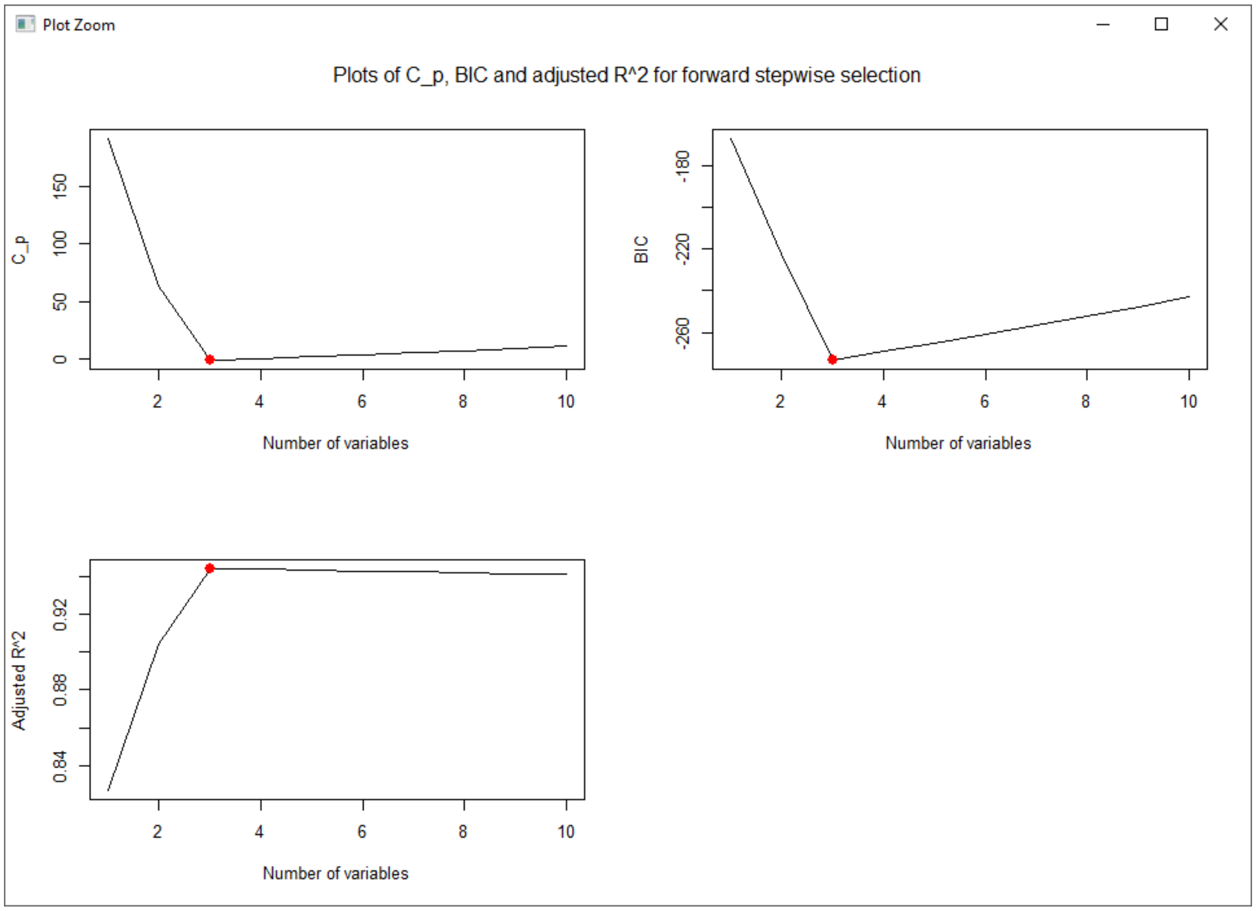
```
> b0 <- 2
> b1 <- 3
> b2 <- -1
> b3 <- 0.5
> y <- b0 + b1 * x + b2 * x^2 + b3 * x^3 + eps
```

(c) (5 points) Use the regsubsets() function to perform best subset selection in order to choose the best model containing the predictors X, X^2 , . . . , X^{10}. What is the best model obtained according to C_p, BIC, and adjusted R^2? Show some plots to provide evidence for your answer, and report the coefficients of the best model obtained. Note you will need to use the data.frame() function to create a single data set containing both X and Y .

Ans.



```
> install.packages("leaps")
Installing package into ‘C:/Users/Yuvesh/Documents/R/win-library/3.5’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.5/leaps_3.0.zip'
Content type 'application/zip' length 360701 bytes (352 KB)
downloaded 352 KB

package ‘leaps’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Yuvesh\AppData\Local\Temp\RtmpI5u9Hw\downloaded_packages
> library(leaps)
> data.full <- data.frame(y = y, x = x)
> regfit.full <- regsubsets(y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) + I(x^6) + I(x^7) + I(x^8) +
 I(x^9) + I(x^10), data = data.full, nvmax = 10)
> reg.summary <- summary(regfit.full)
> par(mfrow = c(2, 2))
> plot(reg.summary$cp, xlab = "Number of variables", ylab = "C_p", type = "l")
> points(which.min(reg.summary$cp), reg.summary$cp[which.min(reg.summary$cp)], col = "red", cex =
2, pch = 20)
> plot(reg.summary$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
> points(which.min(reg.summary$bic), reg.summary$bic[which.min(reg.summary$bic)], col = "red", cex
 = 2, pch = 20)
> plot(reg.summary$adjr2, xlab = "Number of variables", ylab = "Adjusted R^2", type = "l")
> points(which.max(reg.summary$adjr2), reg.summary$adjr2[which.max(reg.summary$adjr2)], col = "red
", cex = 2, pch = 20)
```



We find that, with Cp we pick the 3-variables model, with BIC we pick the 3-variables model, and with adjusted R^2 we pick the 3-variables model.

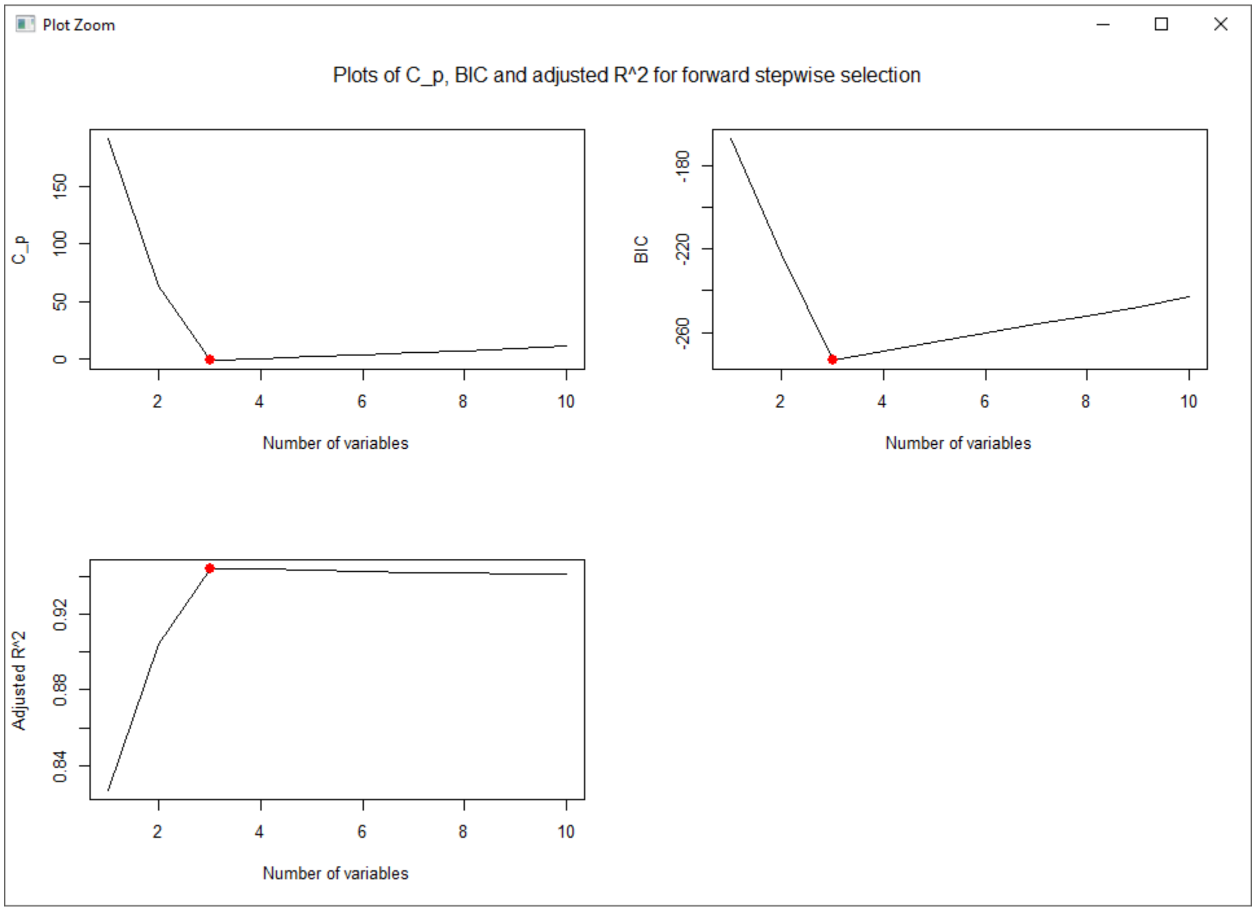
```
> coef(regfit.full, which.max(reg.summary$adjr2))
(Intercept)           x      I(x^2)      I(x^5) 
 2.07219472  3.44514720 -1.15676236  0.09022577 
```

(d) Repeat (c), using forward stepwise selection and also using backwards stepwise selection. How does your answer compare to the results in (c) ?

Ans.

We begin with forward stepwise selection.

```
> regfit.fwd <- regsubsets(y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) + I(x^6) + I(x^7) + I(x^8) +
I(x^9) + I(x^10), data = data.full, nvmax = 10, method = "forward")
> reg.summary.fwd <- summary(regfit.fwd)
> par(mfrow = c(2, 2))
> plot(reg.summary.fwd$cp, xlab = "Number of variables", ylab = "C_p", type = "l")
> points(which.min(reg.summary.fwd$cp), reg.summary.fwd$cp[which.min(reg.summary.fwd$cp)], col = "
red", cex = 2, pch = 20)
> plot(reg.summary.fwd$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
> points(which.min(reg.summary.fwd$bic), reg.summary.fwd$bic[which.min(reg.summary.fwd$bic)], col
= "red", cex = 2, pch = 20)
> plot(reg.summary.fwd$adjr2, xlab = "Number of variables", ylab = "Adjusted R^2", type = "l")
> points(which.max(reg.summary.fwd$adjr2), reg.summary.fwd$adjr2[which.max(reg.summary.fwd$adjr2)]
, col = "red", cex = 2, pch = 20)
> mtext("Plots of C_p, BIC and adjusted R^2 for forward stepwise selection", side = 3, line = -2,
outer = TRUE)
>
```

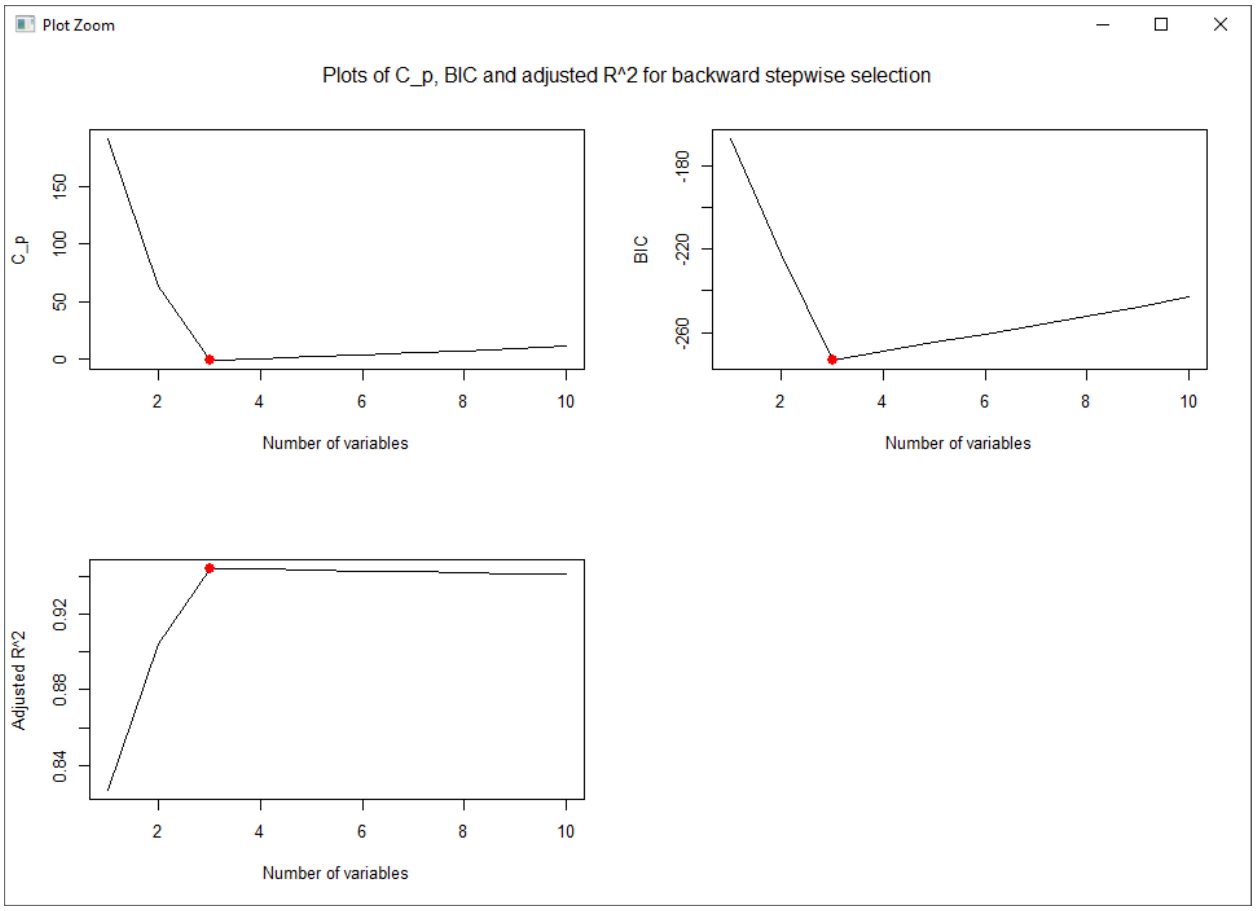


We find that, for forward stepwise selection, with Cp we pick the 3-variables model, with BIC we pick the 3-variables model, and with adjusted R^2 we pick the 3-variables model.

```
> coef(regfit.fwd, which.max(reg.summary.fwd$adjr2))
(Intercept)           x      I(x^2)      I(x^5) 
 2.07219472  3.44514720 -1.15676236  0.09022577 
>
```

Next, we proceed with backward stepwise selection.

```
> regfit.bwd <- regsubsets(y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) + I(x^6) + I(x^7) + I(x^8) +
I(x^9) + I(x^10), data = data.full, nvmax = 10, method = "backward")
> reg.summary.bwd <- summary(regfit.bwd)
> par(mfrow = c(2, 2))
> plot(reg.summary.bwd$cp, xlab = "Number of variables", ylab = "C_p", type = "l")
> points(which.min(reg.summary.bwd$cp), reg.summary.bwd$cp[which.min(reg.summary.bwd$cp)], col = "
red", cex = 2, pch = 20)
> plot(reg.summary.bwd$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
> points(which.min(reg.summary.bwd$bic), reg.summary.bwd$bic[which.min(reg.summary.bwd$bic)], col
= "red", cex = 2, pch = 20)
> plot(reg.summary.bwd$adjr2, xlab = "Number of variables", ylab = "Adjusted R^2", type = "l")
> points(which.max(reg.summary.bwd$adjr2), reg.summary.bwd$adjr2[which.max(reg.summary.bwd$adjr2)]
, col = "red", cex = 2, pch = 20)
> mtext("Plots of C_p, BIC and adjusted R^2 for backward stepwise selection", side = 3, line = -2,
 outer = TRUE)
>
```

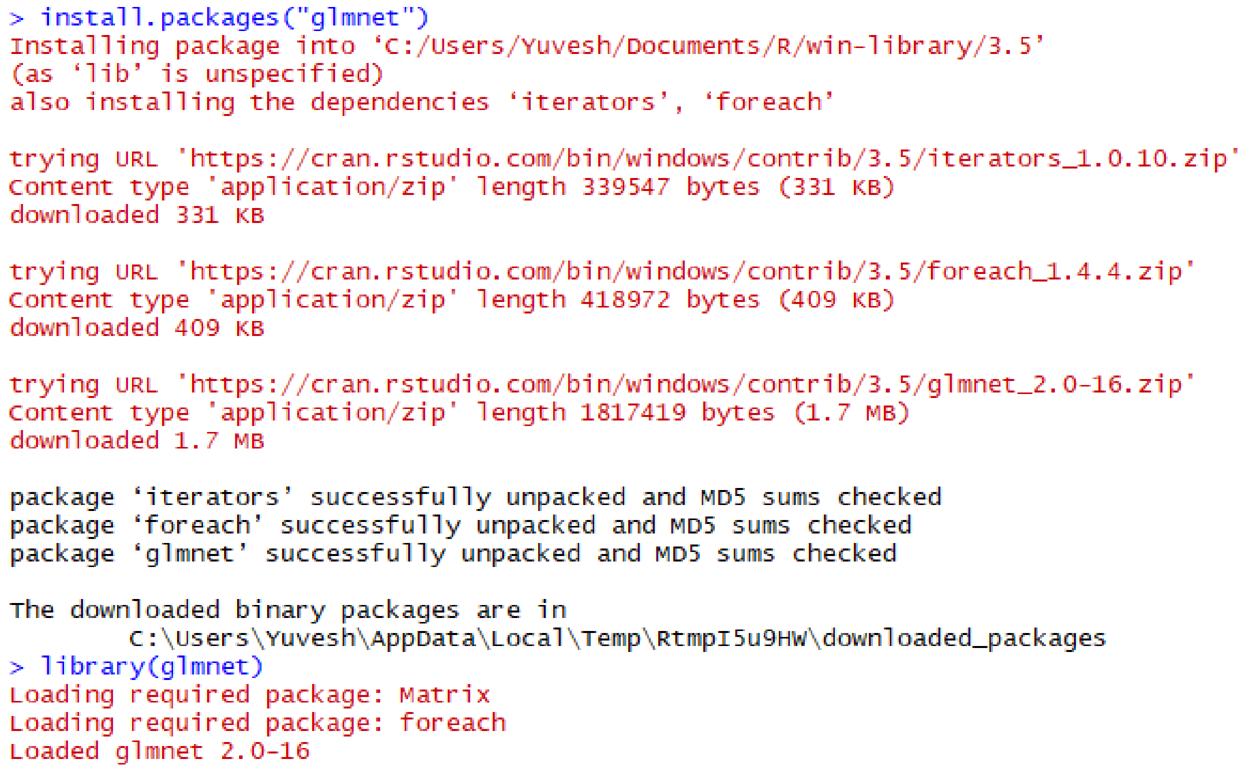


We find that, for backward stepwise selection, with Cp we pick the 3-variables model, with BIC we pick the 3-variables model, and with adjusted R^2 we pick the 3-variables model.

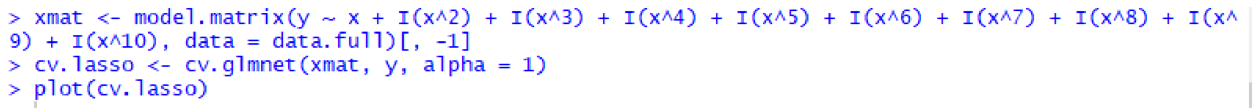
```
> coef(regfit.bwd, which.max(reg.summary.bwd$adjr2))
(Intercept)           x      I(x^2)      I(x^5) 
 2.07219472  3.44514720 -1.15676236  0.09022577 
>
```

Here forward stepwise, backward stepwise and best subset all select the three variables model with X, X2 and X5.

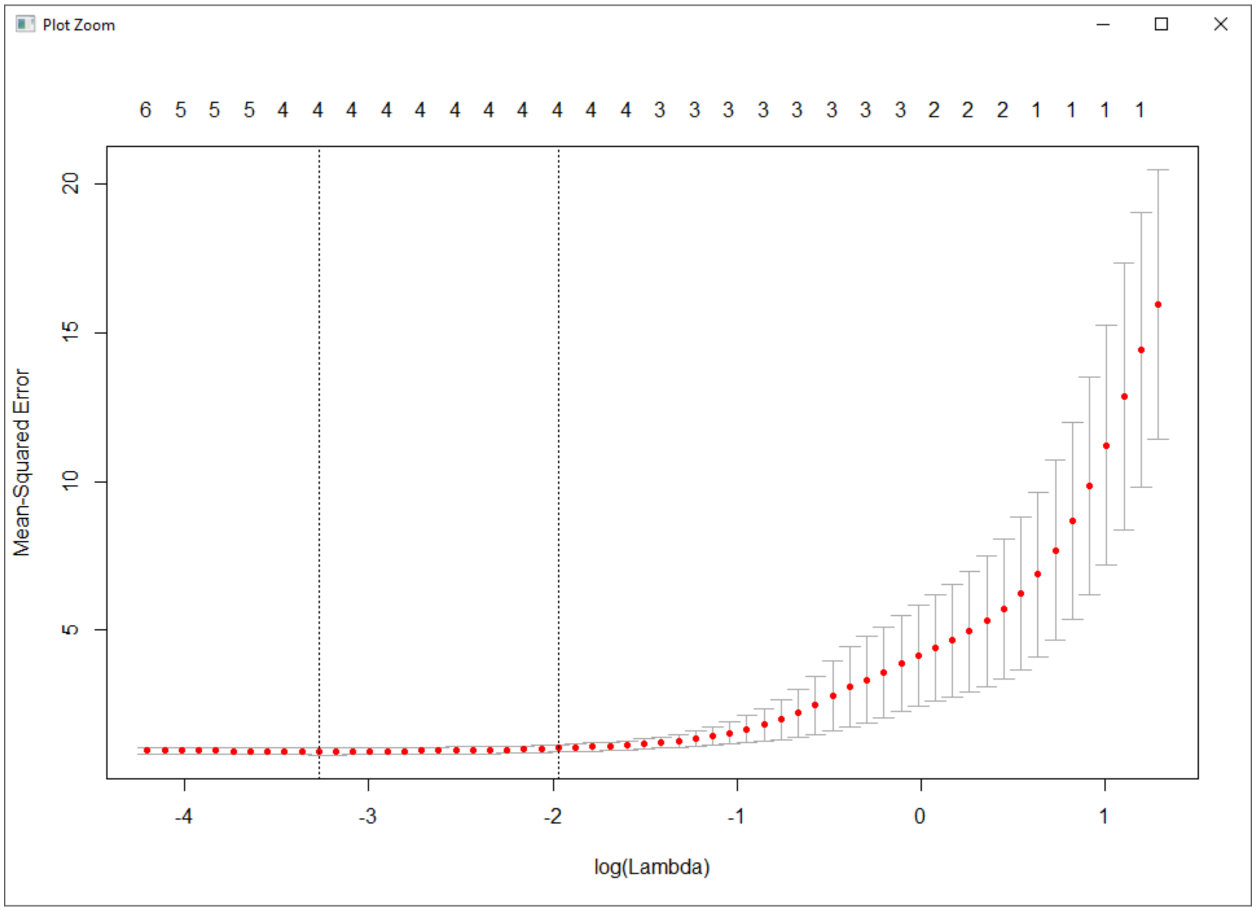
(e) Now fit a lasso model to the simulated data, again using X,X2,⋯,X10 as predictors. Use cross-validation to select the optimal value of λ. Create plots of the cross-validation error as a function of λ. Report the resulting coefficient estimates, and discuss the results obtained.



```
> install.packages("glmnet")
Installing package into ‘C:/Users/Yuvesh/Documents/R/win-library/3.5’
(as ‘lib’ is unspecified)
also installing the dependencies ‘iterators’, ‘foreach’

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.5/iterators_1.0.10.zip'
Content type 'application/zip' length 339547 bytes (331 KB)
downloaded 331 KB

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.5/foreach_1.4.4.zip'
Content type 'application/zip' length 418972 bytes (409 KB)
downloaded 409 KB

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.5/glmnet_2.0-16.zip'
Content type 'application/zip' length 1817419 bytes (1.7 MB)
downloaded 1.7 MB

package ‘iterators’ successfully unpacked and MD5 sums checked
package ‘foreach’ successfully unpacked and MD5 sums checked
package ‘glmnet’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Yuvesh\AppData\Local\Temp\RtmpI5u9HW\downloaded_packages
> library(glmnet)
Loading required package: Matrix
Loading required package: foreach
Loaded glmnet 2.0-16
```



```
> xmat <- model.matrix(y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) + I(x^6) + I(x^7) + I(x^8) + I(x^
9) + I(x^10), data = data.full)[, -1]
> cv.lasso <- cv.glmnet(xmat, y, alpha = 1)
> plot(cv.lasso)
```



```
> bestlam <- cv.lasso$lambda.min
> bestlam
[1] 0.03795616
```

Now we refit our lasso model using the value λ= 0.0379562 chosen by cross-validation.

```
> fit.lasso <- glmnet(xmat, y, alpha = 1)
> predict(fit.lasso, s = bestlam, type = "coefficients")[1:11, ]
(Intercept)           x      I(x^2)      I(x^3)      I(x^4)      I(x^5)      I(x^6) 
 2.04091434  3.28373245 -1.10646862  0.14042235  0.00000000  0.06399454  0.00000000 
     I(x^7)      I(x^8)      I(x^9)     I(x^10) 
 0.00000000  0.00000000  0.00000000  0.00000000 
```

The lasso method picks X, X^2, X^3 and X^5 as variables for the model.

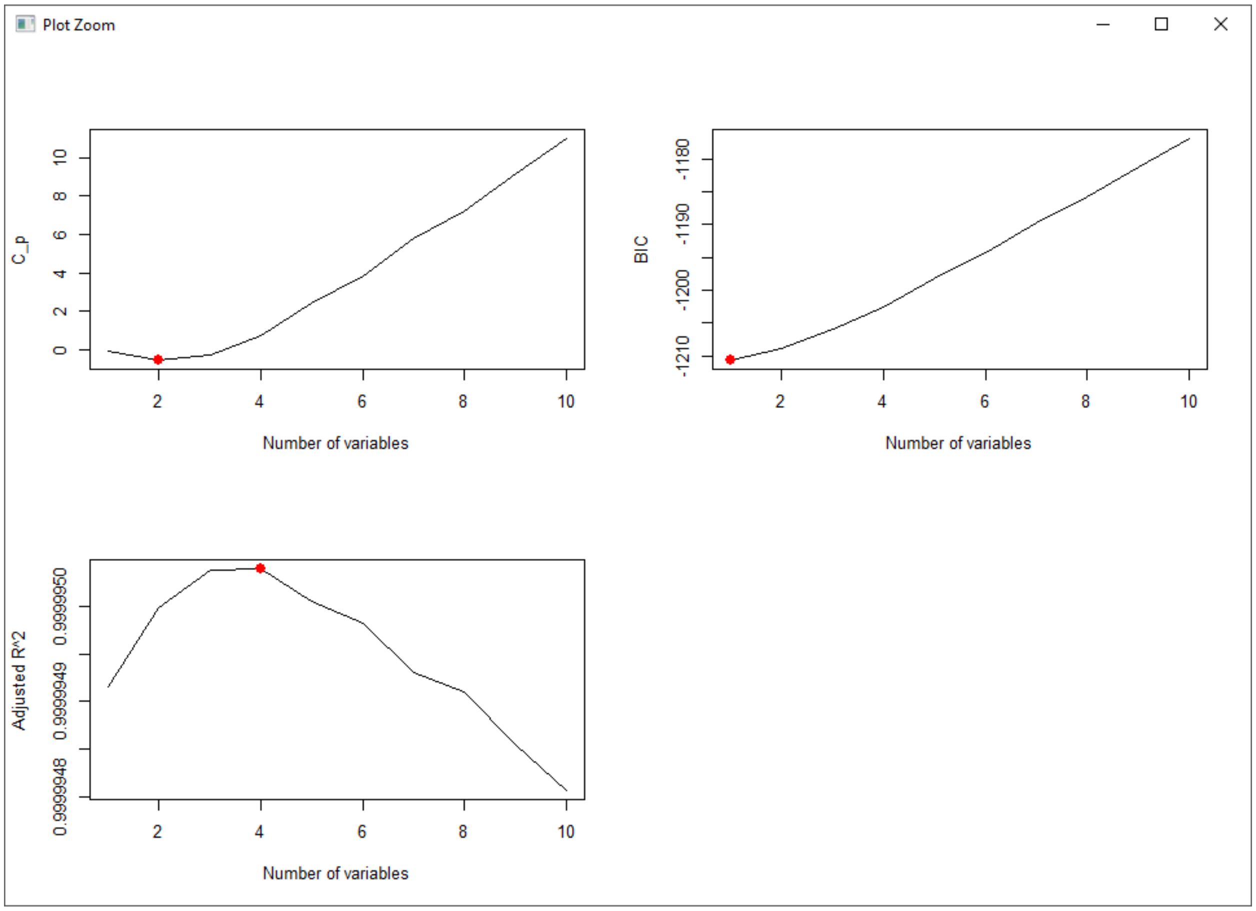
(f) Now generate a response vector Y according to the model

$$Y=\beta_0+\beta_7X_7+\varepsilon,$$

and perform best subset selection and the lasso. Discuss the results obtained. We begin with best subset selection.

Ans.

We begin with best subset selection.



We find that, with Cp we pick the 2-variables model, with BIC we pick the 1-variables model, and with adjusted R^2 we pick the 4-variables model.

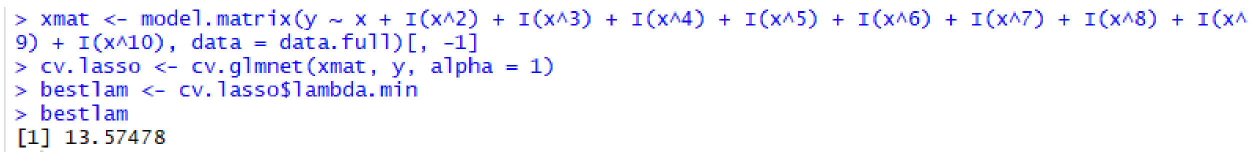
```
> coef(regfit.full, 1)
(Intercept)      I(x^7)
    1.95894     7.00077
```

```
> coef(regfit.full, 2)
(Intercept)      I(x^2)      I(x^7)
  2.0704904  -0.1417084   7.0015552
```

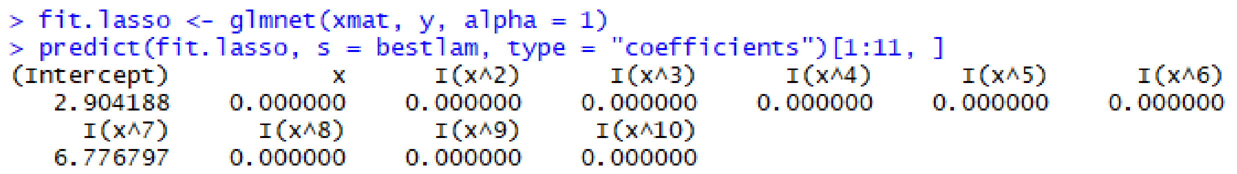
```
> coef(regfit.full, 4)
(Intercept)           x      I(x^2)      I(x^3)      I(x^7)
  2.0762524   0.2914016  -0.1617671  -0.2526527   7.0091338
```

Here best subset selection with BIC picks the most accurate 1-variable model with matching coefficients.

Now we proceed with the lasso.



```
> xmat <- model.matrix(y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) + I(x^6) + I(x^7) + I(x^8) + I(x^
9) + I(x^10), data = data.full)[, -1]
> cv.lasso <- cv.glmnet(xmat, y, alpha = 1)
> bestlam <- cv.lasso$lambda.min
> bestlam
[1] 13.57478
```



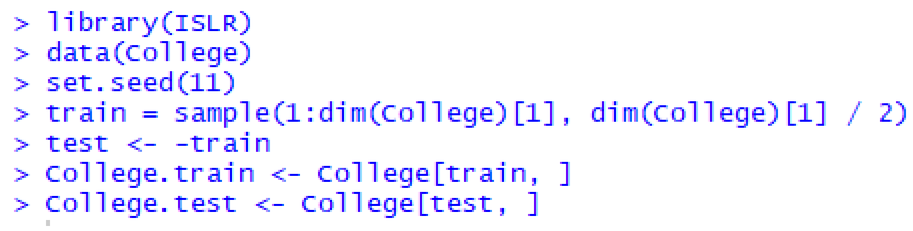
```
> fit.lasso <- glmnet(xmat, y, alpha = 1)
> predict(fit.lasso, s = bestlam, type = "coefficients")[1:11, ]
(Intercept)           x      I(x^2)      I(x^3)      I(x^4)      I(x^5)      I(x^6) 
   2.904188    0.000000    0.000000    0.000000    0.000000    0.000000    0.000000 
     I(x^7)      I(x^8)      I(x^9)     I(x^10) 
   6.776797    0.000000    0.000000    0.000000 
```

Here the lasso also picks the most accurate 1-variable model, but the intercept is quite off.

6. (35 points total) In this exercise, we will predict the number of applications received using the other variables in the College data set.

(a) (5 points) Split the data set into a training set and a test set.

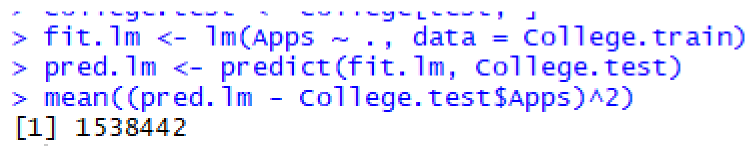
Ans.



```
> library(ISLR)
> data(College)
> set.seed(11)
> train = sample(1:dim(College)[1], dim(College)[1] / 2)
> test <- -train
> College.train <- College[train, ]
> College.test <- College[test, ]
```

(b) (5 points) Fit a linear model using least squares on the training set, and report the test error Obtained.

Ans.

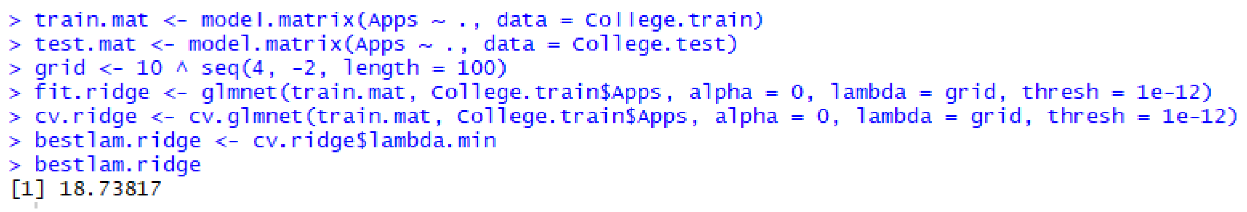


```
> fit.lm <- lm(Apps ~ ., data = College.train)
> pred.lm <- predict(fit.lm, College.test)
> mean((pred.lm - College.test$Apps)^2)
[1] 1538442
```

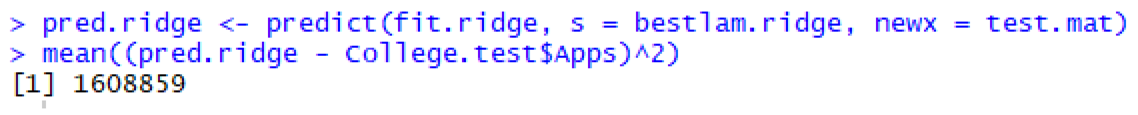
The test MSE is 1.538442210^{6}.

(c) (5 points) Fit a ridge regression model on the training set, with λ chosen by cross-validation. Report the test error obtained.

Ans.



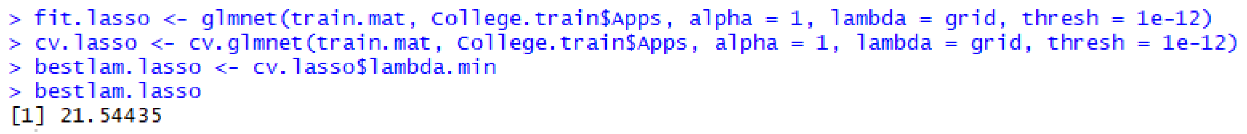
```
> train.mat <- model.matrix(Apps ~ ., data = College.train)
> test.mat <- model.matrix(Apps ~ ., data = College.test)
> grid <- 10 ^ seq(4, -2, length = 100)
> fit.ridge <- glmnet(train.mat, College.train$Apps, alpha = 0, lambda = grid, thresh = 1e-12)
> cv.ridge <- cv.glmnet(train.mat, College.train$Apps, alpha = 0, lambda = grid, thresh = 1e-12)
> bestlam.ridge <- cv.ridge$lambda.min
> bestlam.ridge
[1] 18.73817
```



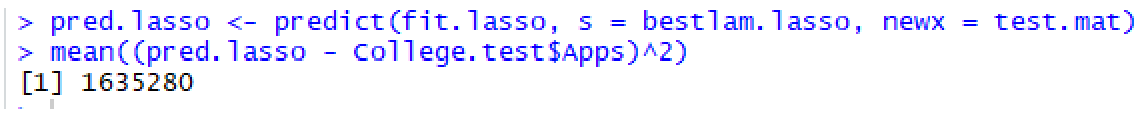
```
> pred.ridge <- predict(fit.ridge, s = bestlam.ridge, newx = test.mat)
> mean((pred.ridge - College.test$Apps)^2)
[1] 1608859
```

The test MSE is higher for ridge regression than for least squares.

(d) (5 points) Fit a lasso model on the training set, with λ chosen by crossvalidation. Report the test error obtained, along with the number f non-zero coefficient estimates.
Ans.

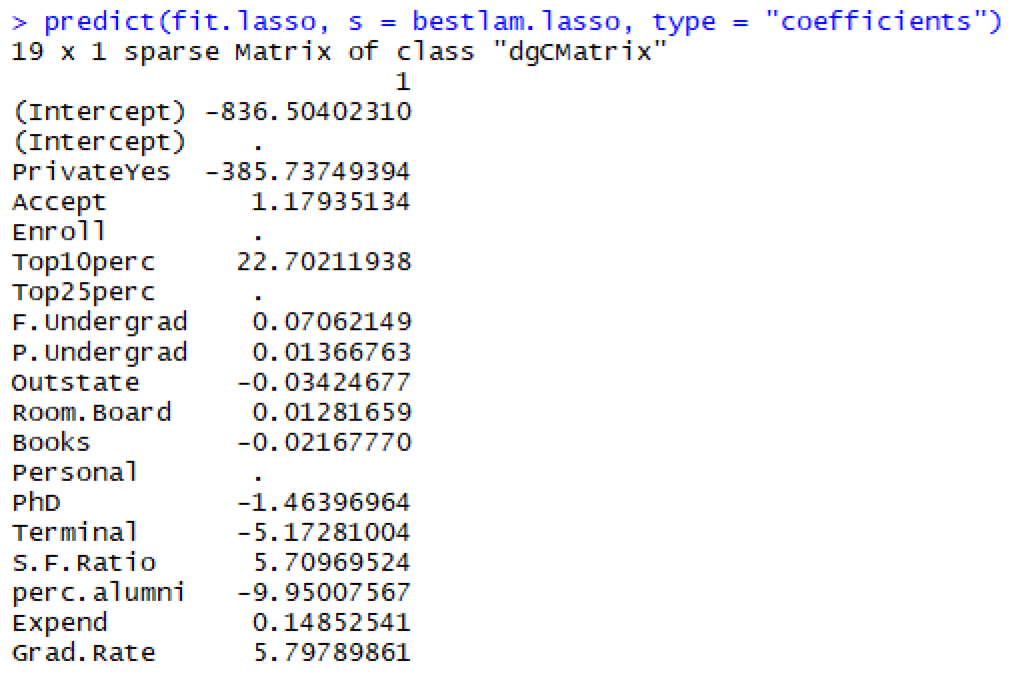


```
> fit.lasso <- glmnet(train.mat, College.train$Apps, alpha = 1, lambda = grid, thresh = 1e-12)
> cv.lasso <- cv.glmnet(train.mat, College.train$Apps, alpha = 1, lambda = grid, thresh = 1e-12)
> bestlam.lasso <- cv.lasso$lambda.min
> bestlam.lasso
[1] 21.54435
```



```
> pred.lasso <- predict(fit.lasso, s = bestlam.lasso, newx = test.mat)
> mean((pred.lasso - College.test$Apps)^2)
[1] 1635280
```

The test MSE is also higher for ridge regression than for least squares.

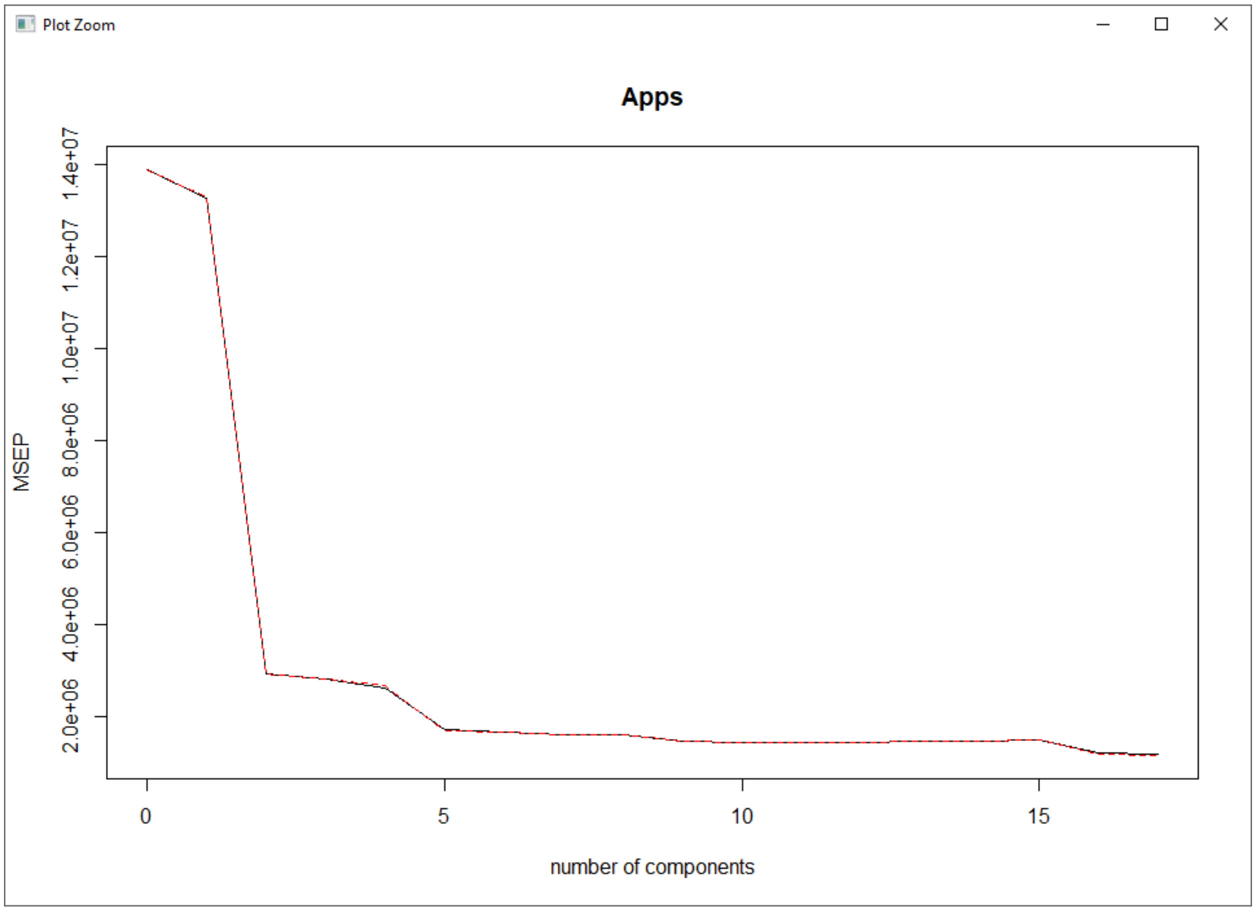


```
> predict(fit.lasso, s = bestlam.lasso, type = "coefficients")
19 x 1 sparse Matrix of class "dgCMatrix"
                        1
(Intercept) -836.50402310
(Intercept)    .
PrivateYes  -385.73749394
Accept         1.17935134
Enroll         .
Top10perc     22.70211938
Top25perc      .
F.Undergrad    0.07062149
P.Undergrad    0.01366763
Outstate      -0.03424677
Room.Board     0.01281659
Books         -0.02167770
Personal       .
PhD           -1.46396964
Terminal      -5.17281004
S.F.Ratio      5.70969524
perc.alumni   -9.95007567
Expend         0.14852541
Grad.Rate      5.79789861
```

(e) (5 points) Fit a PCR model on the training set, with M chosen by crossvalidation. Report the test error obtained, along with the value of M selected by cross-validation.

Ans.

```
> install.packages("pls")
Installing package into ‘C:/Users/Yuvesh/Documents/R/win-library/3.5’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.5/pls_2.7-0.zip'
Content type 'application/zip' length 1224890 bytes (1.2 MB)
downloaded 1.2 MB

package ‘pls’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Yuvesh\AppData\Local\Temp\RtmpI5u9HW\downloaded_packages
> library(pls)

Attaching package: ‘pls’

The following object is masked from ‘package:stats’:

    loadings

> fit.pcr <- pcr(Apps ~ ., data = College.train, scale = TRUE, validation = "CV")
> validationplot(fit.pcr, val.type = "MSEP")
```



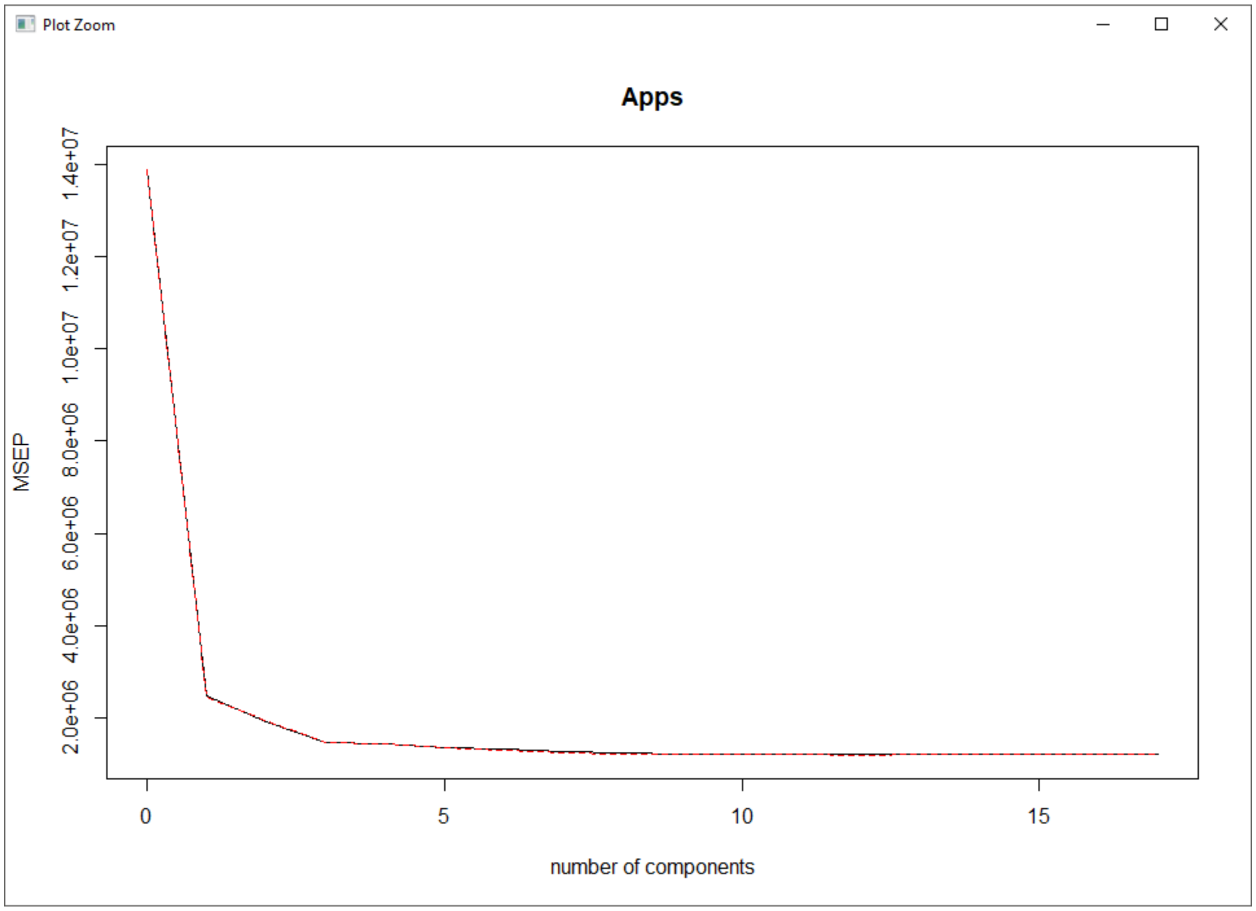
```
> pred.pls <- predict(fit.pcr, College.test, ncomp = 10)
> mean((pred.pls - College.test$Apps)^2)
[1] 3014496
```

The test MSE is also higher for PCR than for least squares.

(f) (5 points) Fit a PLS model on the training set, with M chosen by cross validation. Report the test error obtained, along with the value of M selected by cross-validation

Ans.

```
> fit.pls <- plsr(Apps ~ ., data = College.train, scale = TRUE, validation = "CV")
> validationplot(fit.pls, val.type = "MSEP")
> pred.pls <- predict(fit., College.test, ncomp = 10)
```



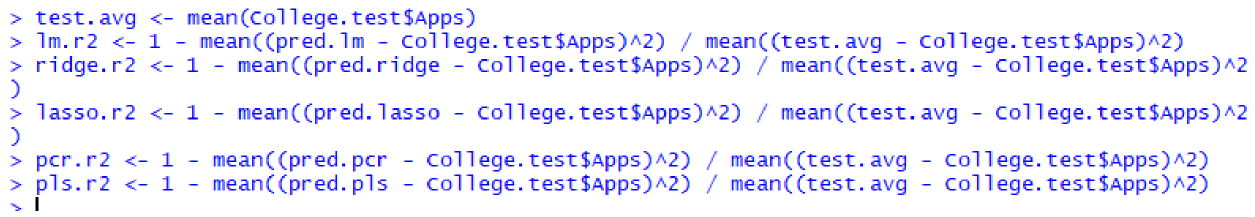
```
> pred.pls <- predict(fit.pls, College.test, ncomp = 10)
> mean((pred.pls - College.test$Apps)^2)
[1] 1508987
```

Here, the test MSE is lower for PLS than for least squares.

(g) (5 points) Comment on the results obtained. How accurately can we predict the number of college applications received? Is there much difference among the test errors resulting from these

five approaches?
Ans.

To compare the results obtained above, we have to compute the test R^2 for all models.



```
> test.avg <- mean(College.test$Apps)
> lm.r2 <- 1 - mean((pred.lm - College.test$Apps)^2) / mean((test.avg - College.test$Apps)^2)
> ridge.r2 <- 1 - mean((pred.ridge - College.test$Apps)^2) / mean((test.avg - College.test$Apps)^2
)
> lasso.r2 <- 1 - mean((pred.lasso - College.test$Apps)^2) / mean((test.avg - College.test$Apps)^2
)
> pcr.r2 <- 1 - mean((pred.pcr - College.test$Apps)^2) / mean((test.avg - College.test$Apps)^2)
> pls.r2 <- 1 - mean((pred.pls - College.test$Apps)^2) / mean((test.avg - College.test$Apps)^2)
> |
```

So the test R2 for least squares is 0.9044281, the test R2 for ridge is 0.9000536, the test R2 for lasso is 0.8984123, the test R2 for pcr is 0.8127319 and the test R2 for pls is 0.9062579. All models, except PCR, predict college applications with high accuracy.