Chapter 5 & 6

Tong Sun

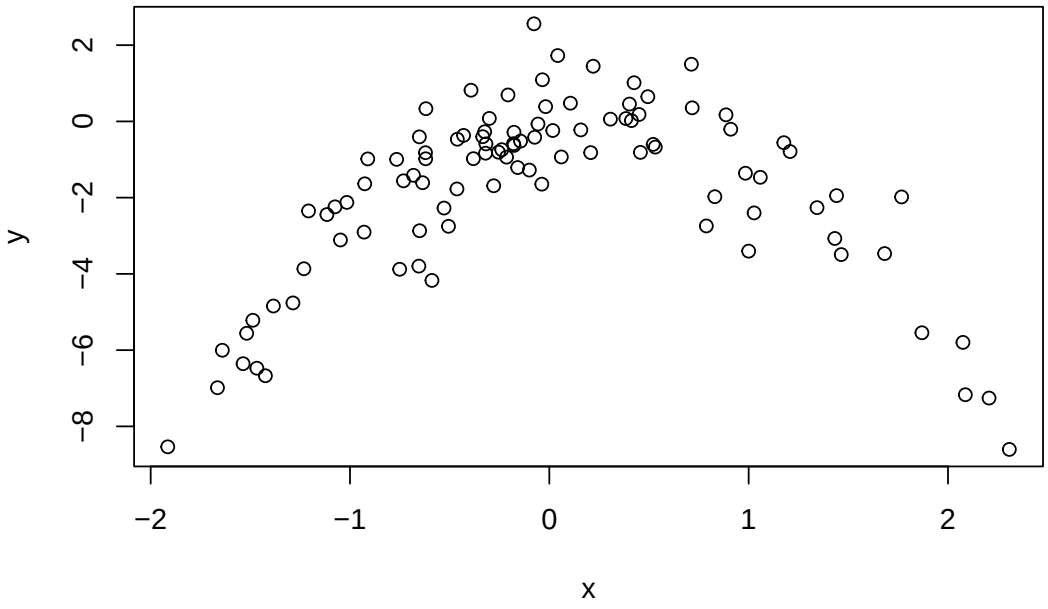
2/2/2022

5.8

We will now perform cross-validation on a simulated data set. ###(a) Generate a simulated data set as follows. In this data set, what is n and what is p? Write out the model used to generate the data in equation form.

Here we have that n = 100 and p = 2, the model used is $Y = X - 2X^2 + \epsilon$.

###(b) Create a scatterplot of X against Y. Comment on what you find.



I find that the data obviously holds a curved relationship.

###(c) Set a random seed, and then compute the LOOCV errors that result from fitting the following four models using least squares: i. $Y = \beta_0 + \beta_1 X + \epsilon$

```
## [1] 5.890979
```

ii. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \epsilon$

```
## [1] 1.086596
```

iii. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon$

```
## [1] 1.102585
```

iv. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \beta_4 X^4 + \epsilon$

```
## [1] 1.114772
```

Note you may find it helpful to use the data.frame() function to create a single data set containing both X and Y.

###(d) Repeat (c) using another random seed, and report your results. Are your results the same as what you got in (c)? Why?

```
## [1] 5.890979
```

```
## [1] 1.086596
```

```
## [1] 1.102585
```

```
## [1] 1.114772
```

I find that both of these two seeds above have the same results because LOOCV evaluates n folds of a single observation. ###(e) Which of the models in (c) had the smallest LOOCV error? Is this what you expected? Explain your answer.

I find that the LOOCV estimates from (c) for the lowest test MSE is "glm.2", which has quadratic predictor in the equation. I also see that in (b) that the relationship between "x" and "y" is quadratic. So I think this result of model fitting is considerable.

###(f) Comment on the statistical significance of the coefficient estimates that results from fitting each of the models in (c) using least squares. Do these results agree with the conclusions drawn based on the cross-validation results?

```
##
## Call:
## glm(formula = y ~ poly(x, 4))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8914  -0.5244   0.0749   0.5932   2.7796
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.8277     0.1041 -17.549   <2e-16 ***
## poly(x, 4)1   2.3164     1.0415   2.224   0.0285 *
## poly(x, 4)2 -21.0586     1.0415 -20.220   <2e-16 ***
## poly(x, 4)3  -0.3048     1.0415  -0.293   0.7704
```

```
## poly(x, 4)4  -0.4926     1.0415  -0.473   0.6373
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 1.084654)
##
##     Null deviance: 552.21  on 99  degrees of freedom
## Residual deviance: 103.04  on 95  degrees of freedom
## AIC: 298.78
##
## Number of Fisher Scoring iterations: 2
```

The p-values show that the linear and quadratic terms are statistically significant , which have the p-values lower than 0.05, and that the cubic and 4th degree terms are not statistically significant. Therefore, this result agrees with the conclusions drawn based on the cross-validation results.

6.2

For parts (a) through (c), indicate which of i through iv is correct. Justify your answer. ###(a) The lasso, relative to least squares, is: i. More flexible and hence will give improved prediction accuracy when its increase in bias is less than its decrease in variance. ii. More flexible and hence will give improved prediction accuracy when its increase in variance is less than its decrease in bias. iii. Less flexible and hence will give improved prediction accuracy when its increase in bias is less than its decrease in variance. iv. Less flexible and hence will give improved prediction accuracy when its increase in variance is less than its decrease in bias.

The third one is right. The lasso, relative to least squares, is less flexible and hence will give improved prediction accuracy when its increase in bias is less than its decrease in variance. Lasso's advantage over least squares is the bias-variance trade-off. When the least squares estimates have excessively high variance, the lasso solution can yield a reduction in variance at the expense of a small increase in bias. Therefore, it can generate more accurate predictions. The other advantage of lasso is that it performs variable selection which makes it easier to interpret than other methods like ridge regression.

###(b) Repeat (a) for ridge regression relative to least squares.

Also the third one is right. The ridge regression relative to least squares, is less flexible and hence will give improved prediction accuracy when its increase in bias is less than its decrease in variance.The same as Lasso, Ridge regression's advantage over least squares is the bias-variance trade-off. As λ increases, the flexibility of the ridge regression fit decreases leading to decreased variance but increased bias. Considering the relationship between λ and variance and bias in different regression methods: when there is small change in the training data, the least squares coefficient produces a big change and a larger value of variance as well. But for ridge regression, it can still perform well by trading off a small increase in bias for a large decrease in variance so that the test MSE will not get larger too much. Therefore, between these two methods, ridge regression works better in the situation where the least squares estimates have high variance.

###(c) Repeat (a) for non-linear methods relative to least squares.

The second one is right. The non-linear methods relative to least squares, is more flexible and hence will give improved prediction accuracy when its increase in variance is less than its decrease in bias.

6.9

In this exercise, we will predict the number of applications received using the other variables in the College data set. ###(a) Split the data set into a training set and a test set.

Here I got a training set named "train" and a test set named "test".

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

```
## [1] 1135758
```

Using least squares on the training set, the test error obtained is 1135758.

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

```
## Loading required package: Matrix
```

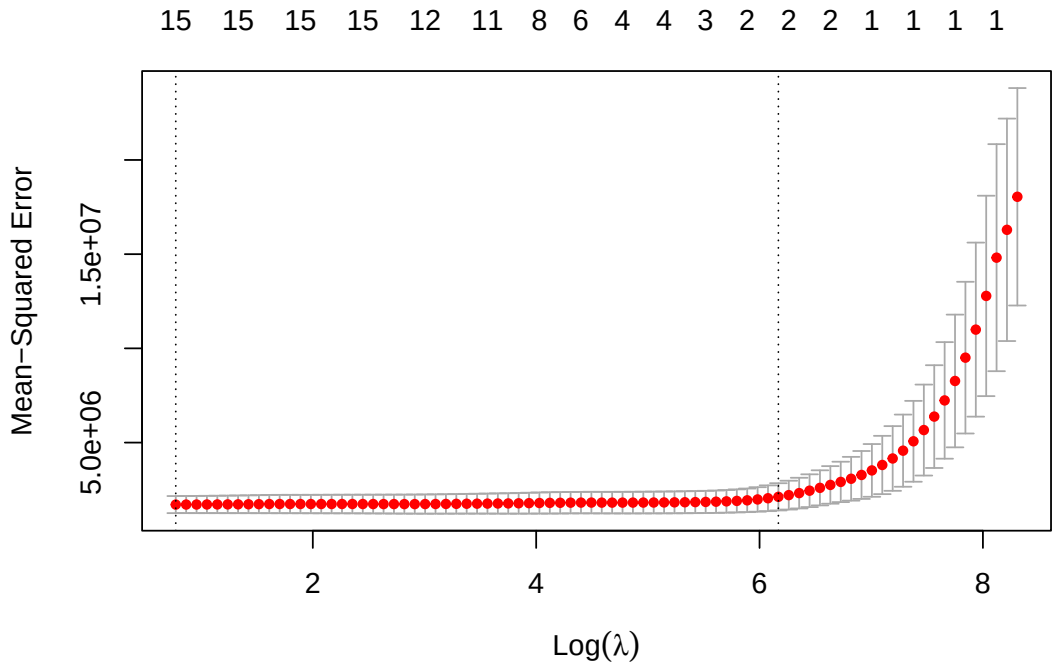
```
## Loaded glmnet 4.1-3
```

```
## [1] 405.8404
```

```
## [1] 1007688
```

It would be better to use cross-validation to choose the tuning parameter λ. Here I set a random seed first so the results will be reproducible, since the choice of the cross-validation folds is random. Therefore, I see that the value of λ that results in the smallest cross-validation error is 405.8404. This time I want to get predictions for a test set, by using "newx" argument. And the test MSE is 1007688, which represents a further improvement over the test MSE that I got using least squares.

###(d) Fit a lasso model on the training set, with λ chosen by cross-validation. Report the test error obtained, along with the number of non-zero coefficient estimates.



```
## [1] 2.165848
```

```
## [1] 1140473
```

We can see from the coefficient plot that depending on the choice of tuning parameter, some of the coefficients will be exactly equal to zero. Next I perform cross-validation and compute the associated test error, which is 1140473 here. So I think the Lasso model is not as strong at predicting as the ridge regression on this data.

```
##   (Intercept)   (Intercept)        Accept        Enroll     Top10perc
## -1.082409e+03  0.000000e+00  1.760769e+00 -1.356850e+00  6.472719e+01
##     Top25perc   F.Undergrad   P.Undergrad      Outstate    Room.Board
## -2.052477e+01  9.044028e-02  1.388779e-02 -1.187542e-01  2.033676e-01
##         Books      Personal           PhD      Terminal     S.F.Ratio
##  2.591197e-01  0.000000e+00 -1.351004e+01  6.464269e+00  2.776541e+01
##   perc.alumni        Expend     Grad.Rate
## -1.445862e-02  5.025451e-02  6.605791e+00
```

```
##   (Intercept)        Accept        Enroll     Top10perc     Top25perc
## -1.082409e+03  1.760769e+00 -1.356850e+00  6.472719e+01 -2.052477e+01
##   F.Undergrad   P.Undergrad      Outstate    Room.Board         Books
##  9.044028e-02  1.388779e-02 -1.187542e-01  2.033676e-01  2.591197e-01
##           PhD      Terminal     S.F.Ratio   perc.alumni        Expend
## -1.351004e+01  6.464269e+00  2.776541e+01 -1.445862e-02  5.025451e-02
##     Grad.Rate
##  6.605791e+00
```

However, the lasso has a substantial advantage over ridge regression in that the resulting coefficient estimates are sparse. Here I find that 7 of the 18 coefficient estimates are exactly zero. So the lasso model with λ chosen by cross-validation contains only 11 variables.

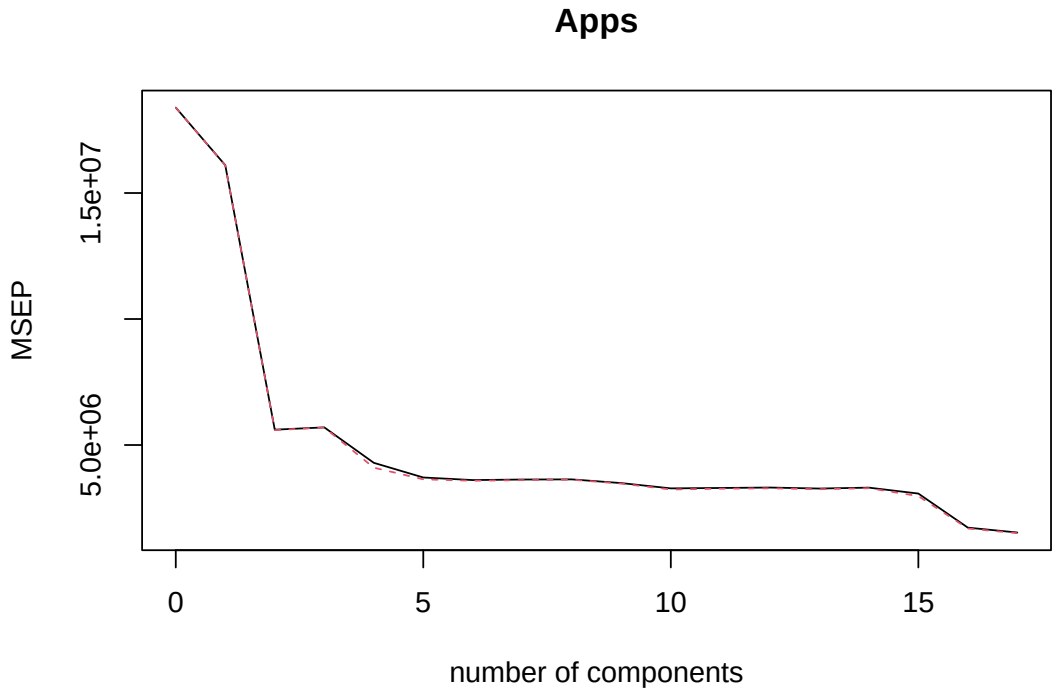
###(e) Fit a PCR model on the training set, with M chosen by cross-validation. Report the test error obtained, along with the value of M selected by cross-validation.

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

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

```
## Data:    X dimension: 388 17
##  Y dimension: 388 1
## Fit method: svdpc
## Number of components considered: 17
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            4288     4013     2368     2388     2072     1926     1900
## adjCV         4288     4012     2364     2386     2025     1907     1893
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
```

```
## CV        1905     1907     1868      1811      1815      1820      1809
## adjCV     1899     1903     1862      1799      1807      1812      1801
##        14 comps  15 comps  16 comps  17 comps
## CV         1819      1753      1312      1236
## adjCV      1813      1725      1298      1225
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       32.20    57.78    65.31    70.99    76.37    81.27     84.8    87.85
## Apps    13.44    70.93    71.07    79.87    81.15    82.25     82.3    82.33
##       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X       90.62     92.91     94.98     96.74     97.79     98.72     99.42
## Apps    83.38     84.76     84.80     84.84     85.11     85.14     90.55
##       16 comps  17 comps
## X        99.88    100.00
## Apps     93.42     93.89
```



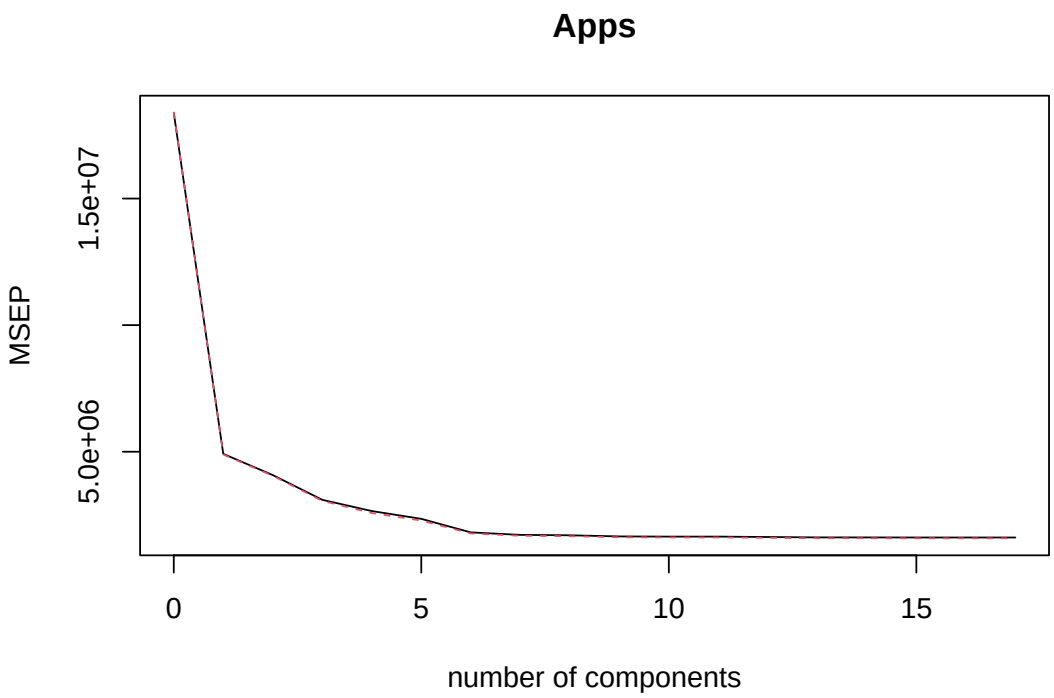
Here I apply PCR to the data.Setting "scale = TRUE" has the effect of standardizing each predictor, prior to generating the principal components, so that the scale on which each variable is measured will not have an effect. Setting 'validation ="CV" ' causes "pcr()" to compute the ten-fold cross-validation error for each possible value of M, the number of principal components used.The CV score is provided for each possible number of components, ranging from M=0 onwards. One can also plot the cross-validation scores using the "validationplot" function. We see that the smallest cross-validation error occurs when about M =17 components are used.

```
## [1] 1963819
```

This test set MSE is 1963819 here.

###(f) 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.

```
## Data:    X dimension: 388 17
##  Y dimension: 388 1
## Fit method: kernelpls
## Number of components considered: 17
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            4288     2217     2019     1761     1630     1533     1347
## adjCV         4288     2211     2012     1749     1605     1510     1331
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        1309     1303     1286      1283      1283      1277      1271
## adjCV     1296     1289     1273      1270      1270      1264      1258
##        14 comps  15 comps  16 comps  17 comps
## CV         1270      1270      1270      1270
## adjCV      1258      1257      1257      1257
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       27.21    50.73    63.06    65.52    70.20    74.20    78.62    80.81
## Apps    75.39    81.24    86.97    91.14    92.62    93.43    93.56    93.68
##       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X       83.29     87.17     89.15     91.37     92.58     94.42     96.98
## Apps    93.76     93.79     93.83     93.86     93.88     93.89     93.89
##       16 comps  17 comps
## X        98.78    100.00
## Apps     93.89     93.89
```




```
## [1] 1181808
```

The test MSE here in partial least squares is 1181808.

###(g) 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?

In order to compare the results obtained above, we need to compute the test R^2 for all models.

```
## [1] 0.9015413
```

```
## [1] 0.9126437
```

```
## [1] 0.9011326
```

```
## [1] 0.8297569
```

```
## [1] 0.8975493
```

So the test R^2 for the least squares is 0.9015413, the test R^2 for ridge regression is 0.9015558, the test R^2 for lasso regression is 0.9011326, the test R^2 for PCR is 0.8297569, and the test R^2 for PLS is 0.8975493. Maybe I can conclude that, except PCR, all models predict college applications with high accuracy.

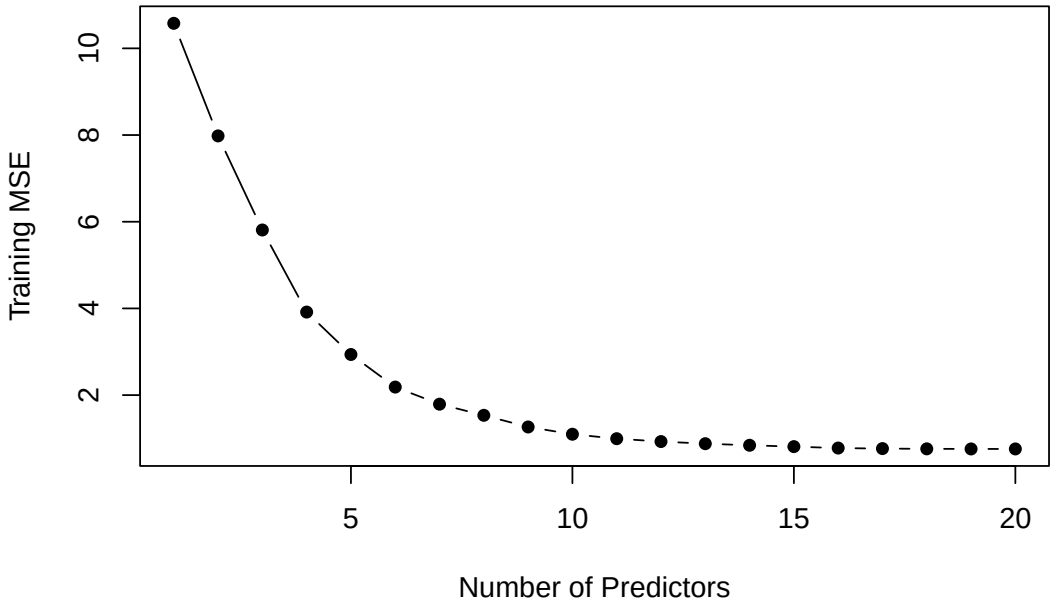
6.10

We have seen that as the number of features used in a model increases, the training error will necessarily decrease, but the test error may not. We will now explore this in a simulated data set. ###(a) Generate a data set with p = 20 features, n = 1,000 observations, and an associated quantitative response vector generated according to the model $Y = X\beta + \epsilon$, where β has some elements that are exactly equal to zero.

###(b) Split your data set into a training set containing 100 observations and a test set containing 900 observations.

###(c) Perform best subset selection on the training set, and plot the training set MSE associated with the best model of each size.

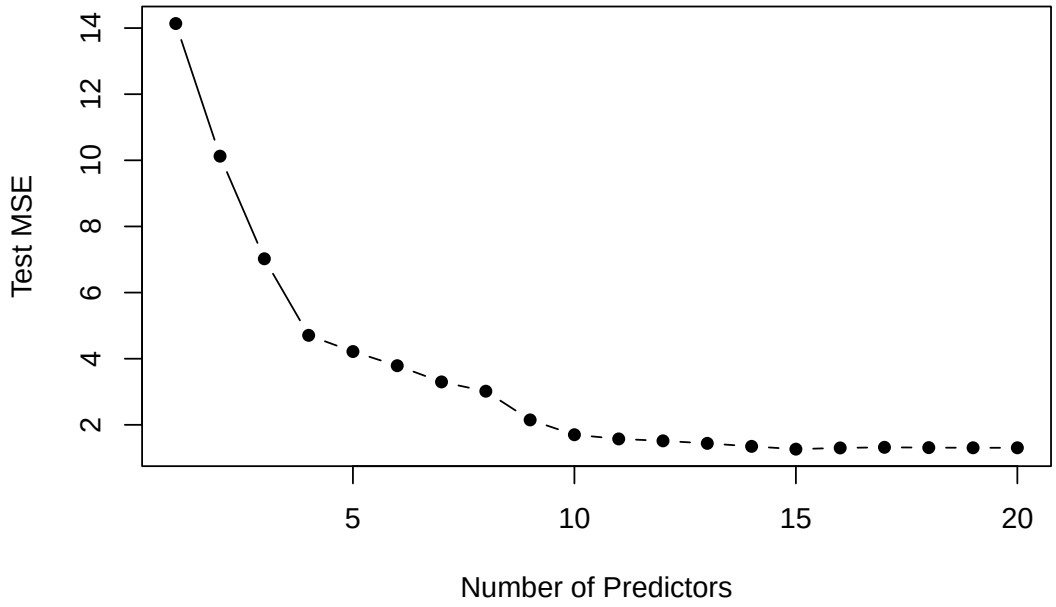
```
##  [1] 0.3356506 0.4987718 0.6351161 0.7540431 0.8155401 0.8627505 0.8875210
##  [8] 0.9036620 0.9205617 0.9310782 0.9375495 0.9416901 0.9447828 0.9470898
## [15] 0.9490339 0.9511045 0.9518888 0.9522450 0.9523812 0.9523831
```



"regsubsets()" function performs best subset selection by identifying the best model that contains a given number of predictors, where best is quantified using RSS. An asterisk indicates that a given variable is included in the corresponding model. For instance, the output above shows that the best two-variable model contains only "x.9" and "x.4". The "nvmax" option can be used in order to return as many variables as are desired. Here I fit uo to a 20-variable model. The "summary()" function also returns R^2, RSS, adjusted R^2, C_p, and BIC. For example, I find that the R^2 statistics increases from 33.6%, when only one variable is included in the model, to almost 95.2%, when all variables are included. As expected, the R^2 statistic increases monotonically as more variables are included. Next I make a model matrix from the training data. And then I run a loop, and for each size i, I extract the coefficients from "regfit.full" for the best model of that size, multiply them into the appropriate columns of the training model matrix to form the predictions, and compute the training MSE. I find that the best model is the one that contains all of the variables.

###(d) Plot the test set MSE associated with the best model of each size.

```
##  [1] 0.2265516 0.4425454 0.6096191 0.7442944 0.7804428 0.8191458 0.8495212
##  [8] 0.8756987 0.9015365 0.9178675 0.9238504 0.9285978 0.9326987 0.9365071
## [15] 0.9401078 0.9402148 0.9402421 0.9402540 0.9402591 0.9402603
```



Here I do the same thing on test data. The R^2 statistics increases from 22.7%, when only one variable is included in the model, to almost 94.0%, when all variables are included. And then after viewing the test MSE, I find that the best model is the one that contains 15 of the variables.

###(e) For which model size does the test set MSE take on its minimum value? Comment on your results. If it takes on its minimum value for a model containing only an intercept or a model containing all of the features, then play around with the way that you are generating the data in (a) until you come up with a scenario in which the test set MSE is minimized for an intermediate model size.

```
## [1] 15
```

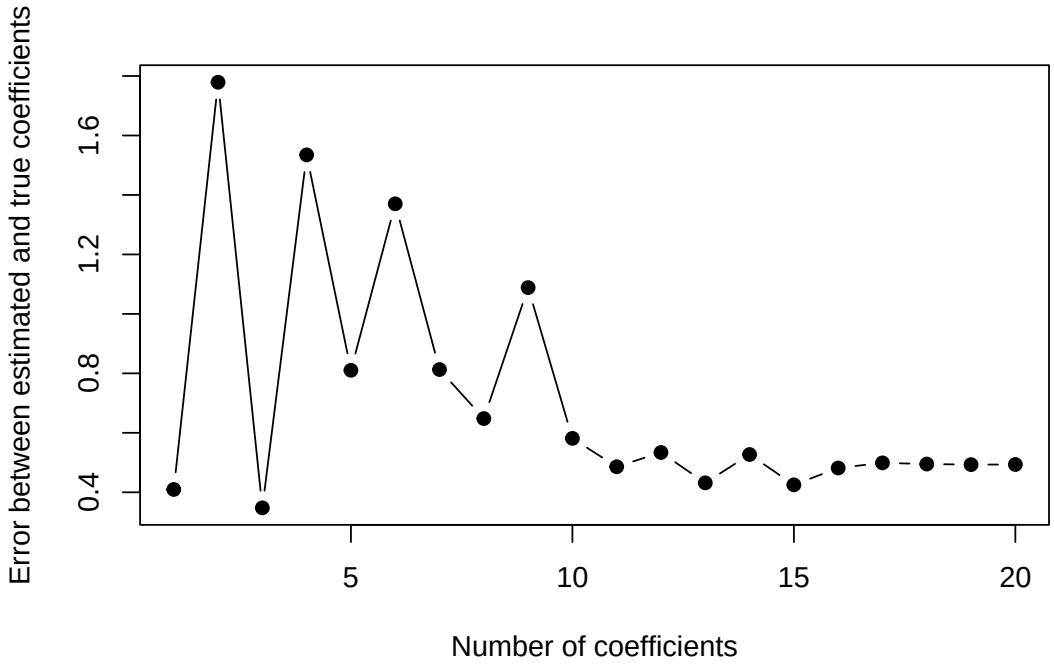
The 15-variables model has the smallest test MSE, which can be concluded by the test MSE in (d).

###(f) How does the model at which the test set MSE is minimized compare to the true model used to generate the data? Comment on the coefficient values.

```
## (Intercept)         x.2         x.3         x.4         x.6         x.7
## -0.03999621  0.34103693 -0.69986968 -1.66718571 -0.26177326 -1.39594243
##         x.8         x.9        x.11        x.12        x.13        x.14
##  0.69098659  2.00166158  0.87224450  0.55589630 -0.21737777 -0.47317440
##        x.16        x.17        x.18        x.19
## -0.33849492  0.21662650  1.61859895  0.71284344
```

The best model caught all zeroed out coefficients.

###(g) Create a plot displaying $\sqrt{\sum_{j=1}^{p}(\beta_j - \hat{\beta}_j^r)^2}$ for a range of values of r where $\hat{\beta}_j^r$ is the jth. coefficient estimate for the best model containing r coefficients. Comment on what you observe. How does this compare to the test MSE plot from (d)?

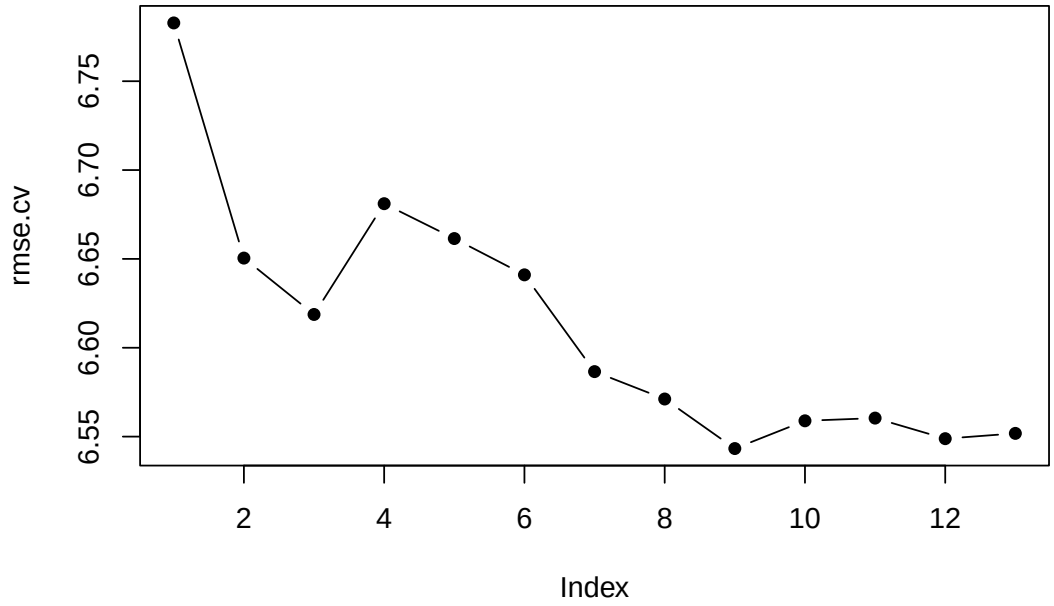


We may see that the model with 3 variables minimizes the error between the estimated and true coefficients. However test error is minimized by the model with 15 variables. So, a better fit of true coefficients does not necessarily mean a lower test MSE.

6.11

We will now try to predict per capita crime rate in the Boston data set. ###(a) Try out some of the regression methods explored in this chapter, such as best subset selection, the lasso, ridge regression, and PCR. Present and discuss results for the approaches that you consider.

Best subset selection



Here I try to choose among the models of different sizes using cross-validation. First, we create a vector that allocates each observation to one of k=10 folds, and I create a matrix in which I will store the results. Then I write a for loop that performs cross-validation. In the ith fold, the elements of folds that equal i are un the test set, and the remainder are in the training set. I make the predictions for each model size, compute the test errors on the appropriate subset, and store them in the appropriate slot in the matrix "cv.errors". The code will automatically use the "predict.regsubsets()" function when I call "predict()" because the "best.fit" object has class "regsubsets". After fitting 13 models, which equals to the number of variables minus 1, I will need to find the one model that minimizes the CV error on the test data.

```
## Subset selection object
## Call: regsubsets.formula(crim ~ ., data = Boston[folds != i, ], nvmax = p)
## 13 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
## black       FALSE      FALSE
## lstat       FALSE      FALSE
## medv        FALSE      FALSE
```

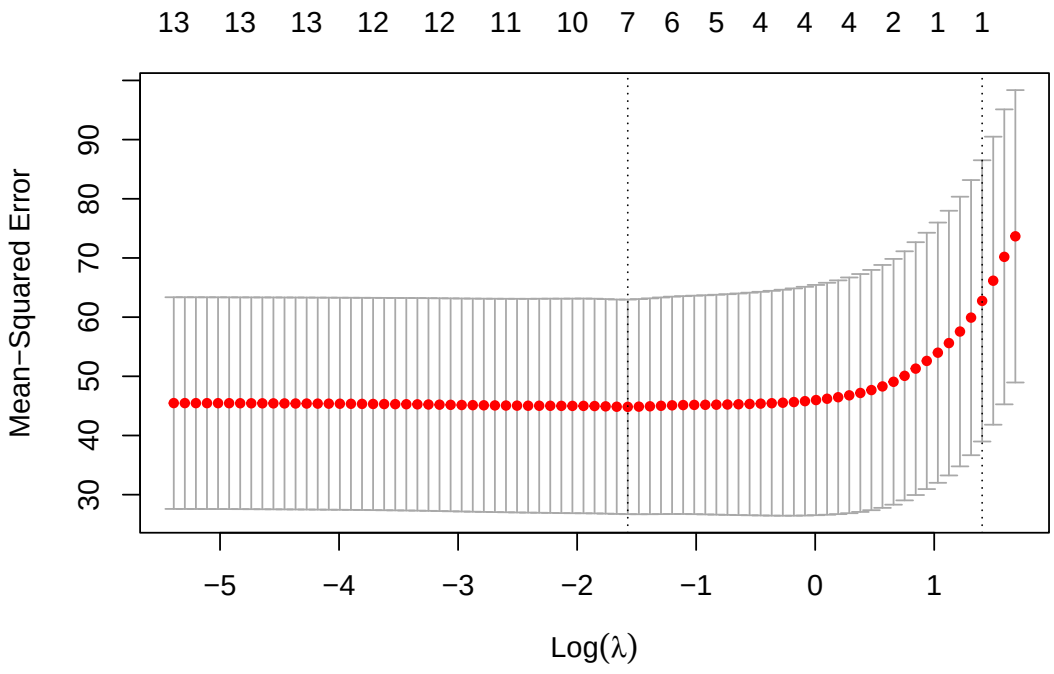
```
## 1 subsets of each size up to 13
## Selection Algorithm: exhaustive
##           zn  indus chas nox rm  age dis rad tax ptratio black lstat medv
## 1  ( 1 )  " " " "   " "  " " " " " " " " "*" " " " "     " "   " "   " "
## 2  ( 1 )  " " " "   " "  " " " " " " " " "*" " " " "     " "   "*"   " "
## 3  ( 1 )  " " " "   " "  " " " " " " " " "*" " " " "     "*"   "*"   " "
## 4  ( 1 )  "*" " "   " "  " " " " " " " " "*" " " " "     "*"   "*"   " "
## 5  ( 1 )  "*" " "   " "  " " " " " " " "*" "*" " " " "     " "   "*"   "*"
## 6  ( 1 )  "*" "*"   " "  " " " " " " " "*" "*" " " " "     " "   "*"   "*"
## 7  ( 1 )  "*" "*"   " "  " " " " " " " "*" "*" " " " "     "*"   "*"   "*"
## 8  ( 1 )  "*" " "   " "  "*" " " " " " "*" "*" " " "*"     "*"   "*"   "*"
## 9  ( 1 )  "*" " "   " "  "*" "*" " " " "*" "*" " " "*"     "*"   "*"   "*"
## 10  ( 1 ) "*" "*"   " "  "*" "*" " " " "*" "*" " " "*"     "*"   "*"   "*"
## 11  ( 1 ) "*" "*"   "*"  "*" "*" " " " "*" "*" " " "*"     "*"   "*"   "*"
## 12  ( 1 ) "*" "*"   "*"  "*" "*" " " " "*" "*" "*" "*"     "*"   "*"   "*"
## 13  ( 1 ) "*" "*"   "*"  "*" "*" "*" "*" "*" "*" "*"     "*"   "*"   "*"
```

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

```
## [1] 42.81453
```

I see that cross-validation selects a 9-variable model based on the test MSE. At 9-variables, the CV estimate for the test MSE is 43.32807. The variables that are included in this model are zn,indus,nox,dis,rad,ptratio,black,lstat, and medv.

The lasso

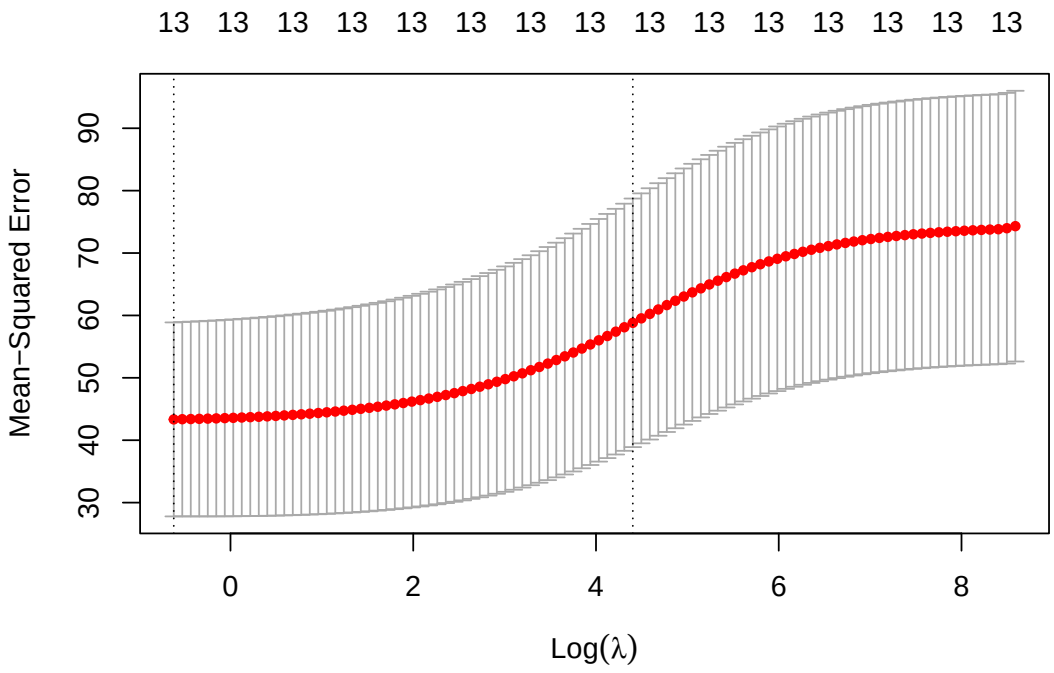


The graph above depicts the relationship between log λ and MSE. To help predict the training model on the test model, I will need to find the λ that reduces the error the most.

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                    s1
## (Intercept) 2.176491
## zn          .
## indus       .
## chas        .
## nox         .
## rm          .
## age         .
## dis         .
## rad         0.150484
## tax         .
## ptratio     .
## black       .
## lstat       .
## medv        .
```

```
## [1] 62.74783
```

As we know that Lasso is a variable reduction method. From the results shown above, the Lasso model that reduces the MSE the model includes only one variable and has an MSE of 55.02399. The only variable included in this model is "rad". # Ridge regression



Ridge regression keeps all the variables but push their coefficient value close to zero if they do not have significance in the relationship with the response.

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept)  1.523899542
## zn          -0.002949852
## indus        0.029276741
## chas        -0.166526007
## nox          1.874769665
## rm          -0.142852604
## age          0.006207995
## dis         -0.094547258
## rad          0.045932737
## tax          0.002086668
## ptratio      0.071258052
## black       -0.002605281
## lstat        0.035745604
## medv        -0.023480540
```

```
## [1] 58.8156
```

The MSE for the ridge regression method is 61.37358 – much larger than those in other two methods. So I think the ridge regression doesn't perform well. # PCR

```
## Data:    X dimension: 506 13
##  Y dimension: 506 1
## Fit method: svdpc
## Number of components considered: 13
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            8.61    7.175    7.180    6.724    6.731    6.727    6.727
## adjCV         8.61    7.174    7.179    6.721    6.725    6.724    6.724
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV       6.722    6.614    6.618     6.607     6.598     6.553     6.488
## adjCV    6.718    6.609    6.613     6.602     6.592     6.546     6.481
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       47.70    60.36    69.67    76.45    82.99    88.00    91.14    93.45
## crim    30.69    30.87    39.27    39.61    39.61    39.86    40.14    42.47
##       9 comps  10 comps  11 comps  12 comps  13 comps
## X       95.40     97.04     98.46     99.52     100.0
## crim    42.55     42.78     43.04     44.13      45.4
```

Based on the CV error as well as the variances explained, I think that the appropriate PCR model would only include 8 components. With 8 components, 93.45% of the variance is explained in the predictors by the model, and 42.47% of the variance is explained in the response variable by the model. Additionally, at 8 components, the MSE is at

###(b) Propose a model (or set of models) that seem to perform well on this data set, and justify your answer. Make sure that you are evaluating model performance using validation set error, cross-validation, or some other reasonable alternative, as opposed to using training error.

As mentioned above, the model that has the lowest cross-validation error is the one chosen by the best subset selection method, which has a MSE of 43.32807. ###(c) Does your chosen model involve all of the features in the data set? Why or why not?

The model that was chosen by Best Subset Selection only includes 9 variables. The variables that are included in this model are zn, indus, nox, dis, rad, ptratio, black, lstat and medv. If the model were to include of the thrown-out features, more variation of the response would be present. For this particular problem, we are looking to have model prediction accuracy with low variance and low MSE.