ISL Chapter 6 Exercises

Jonathan Bryan

June 3, 2018

1. We perform best subset, forward stepwise, and backward stepwise selection on a single data set. For each approach, we obtain $p+1$ models, containing $0, 1, 2, ..., p$ predictors. Explain your answers:

(a) Which of the three models with k predictors has the smallest training RSS?

Best subset selection does an exhaustive search of the model space and therefore will always find the model with the lowest training RSS.

(b) Which of the three models with k predictors has the smallest test RSS?

This depends on whether a validation set or cross-validation was used to evaluate the model selection procedures. If a validation set of cross-validation is not used then either forward or backward stepwise selection are likely to have the lowest test RSS because the best subset model will likely be overfit.

(c) True or False:

i. The predictors in the k-variable model identified by forward stepwise are a subset of the predictors in the (k+1)-variable model identified by forward stepwise selection.

True.

ii. The predictors in the k-variable model identified by backward stepwise are a subset of the predictors in the (k + 1)- variable model identified by backward stepwise selection.

True

iii. The predictors in the k-variable model identified by backward stepwise are a subset of the predictors in the (k + 1)- variable model identified by forward stepwise selection.

False.

iv. The predictors in the k-variable model identified by forward stepwise are a subset of the predictors in the (k+1)-variable model identified by backward stepwise selection.

False.

v. The predictors in the k-variable model identified by best subset are a subset of the predictors in the (k + 1)-variable model identified by best subset selection.

False.

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:

iii is correct. Lasso regression constrains the objective function to include an L_1 penalty. This penalty shrinks, and forces some, coefficients toward zero. This increases the bias of the model while lowering the variance.

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

iii is correct. Ridge regression constrains the objective function to include an L_2 penalty. This shrinks the coefficients toward zero and increases the bias of the model while lowering the variance.

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

ii is correct. Non-linear methods have fewer assumptions regarding the underlying relationship between the response and predictors. This decreases the bias of such methods but makes them more variable because the prediction model is more contingent on the data used to train the model.

3. Suppose we estimate the regression coefficients in a linear regression model by minimizing

$$\Sigma_{i=1}^{n}(y_i - \beta_0 - \Sigma_{j=1}^{p}\beta_j x_{ij})^2 \text{ subject to } \Sigma_{j=1}^{p}|\beta_j| \leq s$$

for a particular value of s. For parts (a) through (e), indicate which of i. through v. is correct. Justify your answer.

(a) As we increase s from 0, the training RSS will:

iv. is correct. As s increases, this reduces the constraint on the β coefficients. At the upper limit of s, we recover the OLS solution which is the most flexible solution compared to lower values of s.

(b) Repeat (a) for test RSS.

ii. is correct. At some value of s between 0 and ∞ the lasso regression model will find an optimal penalty for out-of-sample testing. As s increases the variance of the model will increase as the bias decreases. At some point, the rate of reduction in bias will overtake the increases in variance and the test RSS decreases. Eventually the rate of increase in variance will overtake the reduction in bias, increasing the test RSS as s approaches infinity.

(c) Repeat (a) for variance.

iii. is correct. As s increases the variance of the model increases while the bias lowers until the OLS solution is recovered.

(d) Repeat (a) for (squared) bias.

iv. is correct. As s increases the bias of the model lowers until the unbiased OLS solution is recovered.

(e) Repeat (a) for the irreducible error.

v. is correct. Irreducible error is not a function of the model and remains constant.

4. Suppose we estimate the regression coefficients in a linear regression model by minimizing

$$\Sigma_{i=1}^{n}(y_i - \beta_0 - \Sigma_{j=1}^{p}\beta_j x_{ij})^2 + \lambda\Sigma_{j=1}^{p}\beta_j^2$$

for a particular value of λ. For parts (a) through (e), indicate which of i. through v. is correct. Justify your answer.

(a) As we increase λ from 0, the training RSS will:

iii is correct. Increasing λ increases the L_2 penalty and shrinks the coefficients toward zero, increasing the bias of the model. At zero, the ridge regression model is the OLS solution which is the more flexible than the ridge regression with a nonzero λ and fits the training data the best.

(b) Repeat (a) for test RSS.

ii. is correct. At some value of λ between 0 and ∞ the ridge regression model will find an optimal penalty for out-of-sample testing. Initially the variance of the model will decrease faster than the increase in bias but eventually the rate of increase in bias will overtake the reduction in variance, increasing the test RSS as λ approaches infinity.

(c) Repeat (a) for variance.

iv. is correct. The variance of the model will steadily decrease as the ridge penalty further constrains the model and increases the bias.

(d) Repeat (a) for (squared) bias.

iii. is correct. The bias of the model will steadily increase as the coefficients are further constrained by the increasing L_2 penalty.

(e) Repeat (a) for the irreducible error.

v. is correct. Irreducible error is not a function of the model and remains constant.

5. It is well-known that ridge regression tends to give similar coefficient values to correlated variables, whereas the lasso may give quite different coefficient values to correlated variables. We will now explore this property in a very simple setting. Suppose that $n = 2$, $p = 2$, $x_{11} = x_{12}$, $x_{21} = x_{22}$. Furthermore, suppose that $y_1 + y_2 = 0$ and $x_{11} + x_{21} = 0$ and $x_{12} + x_{22} = 0$, so that the estimate for the intercept in a least squares, ridge regression, or lasso model is zero: $\hat{\beta_0} = 0$

(a) Write out the ridge regression optimization problem in this setting.

$$x_{11}, x_{12} = x_1,\ x_{21}, x_{22} = x_2$$
$$arg\ \min_{\beta}\ \Sigma_{i=1}^{2}(\hat{y_i} - y_i)^2 + \lambda\Sigma_{j=1}^{p}\beta_j^2$$
$$arg\ \min_{\beta}\ (\hat{y_1} - y_1)^2 + (\hat{y_2} - y_2)^2 + \lambda(\beta_1^2 + \beta_2^2)$$
$$arg\ \min_{\beta}\ (\hat{\beta}_1 x_1 + \hat{\beta}_2 x_1 - y_1)^2 + (\hat{\beta}_1 x_2 + \hat{\beta}_2 x_2 - y_2)^2 + \lambda(\beta_1^2 + \beta_2^2)$$

(b) Argue that in this setting, the ridge coefficient estimates satisfy $\hat{\beta}_1 = \hat{\beta}_2$.

$$
\begin{aligned}
&arg\ \min_{\beta}\ (\hat{\beta}_1 x_1 + \hat{\beta}_2 x_1 - y_1)^2 + (\hat{\beta}_1 x_2 + \hat{\beta}_2 x_2 - y_2)^2 + \lambda(\beta_1^2 + \beta_2^2) \\
&arg\ \min_{\beta}\ (\hat{\beta}_1 x_1 + \hat{\beta}_2 x_1)^2 - 2y_1(\hat{\beta}_1 x_1 + \hat{\beta}_2 x_1) + y_1^2 + \\
&\qquad (\hat{\beta}_1 x_2 + \hat{\beta}_2 x_2)^2 - 2y_1(\hat{\beta}_1 x_2 + \hat{\beta}_2 x_2) + y_2^2 + \\
&\qquad \lambda(\beta_1^2 + \beta_2^2) \\
&arg\ \min_{\beta}\ \hat{\beta}_1 x_1^2 + 2\hat{\beta}_1 x_1 \hat{\beta}_2 x_1 + \hat{\beta}_2 x_1^2 - 2y_1\hat{\beta}_1 x_1 - 2y_1\hat{\beta}_2 x_1 + y_1^2 + \\
&\qquad \hat{\beta}_1 x_2^2 + 2\hat{\beta}_1 x_2 \hat{\beta}_2 x_2 + \hat{\beta}_2 x_2^2 - 2y_1\hat{\beta}_1 x_2 - 2y_1\hat{\beta}_2 x_2 + y_2^2 + \\
&\qquad \lambda(\beta_1^2 + \beta_2^2) \\
&\frac{d}{d\hat{\beta}_1} = \hat{\beta}_1(x_1^2 + x_2^2 + \lambda) + \hat{\beta}_2(x_1^2 + x_2^2) = y_1 x_1 + y_2 x_2 \\
&\frac{d}{d\hat{\beta}_2} = \hat{\beta}_1(x_1^2 + x_2^2) + \hat{\beta}_2(x_1^2 + x_2^2 + \lambda) = y_1 x_1 + y_2 x_2 \\
&\frac{d}{d\hat{\beta}_1} - \frac{d}{d\hat{\beta}_2} = \lambda(\hat{\beta}_1 - \hat{\beta}_2) = 0 : \ \ \lambda \neq 0 \implies \hat{\beta}_1 = \hat{\beta}_2
\end{aligned}
$$

(c) Write out the lasso optimization problem in this setting.

$$
\begin{aligned}
&x_{11}, x_{12} = x_1,\ x_{21}, x_{22} = x_2 \\
&arg\ \min_{\beta}\ \Sigma_{i=1}^2 (\hat{y_i} - y_i)^2 + \lambda \Sigma_{j=1}^p |\beta_j| \\
&arg\ \min_{\beta}\ (\hat{y_1} - y_1)^2 + (\hat{y_2} - y_2)^2 + \lambda(|\beta_1| + |\beta_2|) \\
&arg\ \min_{\beta}\ (\hat{\beta}_1 x_1 + \hat{\beta}_2 x_1 - y_1)^2 + (\hat{\beta}_1 x_2 + \hat{\beta}_2 x_2 - y_2)^2 + \lambda(|\beta_1| + |\beta_2|)
\end{aligned}
$$

(d) Argue that in this setting, the lasso coefficients $\hat{\beta}_1$ and $\hat{\beta}_2$ are not unique-in other words, there are many possible solutions to the optimization problem in (c). Describe these solutions.

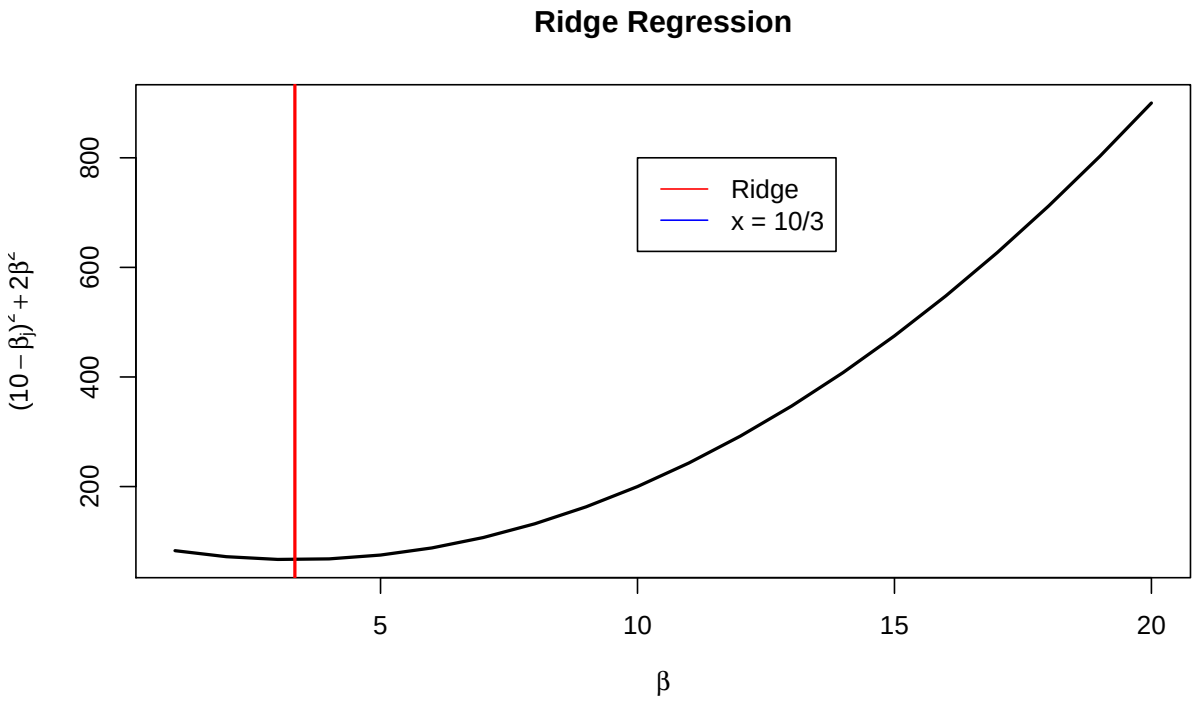
$$
\begin{aligned}
&arg\ \min_{\beta}\ (\hat{\beta}_1 x_1 + \hat{\beta}_2 x_1 - y_1)^2 + (\hat{\beta}_1 x_2 + \hat{\beta}_2 x_2 - y_2)^2 + \lambda(|\beta_1| + |\beta_2|) \\
&arg\ \min_{\beta}\ (\hat{\beta}_1 x_1 + \hat{\beta}_2 x_1)^2 - 2y_1(\hat{\beta}_1 x_1 + \hat{\beta}_2 x_1) + y_1^2 + \\
&\qquad (\hat{\beta}_1 x_2 + \hat{\beta}_2 x_2)^2 - 2y_1(\hat{\beta}_1 x_2 + \hat{\beta}_2 x_2) + y_2^2 + \\
&\qquad \lambda(|\beta_1| + |\beta_2|) \\
&arg\ \min_{\beta}\ \hat{\beta}_1 x_1^2 + 2\hat{\beta}_1 x_1 \hat{\beta}_2 x_1 + \hat{\beta}_2 x_1^2 - 2y_1\hat{\beta}_1 x_1 - 2y_1\hat{\beta}_2 x_1 + y_1^2 + \\
&\qquad \hat{\beta}_1 x_2^2 + 2\hat{\beta}_1 x_2 \hat{\beta}_2 x_2 + \hat{\beta}_2 x_2^2 - 2y_1\hat{\beta}_1 x_2 - 2y_1\hat{\beta}_2 x_2 + y_2^2 + \\
&\qquad \lambda(|\beta_1| + |\beta_2|) \\
&\frac{d}{d\hat{\beta}_1} = \hat{\beta}_1(x_1^2 + x_2^2) + \lambda\frac{|\beta_1|}{\beta_1} + \hat{\beta}_2(x_1^2 + x_2^2) = y_1 x_1 + y_2 x_2 \\
&\frac{d}{d\hat{\beta}_2} = \hat{\beta}_1(x_1^2 + x_2^2) + \hat{\beta}_2(x_1^2 + x_2^2) + \lambda\frac{|\beta_2|}{\beta_2} = y_1 x_1 + y_2 x_2 \\
&\frac{d}{d\hat{\beta}_1} - \frac{d}{d\hat{\beta}_2} = \lambda(\frac{|\beta_1|}{\beta_1} - \frac{|\beta_2|}{\beta_2}) = 0 : \ \ \lambda \neq 0 \implies \hat{\beta}_1, \hat{\beta}_2 \neq 0
\end{aligned}
$$

This demonstrates that $\hat{\beta}_1$ and $\hat{\beta}_2$ can be any real number other than 0 and that there is no unique solution.

6. We will now explore (6.12) and (6.13) further.

(a) Consider (6.12) with p = 1. For some choice of y_1 and $\lambda > 0$, plot (6.12) as a function of β_1. Your plot should confirm that (6.12) is solved by (6.14).

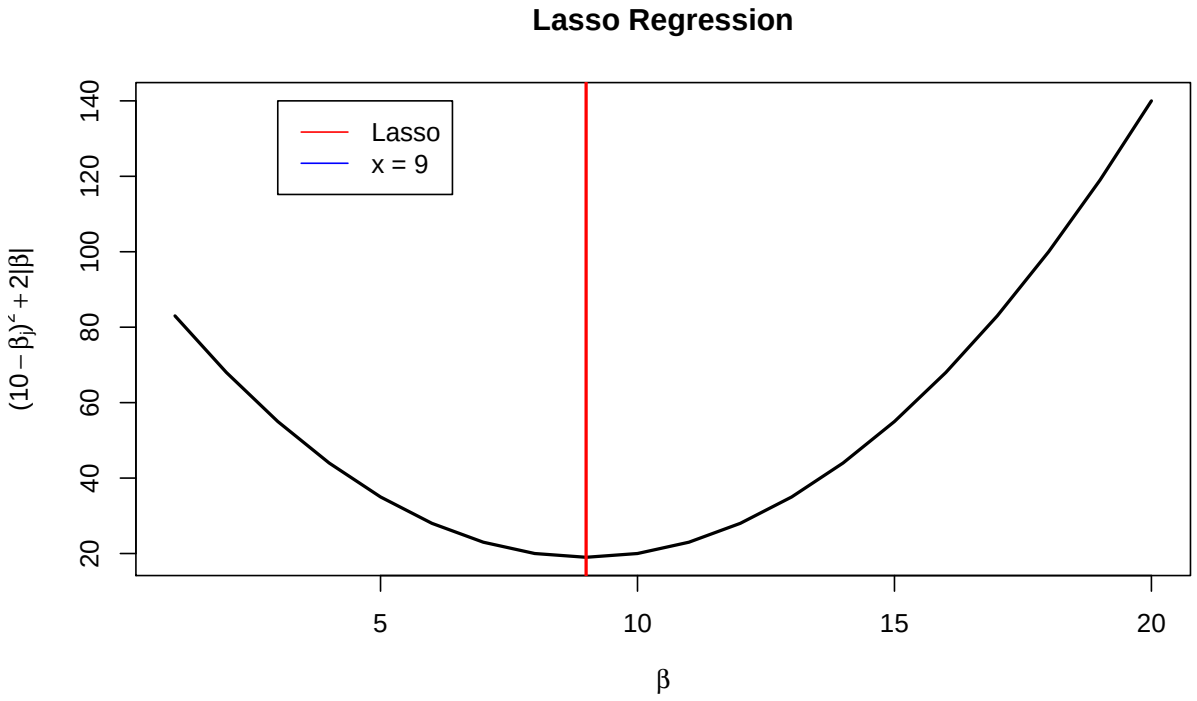
```
library(latex2exp)
y1 = 10
lambda = 2
beta = seq(1,20, by=1)
ridge = (10 - beta)^2 + 2*beta^2
plot(beta, ridge, main = "Ridge Regression",
     ylab = latex2exp("$(10 - \\beta_j)^2 + 2\\beta^2$"),
     xlab = latex2exp("$\\beta$"),
     type ="l",
     lwd=2)
abline(v=10/(1+2), col="red",lwd=2)
legend(10,800,legend = c("Ridge", "x = 10/3"),
                  col = c("red", "blue"),lty=1)
```



(b) Consider (6.13) with p = 1. For some choice ofy_1 and $\lambda > 0$, plot (6.13) as a function of β_1. Your plot should confirm that (6.13) is solved by (6.15).

```
y1 = 10
lambda = 2
beta = seq(1,20, by=1)
lasso = (10 - beta)^2 + 2*abs(beta)
```

```r
plot(beta, lasso, main = "Lasso Regression",
     ylab = latex2exp("$(10 - \\beta_j)^2 + 2| \\beta |$"),
     xlab = latex2exp("$\\beta$"),
     type ="l",
     lwd=2)
abline(v=10 - 1, col="red",lwd=2)
legend(3,140,legend = c("Lasso", "x = 9"),
                  col = c("red", "blue"),lty=1)
```



7. We will now derive the Bayesian connection to the lasso and ridge regression discussed in Section 6.2.2.

(a) Suppose that $y_i = \beta_0 + \Sigma_{j=1}^{p} x_{ij}\beta_j + \epsilon_i$ where $\epsilon_1, ..., \epsilon_n$ are independent and identically distributed from a $N(0, \sigma^2)$ distribution. Write out the likelihood for the data.

$$y_i \sim N(\beta_0 + \Sigma_{j=1}^{p} x_{ij}\beta_j,\ \sigma^2)$$

$$Pr(y_1, ..., y_n | x, \beta, \sigma^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} exp\{-\frac{1}{2\sigma^2}(y_i - \beta_0 + \Sigma_{j=1}^{p} x_{ij}\beta_j)^2\}$$

$$Pr(y_1, ..., y_n | x, \beta, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}^{n} exp\{-\frac{1}{2\sigma^2}\Sigma_{i=1}^{n}(y_i - \beta_0 + \Sigma_{j=1}^{p} x_{ij}\beta_j)^2\}$$

(b) Assume the following prior for β : $\beta_1, ..., \beta_p$ are independent and identically distributed according to a double-exponential distribution with mean 0 and common scale parameter b: i.e. $p(\beta) = \frac{1}{2b} exp(-\frac{|\beta|}{b})$.

$$Pr(\beta|(y_1,...,y_n,x,\sigma^2) = \frac{Pr(y_1,...,y_n|x,\beta,\sigma^2)Pr(\beta)}{\int_{y\in Y} Pr(y_1,...,y_n|x,\beta,\sigma^2)Pr(\beta)dy}$$

$$Pr(\beta|(y_1,...,y_n,x,\sigma^2) \propto \frac{1}{\sqrt{2\pi\sigma^2}}^n \frac{1}{2b} exp\{-\frac{1}{2\sigma^2}\Sigma_{i=1}^n(y_i-\beta_0+\Sigma_{j=1}^p x_{ij}\beta_j)^2\} exp\{-\frac{|\Sigma_{j=0}^p\beta_j|}{b}\}$$

$$Pr(\beta|(y_1,...,y_n,x,\sigma^2) \propto \frac{1}{\sqrt{2\pi\sigma^2}}^n \frac{1}{2b} exp\{-\frac{1}{2\sigma^2}(\Sigma_{i=1}^n(y_i-\beta_0+\Sigma_{j=1}^p x_{ij}\beta_j)^2 + \frac{2\sigma^2|\Sigma_{j=0}^p\beta_j|}{b})\}$$

(c) Argue that the lasso estimate is the mode for β under this posterior distribution.

$$arg\ \max_\beta\ Pr(\beta|(y_1,...,y_n,x,\sigma^2) = arg\ \max_\beta \frac{1}{\sqrt{2\pi\sigma^2}}^n \frac{1}{2b} exp\{-\frac{1}{2\sigma^2}(\Sigma_{i=1}^n(y_i-\beta_0+\Sigma_{j=1}^p x_{ij}\beta_j)^2 + \frac{2\sigma^2|\Sigma_{j=0}^p\beta_j|}{b})\}$$

$$arg\ \max_\beta\ Pr(\beta|(y_1,...,y_n,x,\sigma^2) = arg\ \min_\beta\ log(exp\{-\frac{1}{2\sigma^2}(\Sigma_{i=1}^n(y_i-\beta_0+\Sigma_{j=1}^p x_{ij}\beta_j)^2 + \frac{2\sigma^2|\Sigma_{j=0}^p\beta_j|}{b})\})$$

$$arg\ \max_\beta\ Pr(\beta|(y_1,...,y_n,x,\sigma^2) = arg\ \min_\beta\ \Sigma_{i=1}^n(y_i-\beta_0+\Sigma_{j=1}^p x_{ij}\beta_j)^2 + \frac{2\sigma^2|\Sigma_{j=0}^p\beta_j|}{b}$$

$$\lambda = \frac{2\sigma^2}{b}$$

$$Pr(\beta|(y_1,...,y_n,x,\sigma^2) = arg\ \min_\beta\ \Sigma_{i=1}^n(y_i-\beta_0+\Sigma_{j=1}^p x_{ij}\beta_j)^2 + \lambda|\Sigma_{j=0}^p\beta_j| \text{ (Lasso optimization)}$$

(d) Now assume the following prior for β : $\beta_1,...,\beta_p$ are independent and identically distributed according to a normal distribution with mean zero and variance c. Write out the posterior for β in this setting.

$$Pr(\beta|(y_1,...,y_n,x,\sigma^2) = \frac{Pr(y_1,...,y_n|x,\beta,\sigma^2)Pr(\beta)}{\int_{y\in Y} Pr(y_1,...,y_n|x,\beta,\sigma^2)Pr(\beta)dy}$$

$$Pr(\beta|(y_1,...,y_n,x,\sigma^2) \propto \frac{1}{\sqrt{2\pi\sigma^2}}^n \frac{1}{\sqrt{2\pi c^2}} exp\{-\frac{1}{2\sigma^2}(\Sigma_{i=1}^n(y_i-\beta_0+\Sigma_{j=1}^p x_{ij}\beta_j)^2\} exp\{-\frac{1}{2c^2}\Sigma_{j=0}^p\beta_j^2\}$$

$$Pr(\beta|(y_1,...,y_n,x,\sigma^2) \propto \frac{1}{\sqrt{2\pi\sigma^2}}^n \frac{1}{\sqrt{2\pi c^2}} exp\{-\frac{1}{2\sigma^2}(\Sigma_{i=1}^n(y_i-\beta_0+\Sigma_{j=1}^p x_{ij}\beta_j)^2 + \frac{\sigma^2}{c^2}\Sigma_{j=0}^p\beta_j^2)\}$$

(e) Argue that the ridge regression estimate is both the mode and the mean for β under this posterior distribution.

$$arg\ \max_\beta\ Pr(\beta|(y_1,...,y_n,x,\sigma^2) = arg\ \max_\beta \frac{1}{\sqrt{2\pi\sigma^2}}^n \frac{1}{\sqrt{2\pi c^2}} exp\{-\frac{1}{2\sigma^2}(\Sigma_{i=1}^n(y_i-\beta_0+\Sigma_{j=1}^p x_{ij}\beta_j)^2 + \frac{\sigma^2}{c^2}\Sigma_{j=0}^p\beta_j^2)\}$$

$$arg\ \max_\beta\ Pr(\beta|(y_1,...,y_n,x,\sigma^2) = arg\ \min_\beta\ log(exp\{-\frac{1}{2\sigma^2}(\Sigma_{i=1}^n(y_i-\beta_0+\Sigma_{j=1}^p x_{ij}\beta_j)^2 + \frac{\sigma^2}{c^2}\Sigma_{j=0}^p\beta_j^2)\})$$

$$arg\ \max_\beta\ Pr(\beta|(y_1,...,y_n,x,\sigma^2) = arg\ \min_\beta\ \Sigma_{i=1}^n(y_i-\beta_0+\Sigma_{j=1}^p x_{ij}\beta_j)^2 + \frac{\sigma^2}{c^2}\Sigma_{j=0}^p\beta_j^2)$$

$$\lambda = \frac{\sigma^2}{c^2}$$

$$Pr(\beta|(y_1,...,y_n,x,\sigma^2) = arg\ \min_\beta\ \Sigma_{i=1}^n(y_i-\beta_0+\Sigma_{j=1}^p x_{ij}\beta_j)^2 + \lambda\Sigma_{j=0}^p\beta_j^2 \text{ (Ridge optimization)}$$

8. In this exercise, we will generate simulated data, and will then use this data to perform best subset selection.

(a) 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) #mean=3 and sd=3
e = rnorm(100) #mean=0 and sd =1
```

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

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

where $\beta_0, \beta_1, \beta_2$, and β_3 are constants of your choice.

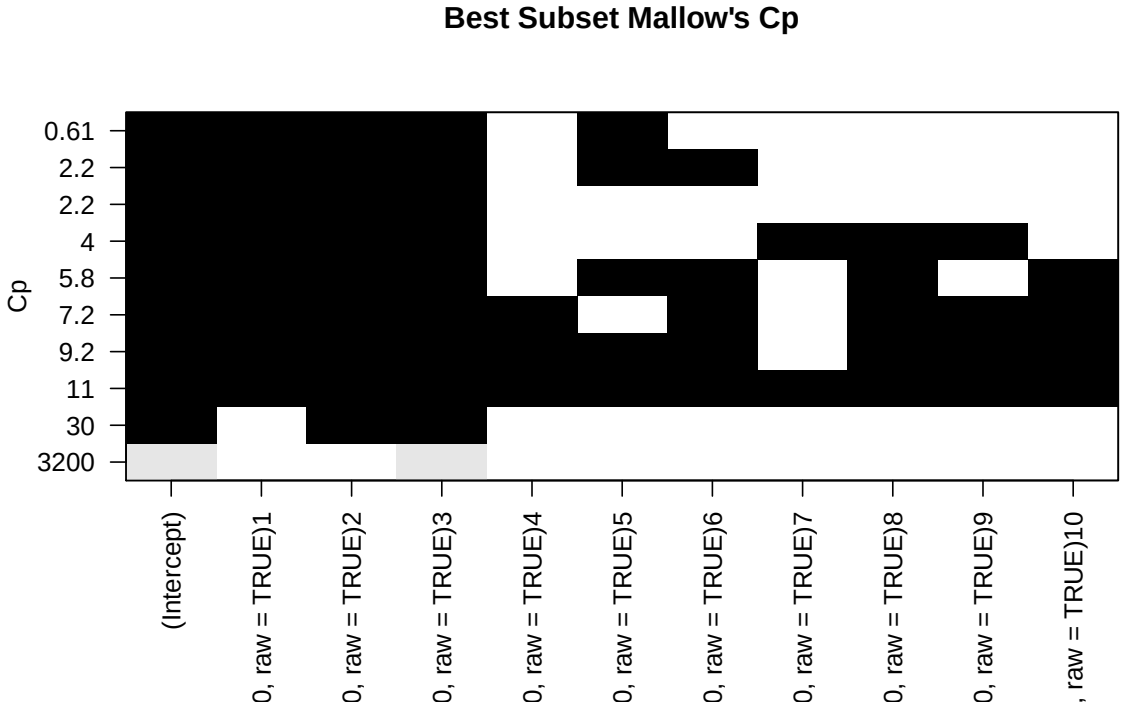
```
y = rep(NA,100)
for (i in 1:100){
  y[i] = 4 -1*X[i] + 5*X[i]^2 - 3.5*X[i]^3 + e[i]
}
```

(c) 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 .

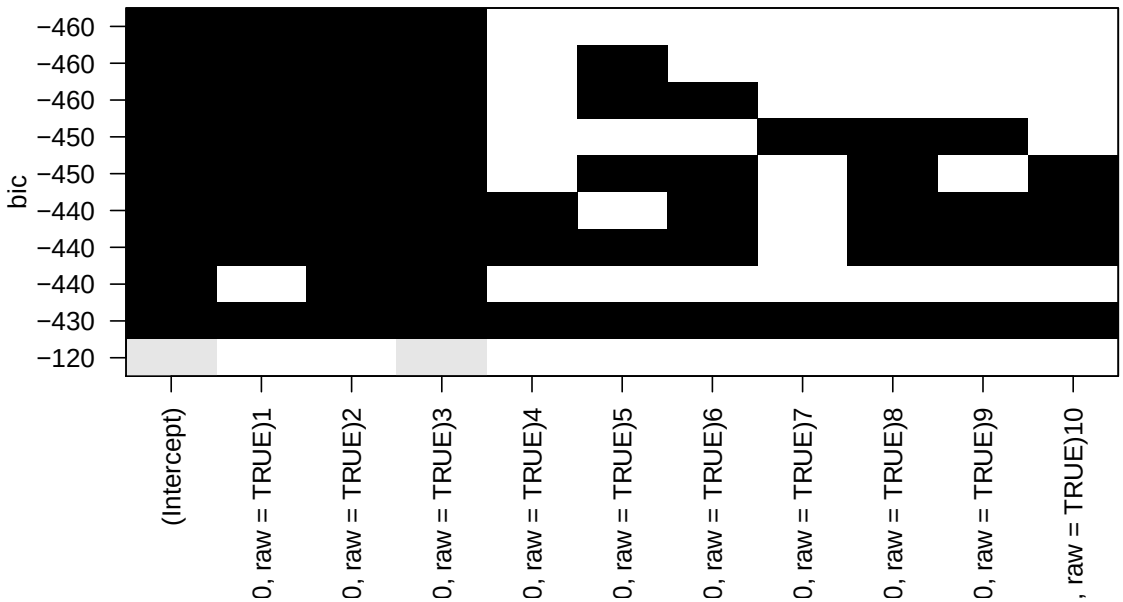
```
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 3.4.4
```

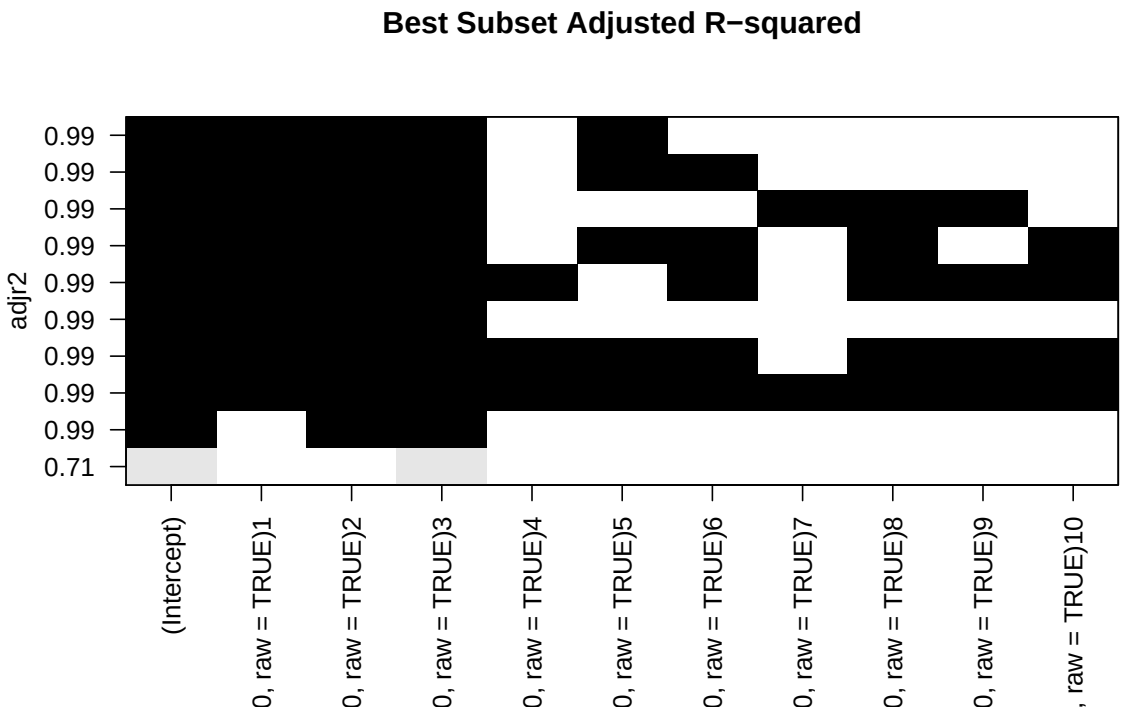
```
df = data.frame(X=X,y=y)
bestsub = regsubsets(y ~ poly(X,10, raw=TRUE), data =df, nvmax=10)
plot(bestsub, scale=c("Cp"), main ="Best Subset Mallow's Cp")
```




```
plot(bestsub, scale=c("bic"), main ="Best Subset BIC")
```



```
plot(bestsub, scale=c("adjr2"), main ="Best Subset Adjusted R-squared")
```



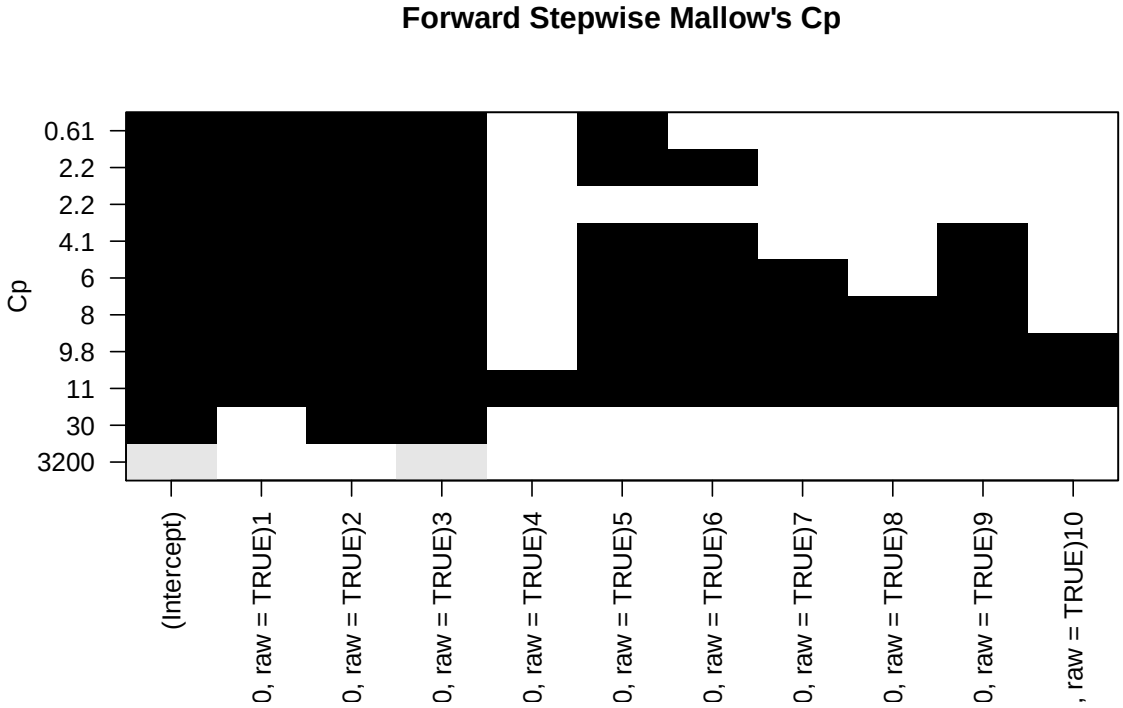

```
bestsub_coef = coefficients(bestsub, id=3) - c(4,-1,5,-3.5)
bestsub_coef
```

```
##              (Intercept) poly(X, 10, raw = TRUE)1 poly(X, 10, raw = TRUE)2
##               0.06150718             -0.02471973             -0.12379099
## poly(X, 10, raw = TRUE)3
##               0.01763858
```

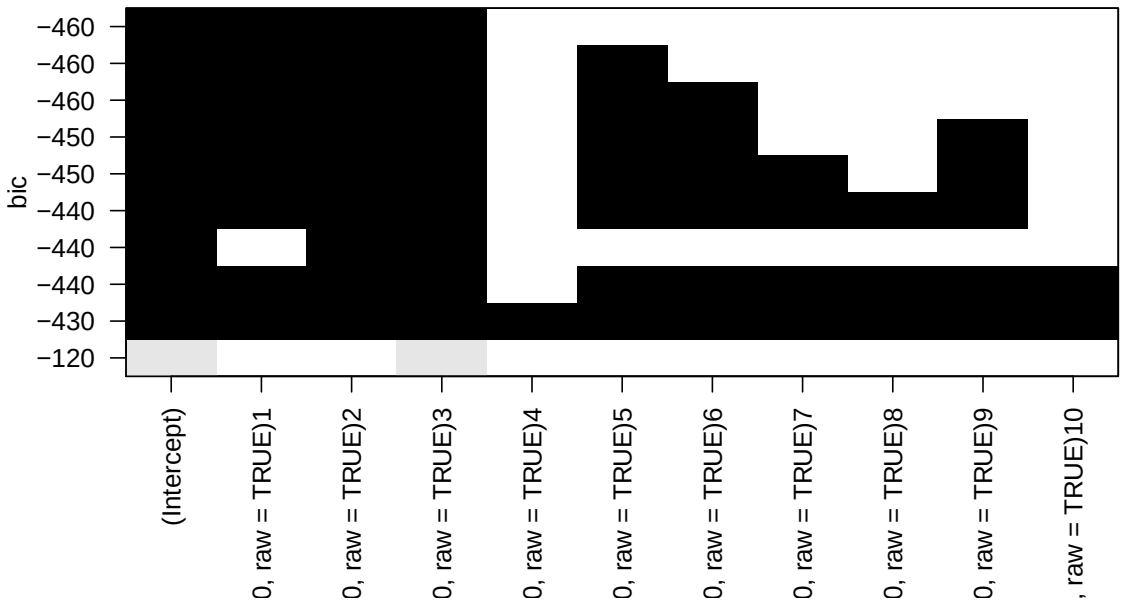
The best model is the four variable five-degree polynomial model exluding the quartic term as determined by the lowest values of Mallow's C_p. Nine of the models, including the true model, are indistinguishable using adjusted R^2. The third-degree polynomial model and the previously specified model best model selected by Mallow's C_p have equal BIC. The coefficients for the third-degree polynomial model are approximate to the true model coefficients.

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

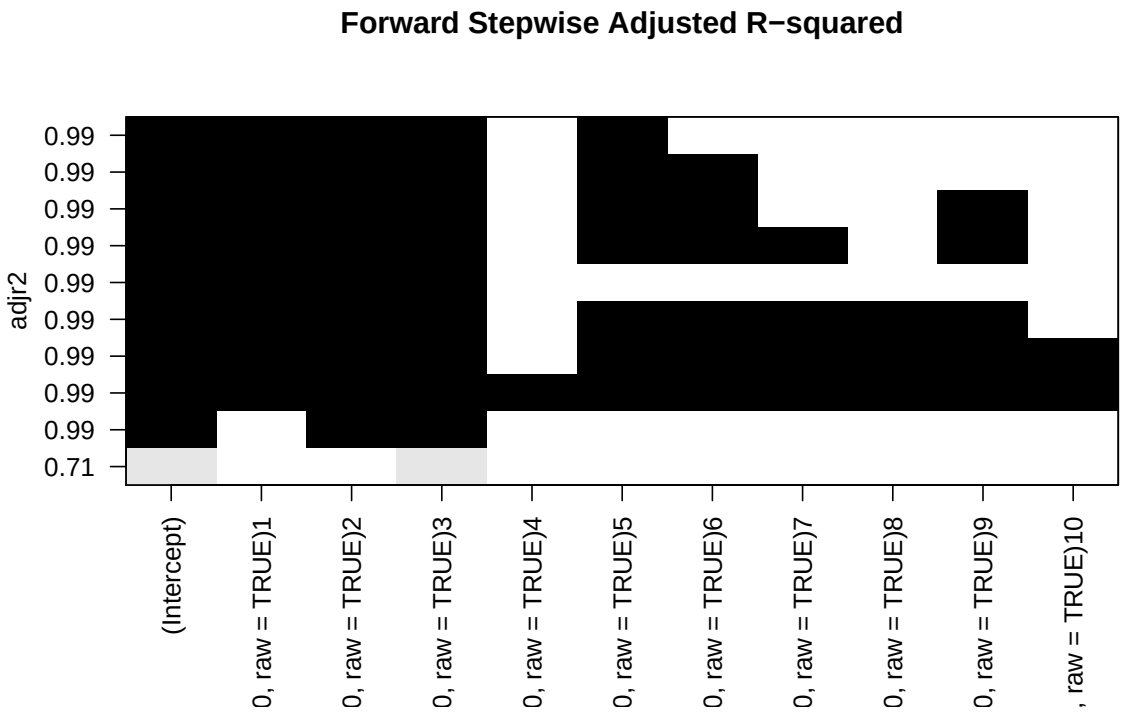
```
#forward stepwise selection
forward = regsubsets(y ~ poly(X,10, raw=TRUE), data =df, method="forward", nvmax=10)
plot(forward , scale=c("Cp"), main ="Forward Stepwise Mallow's Cp")
```



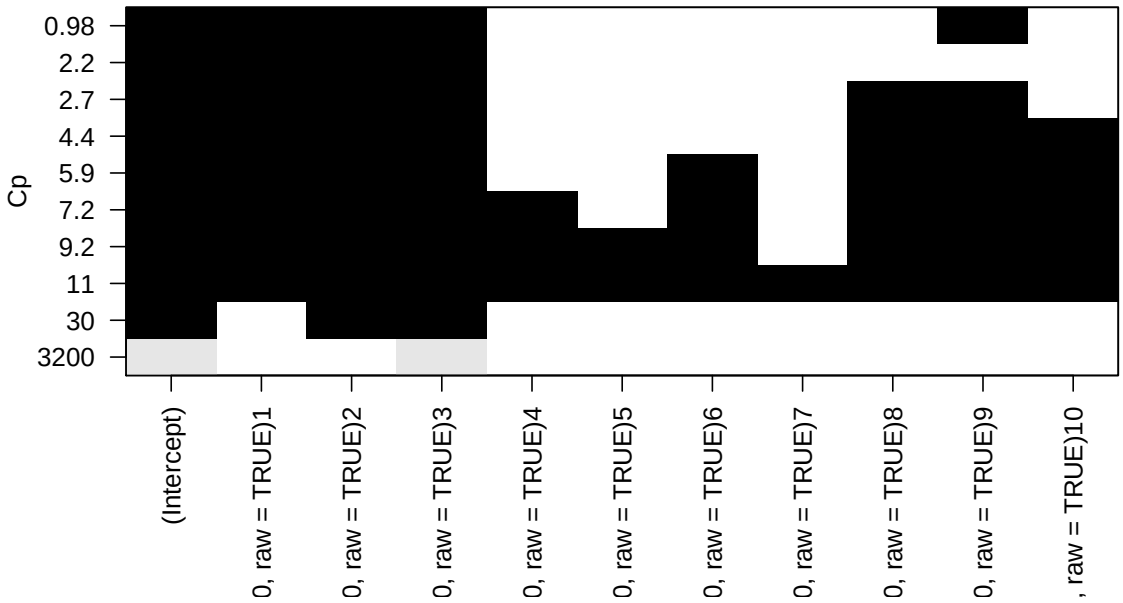

```
plot(forward , scale=c("bic"), main ="Forward Stepwise BIC")
```



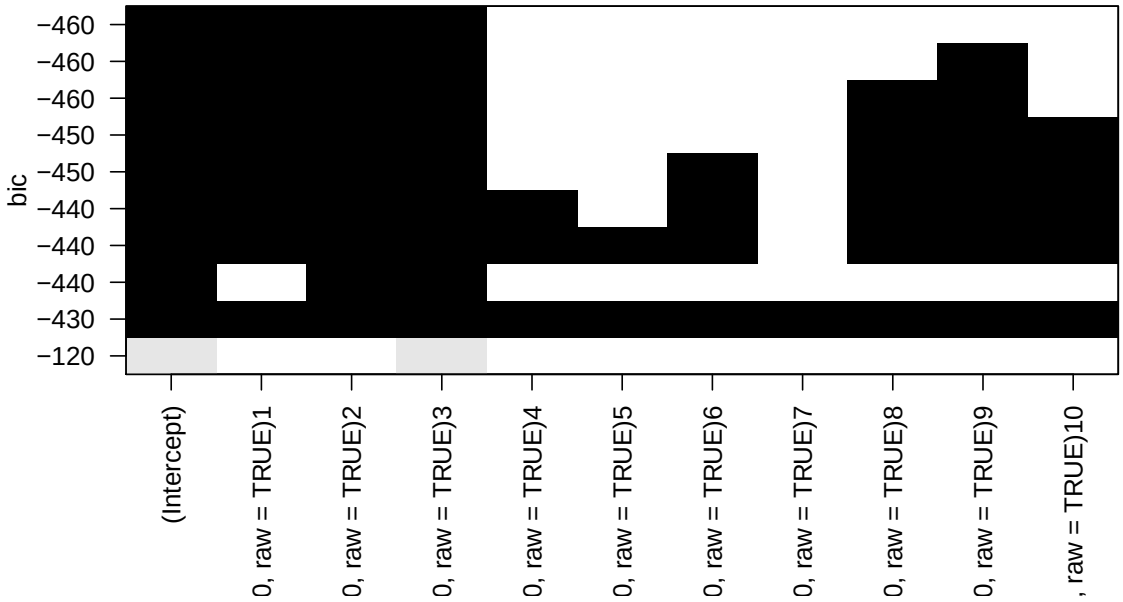
```
plot(forward , scale=c("adjr2"), main ="Forward Stepwise Adjusted R-squared")
```



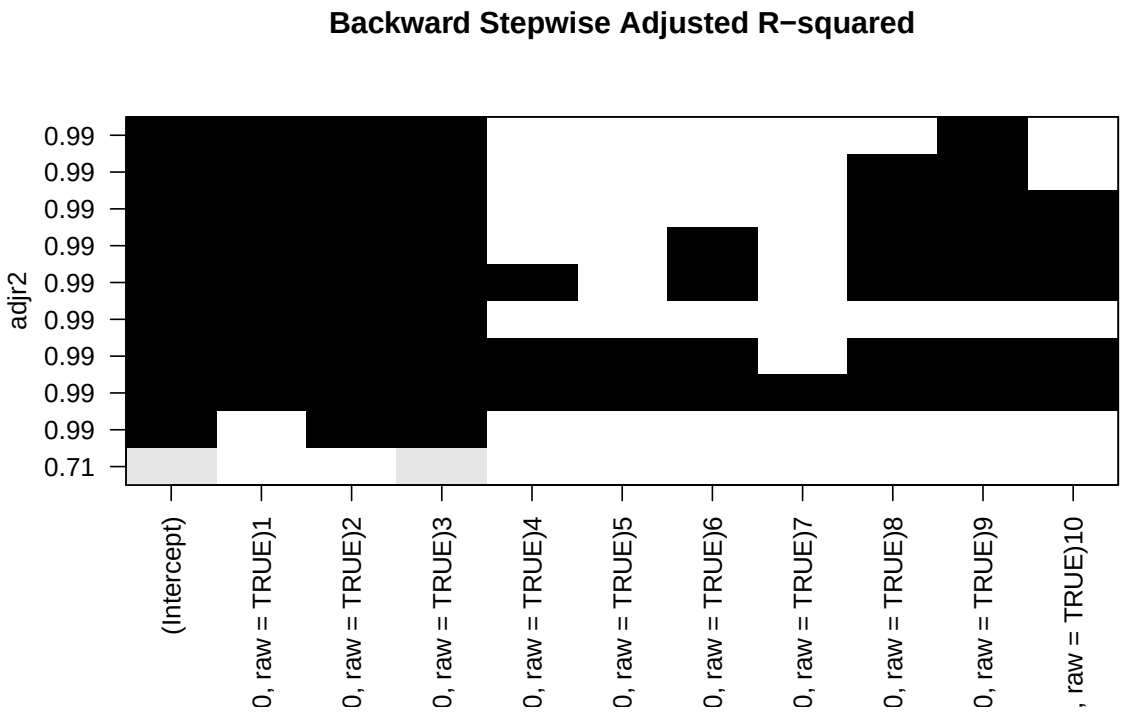

```
forward_coef =coefficients(forward, id=3) - c(4,-1,5,-3.5)

#backward stepwise selection
backward = regsubsets(y ~ poly(X,10, raw=TRUE), data =df, method="backward", nvmax=10)
plot(backward, scale=c("Cp"), main ="Backward Stepwise Mallow's Cp")
```




```
plot(backward, scale=c("bic"), main ="Backward Stepwise BIC")
```



```r
plot(backward, scale=c("adjr2"), main ="Backward Stepwise Adjusted R-squared")
```




```r
backward_coef = coefficients(backward, id=3) - c(4,-1,5,-3.5)

#coefficient table
regselect_coefs = data.frame("Best Subset" = bestsub_coef,
                             "Forward Stepwise" = forward_coef,
                             "Backward Stepwise" = backward_coef)
knitr::kable(regselect_coefs, caption ="Selection Method Coef. Difference Comparison")
```

Table 1: Selection Method Coef. Difference Comparison

	Best.Subset	Forward.Stepwise	Backward.Stepwise
(Intercept)	0.0615072	0.0615072	0.0615072
poly(X, 10, raw = TRUE)1	-0.0247197	-0.0247197	-0.0247197
poly(X, 10, raw = TRUE)2	-0.1237910	-0.1237910	-0.1237910
poly(X, 10, raw = TRUE)3	0.0176386	0.0176386	0.0176386

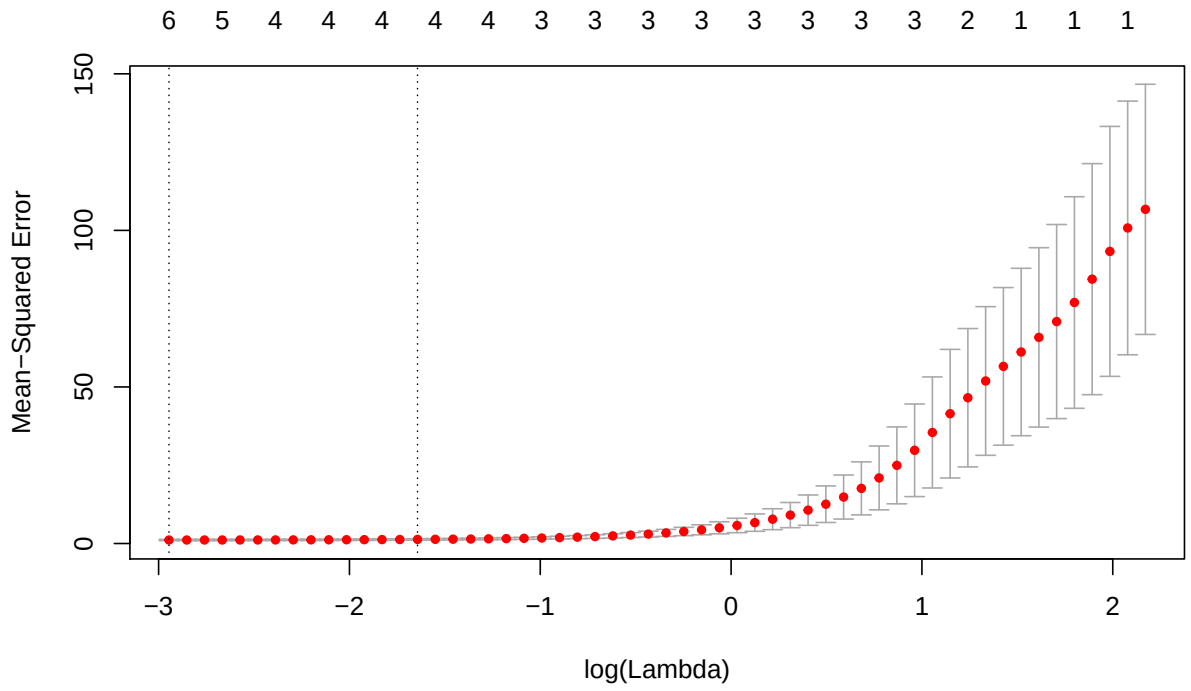
A similar pattern to the best subset results is observed in the forward stepwise approaches. However, the backwards stepwise approach no longer has the true model tied for second when evaluating Mallow's C_p and instead is solidily the second best model. This is a marginal improvement over the best subset and forward stepwise search performance as determined by Mallow's C_p. The coefficients for all model selection methods appear identical.

(e) Now fit a lasso model to the simulated data, again using $X, X^2, ..., X^{10}$ 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.

```
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.4.4
```

```
set.seed(1)
X.model.matrix = model.matrix(y ~ poly(X,10, raw = TRUE))[,-1]
lasso.cv = cv.glmnet(X.model.matrix,y)
plot(lasso.cv)
```




```
optimal.lambda = lasso.cv$lambda.min
print(paste0("The optimal lambda value is ", round(optimal.lambda,3)))
```

```
## [1] "The optimal lambda value is 0.053"
```

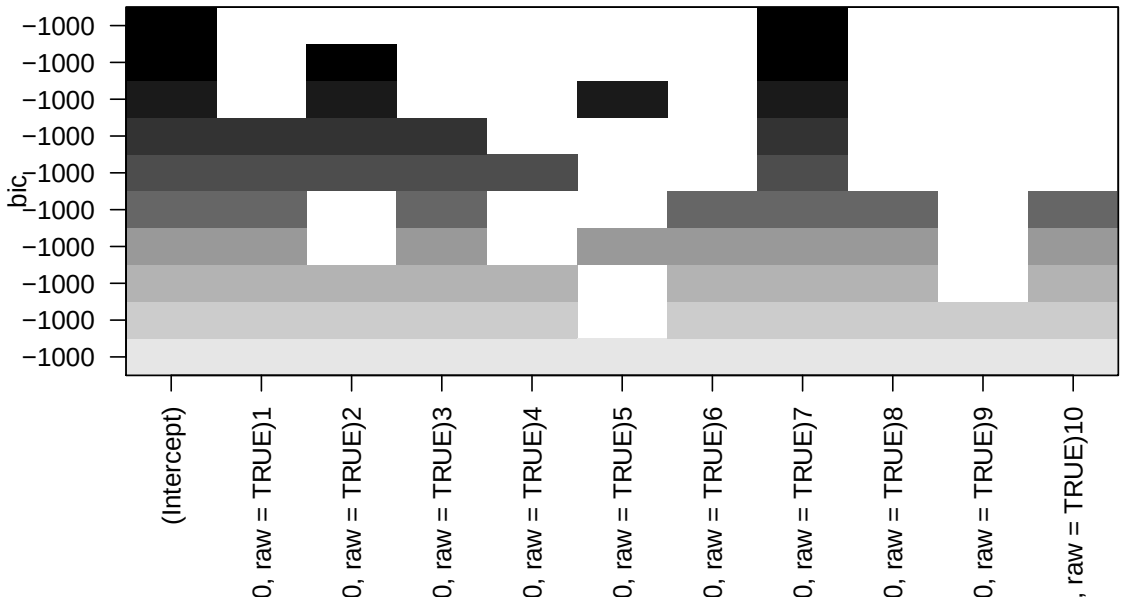
```
lasso.fit = glmnet(X.model.matrix,y,alpha = optimal.lambda)
predict(lasso.fit, type="coefficients", s=optimal.lambda)
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                                       1
## (Intercept)                4.230617e+00
## poly(X, 10, raw = TRUE)1  -1.411556e+00
## poly(X, 10, raw = TRUE)2   4.413799e+00
## poly(X, 10, raw = TRUE)3  -2.861441e+00
## poly(X, 10, raw = TRUE)4   1.148340e-01
## poly(X, 10, raw = TRUE)5  -1.893992e-01
## poly(X, 10, raw = TRUE)6   .
## poly(X, 10, raw = TRUE)7   2.979014e-05
## poly(X, 10, raw = TRUE)8  -8.132305e-06
## poly(X, 10, raw = TRUE)9   3.146140e-03
## poly(X, 10, raw = TRUE)10 -1.856886e-04
```

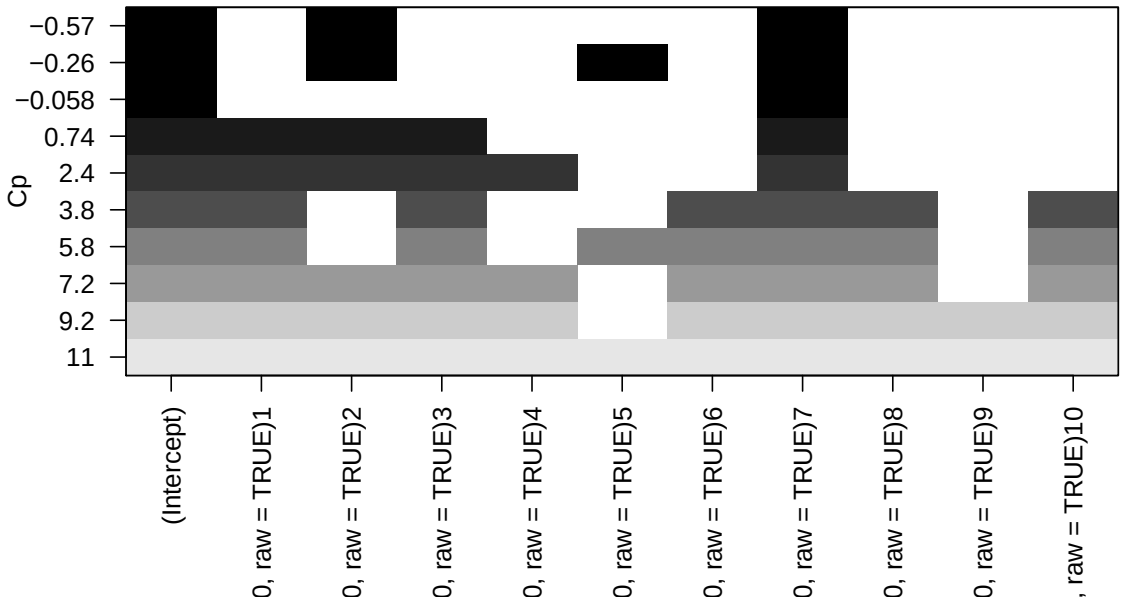
The optimal λ is 0.053. Using $\lambda = 0.053$, we obtain coefficient estimates approximate to the true cubic model. The lasso, however, does retain small coefficient values for the 4th-,6th-, and 8th-degreee polynomial terms.

(f) Now generate a response vector Y according to the model $Y = \beta_0 + \beta_7 X^7 + \epsilon$ and perform best subset selection and the lasso. Discuss the results obtained.

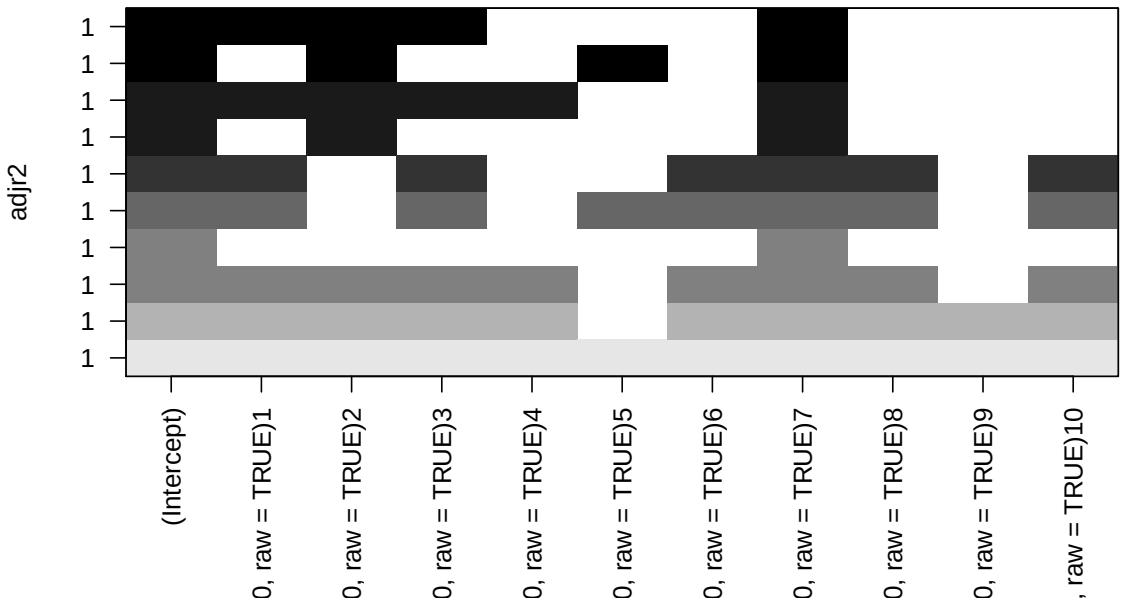
```
y2 = 5 + -3*X^7 + e
df2 = data.frame(X,y2)
#best subset
bestsub2 = regsubsets(y2 ~ poly(X,10,raw=TRUE), data=df2, nvmax=10)
plot(bestsub2, scale=c("bic"))
```




```
plot(bestsub2, scale=c("Cp"))
```



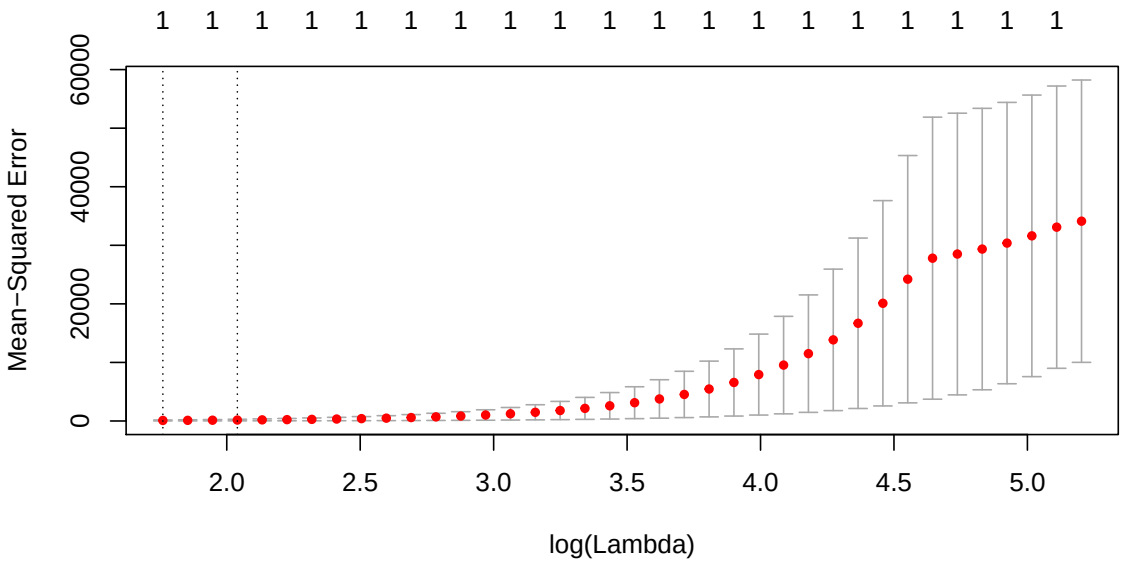

```
plot(bestsub2, scale=c("adjr2"))
```



```r
coefficients(bestsub2, id=1)
```

```
##              (Intercept) poly(X, 10, raw = TRUE)7
##                  4.95894                -2.99923
```

```r
#lasso
set.seed(1)
lasso2.cv = cv.glmnet(X.model.matrix,y2)
optimal.lambda2 = lasso2.cv$lambda.min
plot(lasso2.cv)
```




```r
print(paste0("The optimal lambda value is ", round(optimal.lambda2,3)))
```

```
## [1] "The optimal lambda value is 5.816"
```

```r
predict(lasso2.cv, type="coefficients", s=optimal.lambda2 )
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                                      1
## (Intercept)                 4.553983
## poly(X, 10, raw = TRUE)1    .
## poly(X, 10, raw = TRUE)2    .
## poly(X, 10, raw = TRUE)3    .
## poly(X, 10, raw = TRUE)4    .
## poly(X, 10, raw = TRUE)5    .
## poly(X, 10, raw = TRUE)6    .
## poly(X, 10, raw = TRUE)7   -2.903276
## poly(X, 10, raw = TRUE)8    .
## poly(X, 10, raw = TRUE)9    .
## poly(X, 10, raw = TRUE)10   .
```

We find that best subset selection using C_p, adjusted R^2, and BIC return poor results. With the exception of Mallow's C_p (which only has the true model as the third best model), both the R^2 and BIC suggest that the each of the ten best models are indistinguishable. One might default to the most parsimonious model for simplicity but the model selection tests give no strong evidence either way. The coefficient for the best one parameter model is for the X^7 term and is approximately the true value ($-2.99 \approx -3$). At $\lambda = 5.8$, the lasso regression correctly shrinks the model to the X^7 term with an approximately, though less accurate than the best subset derived coefficient estimate, accurate coefficient estimate ($-2.90 \approx -3$). The intercept calculation for the best subset model is also more accurate than the lasso estimate.

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.

```
library(ISLR)
set.seed(1)
train = sample(seq(1,nrow(College)),size=nrow(College)*.75, replace = FALSE)
college.train = College[train,]
college.test = College[-train,]
```

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

```
RMSE = function(preds, test){
  sqrt(mean((test - preds)^2))
}

lm = lm(Apps ~., data = college.train)
lm.preds = predict(lm, newdata=college.test)
RMSE(lm.preds,college.test$Apps)
```

```
## [1] 1071.1
```

The RMSE of the linear model is 1071.

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

```
set.seed(5)
college.model.matrix.train = model.matrix(Apps ~ ., data= college.train)[,-1]
college.model.matrix.test = model.matrix(Apps ~ ., data= college.test)[,-1]

grid = 10 ^ seq(4, -2, length=100)
ridge = cv.glmnet(college.model.matrix.train,
                  college.train$Apps,
                  alpha=0,
                  lambda = grid)
ridge = glmnet(college.model.matrix.train, college.train$Apps, lambda = ridge$lambda.min)
ridge.preds = predict(ridge, newx=college.model.matrix.test)
RMSE(ridge.preds,college.test$Apps)
```

```
## [1] 1070.841
```

The RMSE of the ridge regression modle is 1071.

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

```
set.seed(10)
lasso = cv.glmnet(college.model.matrix.train,
                  college.train$Apps,
                  alpha=1,
                  lambda = grid)
lasso = glmnet(college.model.matrix.train,
               college.train$Apps,
               lambda =lasso$lambda.min)
lasso.preds = predict(lasso, newx=college.model.matrix.test)
RMSE(lasso.preds,college.test$Apps)
```

```
## [1] 1069.711
```

```
predict(lasso, type="coefficients", s=lasso$lambda.min)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                         s0
## (Intercept) -591.14556119
## PrivateYes  -340.13143520
## Accept         1.45465948
## Enroll        -0.15043566
## Top10perc     29.20277468
## Top25perc      .
## F.Undergrad    .
## P.Undergrad    .
## Outstate      -0.03918920
## Room.Board     0.11033302
## Books          .
## Personal       .
## PhD           -1.62611543
## Terminal      -5.92881731
## S.F.Ratio      .
## perc.alumni   -2.46575052
## Expend         0.05746676
## Grad.Rate      3.48794578
```

The RMSE pf the lasso regression modle is 1070. The lasso regression removed six variables from the full model.

(e) 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.

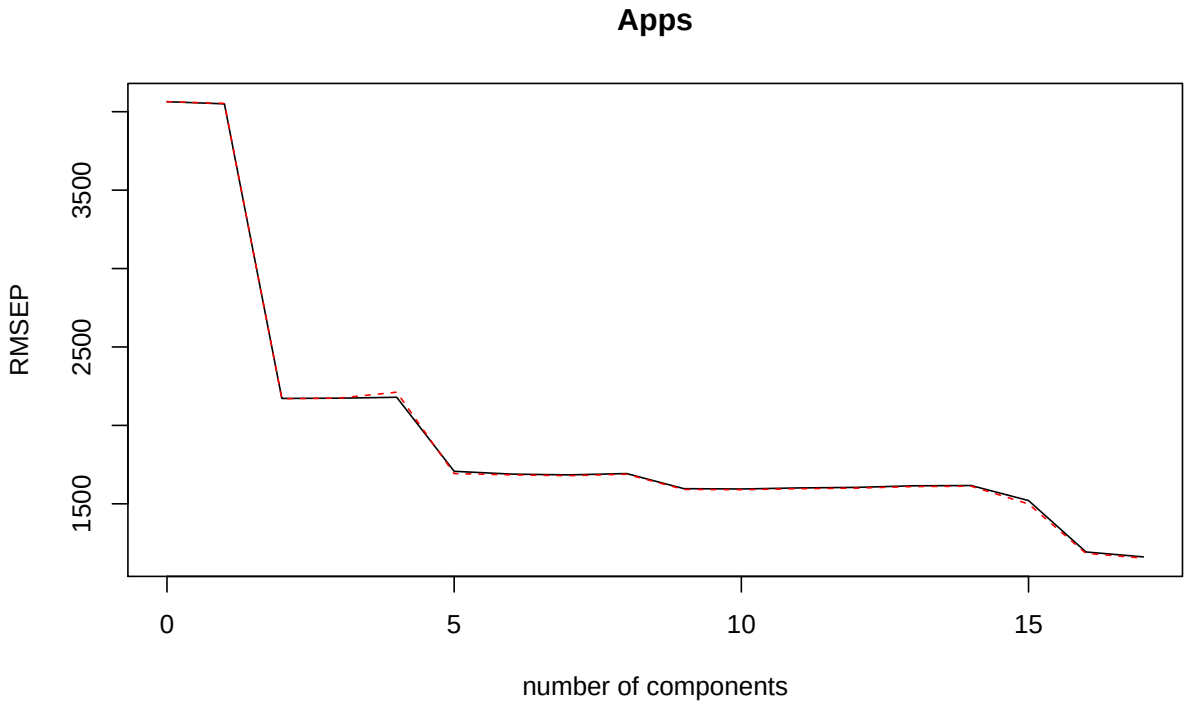
```
library(pls)
```

```
## Warning: package 'pls' was built under R version 3.4.4
```

```
set.seed(1)
pcr = pcr(Apps ~., data = college.train,
          scale=TRUE,
          validation="CV",
          segments = 10)
pcr$ncomp
```

```
## [1] 17
```

```
validationplot(pcr)
```




```
pcr.preds = predict(pcr, newdata = college.test)
RMSE(pcr.preds,college.test$Apps)
```

```
## [1] 1547.992
```

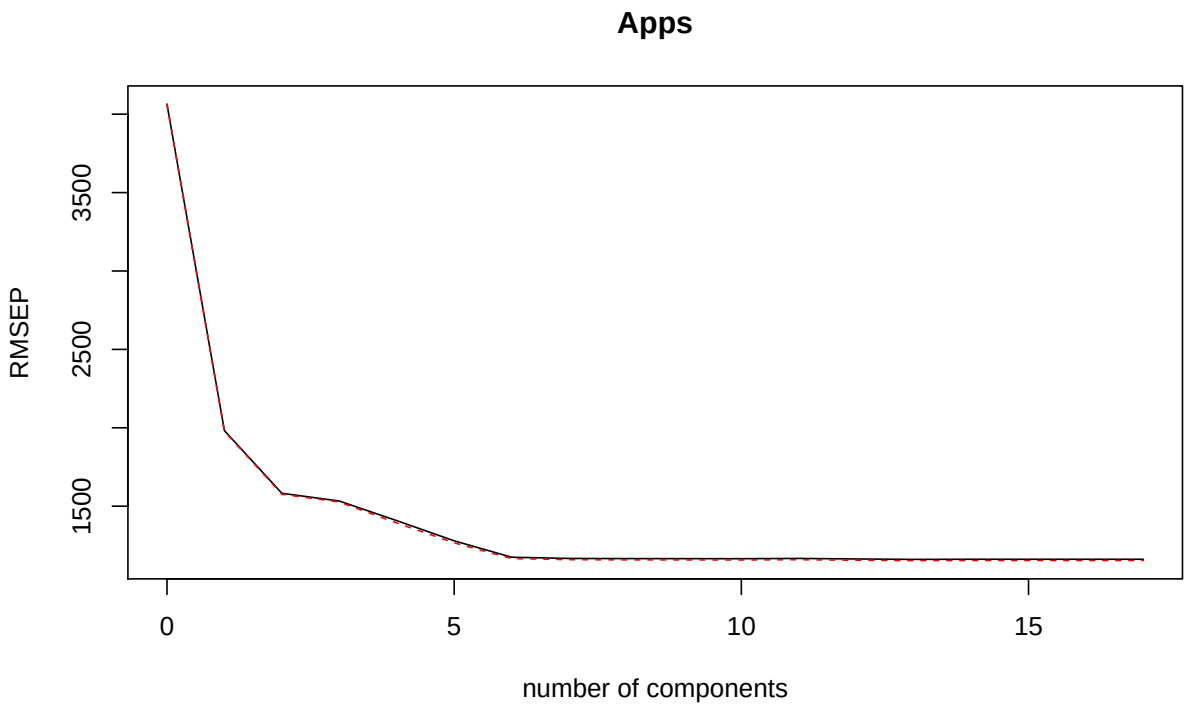
The PCR regression model selected 17 components with k=10 cross-validaiton. The RMSE of the model is 1548.

(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.

```
set.seed(1)
pls = plsr(Apps ~., data = college.train,
           scale=TRUE,
           validation="CV",
           segments = 10)
pls$ncomp
```

```
## [1] 17
```

```
validationplot(pls)
```




```
pls.preds = predict(pls, newdata = college.test)
RMSE(pls.preds,college.test$Apps)
```

```
## [1] 1155.772
```

The PLS regression model selected 17 components with k=10 cross-validaiton. The RMSE of the model is 1156.

(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?

```
test_rmse = c(RMSE(lm.preds,college.test$Apps),
              RMSE(ridge.preds,college.test$Apps),
              RMSE(lasso.preds,college.test$Apps),
              RMSE(pcr.preds,college.test$Apps),
              RMSE(pls.preds,college.test$Apps))

ridge_residuals = predict(ridge, newx=college.model.matrix.train) - college.train$Apps
lasso_residuals = predict(lasso, newx=college.model.matrix.train) - college.train$Apps
train_rmse = c(sqrt(mean(lm$residuals^2)),
               sqrt(mean(ridge_residuals^2)),
               sqrt(mean(lasso_residuals^2)),
               sqrt(mean(pcr$residuals^2)),
               sqrt(mean(pls$residuals^2)))

table5 = data.frame("Train RMSE" = train_rmse, "Test RMSE" = test_rmse)
row.names(table5) = c("OLS", "Ridge", "Lasso", "PCR", "PLS")
knitr::kable(table5, caption ="Comparison of College Application Models")
```

Table 2: Comparison of College Application Models

	Train.RMSE	Test.RMSE
OLS	1024.205	1071.100
Ridge	1024.214	1070.841
Lasso	1058.957	1069.711
PCR	1844.110	1547.992
PLS	1169.703	1155.772

The lasso, ridge, and OLS regressions all had similar root means square test errors. Using this approach, the PCR regression performed the worst following the PLS regression.

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, $n = 1000$ observations, and an associated quantitative response vector generated according to the model $Y = X\beta + \epsilon$ where β has some elements that are exactly zero.

```
library(mvtnorm)
set.seed(10)
mu = rep(0,20)
sigma = diag(1,20)
X = rmvnorm(1000,mean = mu, sigma = sigma)
betas = sample(-10:10, size = 20, replace=TRUE)
betas[10] = 0
betas[17] = 0
betas
```

```
##  [1]   7   8   0   0   9  -6   4  -6  -7   0  -9  -6  10   7 -10  10   0
## [18]   8  -3   1
```

```
e = rnorm(n = 1000, 0, sd=1)
y = X%*%betas + e
df.sim = data.frame(x = X, y = y)
```

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

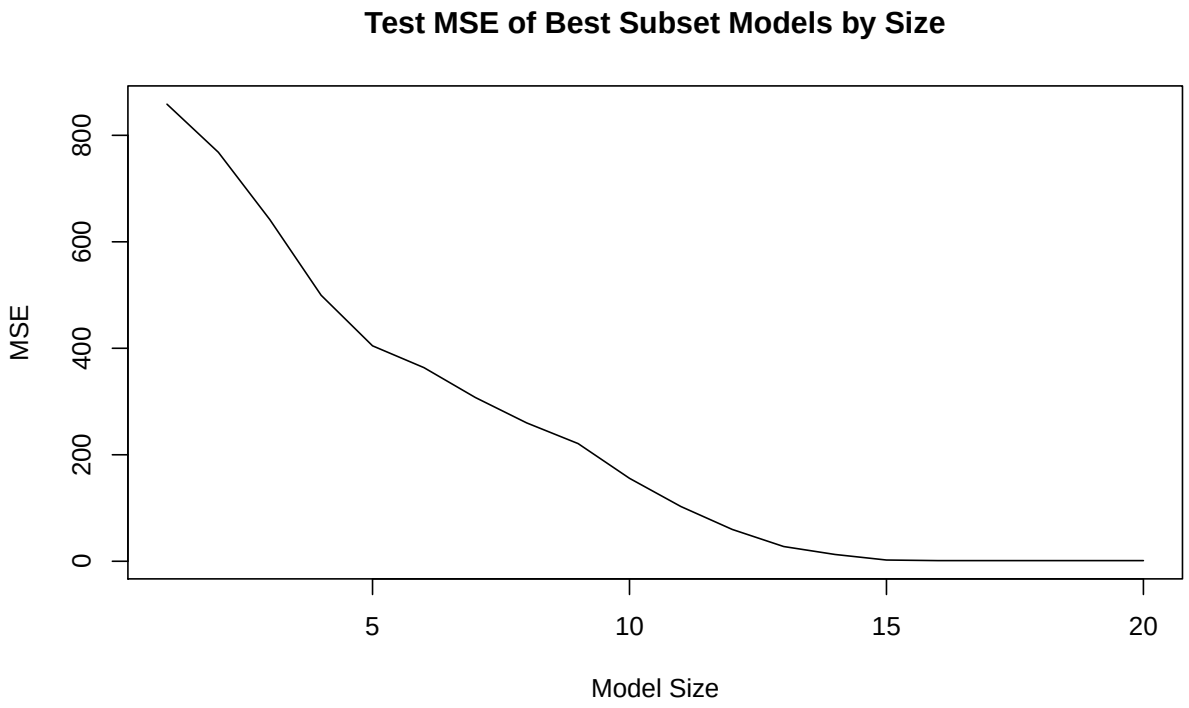
```
train = sample(1:1000, size = length(y)*.75, replace = FALSE)
df.sim.train = df.sim[train,]
df.sim.test = df.sim[-train,]
```

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

```
bestsub.sim = regsubsets(y ~ .,data=df.sim.train, nvmax = 20)
plot(1:21, bestsub.sim$rss/length(df.sim.train$y),
     xlab = "Model Size",
     ylab = "MSE",
     main = "Training MSE of Best Subset Models by Size",
     type ="l")
```

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

```
MSE = function(preds, test){
  mean((test - preds)^2)
}

predict.regsubsets = function(object, newdata, id, ...) {
    form = as.formula(object$call[[2]])
    mat = model.matrix(form, newdata)
    coefi = coef(object, id = id)
    mat[, names(coefi)] %*% coefi
}
bestsub.mse = rep(NA,20)
for (i in 1:20){
  bestsub.preds = predict.regsubsets(bestsub.sim, df.sim.test, id=i)
  bestsub.mse[i] = MSE(bestsub.preds,df.sim.test$y)
}

plot(1:20, bestsub.mse,
     xlab = "Model Size",
     ylab = "MSE",
     main = "Test MSE of Best Subset Models by Size",
     type ="l")
```



(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.

```
which.min(bestsub.mse)
```

```
## [1] 16
```

Model size 16 has the smallest test MSE.

(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.

```
betas
```

```
##  [1]   7   8   0   0   9  -6   4  -6  -7   0  -9  -6  10   7 -10  10   0
## [18]   8  -3   1
```

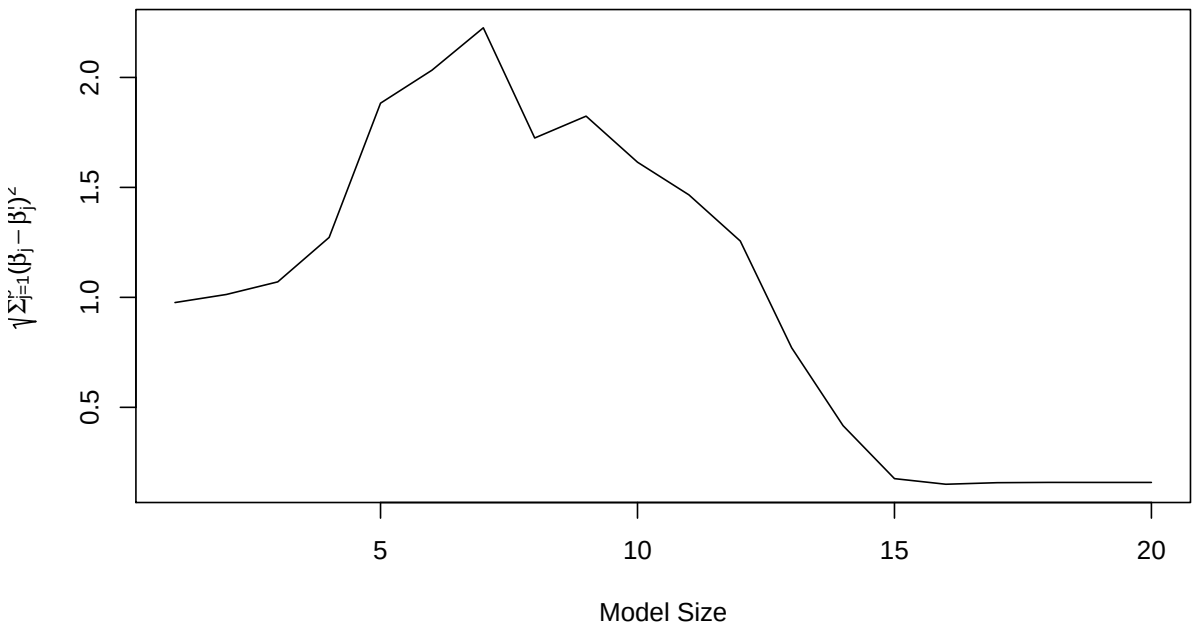
```
coefficients(bestsub.sim, id=16)
```

```
## (Intercept)         x.1         x.2         x.5         x.6         x.7
##  -0.0197531   6.9362157   7.9618777   9.0262420  -6.0277104   4.0069658
##         x.8         x.9        x.11        x.12        x.13        x.14
##  -6.0382272  -7.0007108  -8.9791852  -5.9971759   9.9618828   7.0549651
##        x.15        x.16        x.18        x.19        x.20
## -10.0349740  10.0541292   7.9916932  -3.0584152   0.9610219
```

In the true model X_3, X_4, X_{10}, and X_{17} have coefficients of zero. The best subset for size 16 removes these variables from the model and the coefficient. The coefficient estimates are appoximately the same as the true model values.

(g) Create a plot displaying $\sqrt{\Sigma_{j=1}^{p}(\beta_j - \hat{\beta}_j^r{}^2}$ 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 his compare to the MSE plot from (d)?

```
beta_error = rep(NA,20)
for (i in 1:20){
  vars = names(df.sim) %in% names(coefficients(bestsub.sim, id=i))
  est_coef = coefficients(bestsub.sim, id=i)[-1]
  true_coef = betas[names(df.sim) %in% names(coefficients(bestsub.sim, id=i))]
  beta_error[i] = sqrt(sum((true_coef - est_coef)^2))
}
plot(1:20, beta_error, xlab = "Model Size",
                       ylab = latex2exp("$\\sqrt{\\Sigma_{j=1}^p (\\beta_j - \\hat{\\beta_j^r)^2}$"),
                       type="l")
```




```
which.min(beta_error)
```

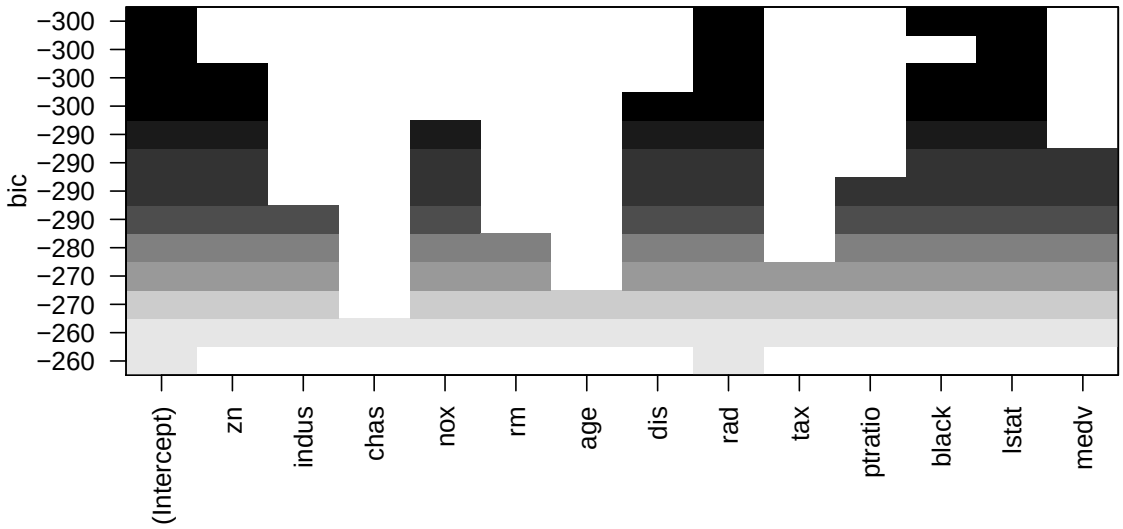
```
## [1] 16
```

This plot increases substantially and then deceases, in contrast to the plot in part (d) which is monotonically decreasing for much of the plot until it hits its minimum and then rises slightly. This plot hits a minimum at model size 16 so both plots confirm that model size 16 is optimal.

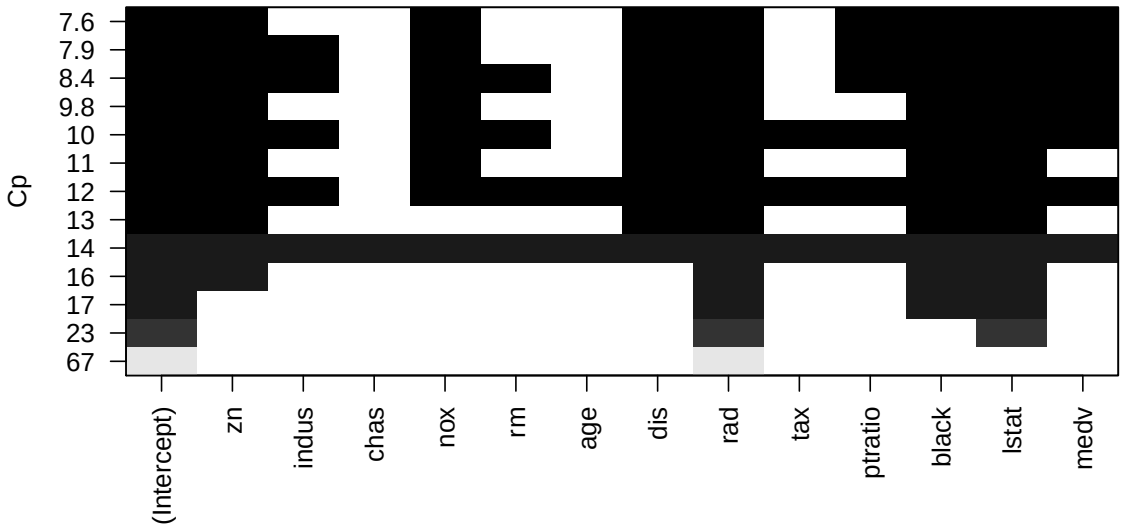
11. We will now try to predict per capita crime rate in the Boston data set.

(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.

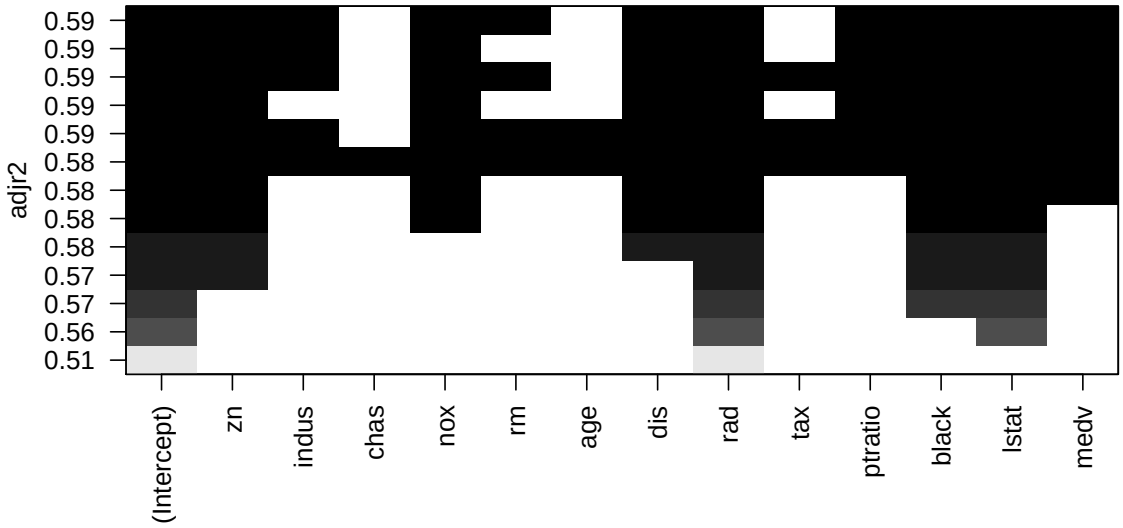
```
library(MASS)
test.rmse = rep(NA,5)

#training and test set
set.seed(500)
train = sample(1:nrow(Boston), size =nrow(Boston)*.75, replace=FALSE)
boston.train = Boston[train,]
boston.test = Boston[-train,]

#best subset
boston.bestsub = regsubsets(crim ~., data = boston.train, nvmax = ncol(Boston)-1)
plot(boston.bestsub)
```




```
plot(boston.bestsub, scale="Cp")
```




```
plot(boston.bestsub, scale="adjr2")
```




```
#test rmse
boston.bestsub.preds = predict(boston.bestsub, newdata=boston.test, id=3)
```

```
test.rmse[1] = RMSE(boston.bestsub.preds,boston.test$crim)
```

```
#ridge
set.seed(1)
boston.X.train = model.matrix(crim ~., data=boston.train)[,-1]
boston.X.test = model.matrix(crim ~., data=boston.test)[,-1]
boston.ridge = cv.glmnet(x = boston.X.train, y=boston.train$crim, alpha=0, lambda=grid)
boston.ridge = glmnet(x =boston.X.train, y=boston.train$crim, lambda=boston.ridge$lambda.min)

#test rmse
boston.ridge.preds = predict(boston.ridge, newx=boston.X.test)
test.rmse[2] = RMSE(boston.ridge.preds,boston.test$crim)

#lasso
set.seed(1)
boston.lasso = cv.glmnet(x = boston.X.train, y=boston.train$crim, alpha=1, lambda=grid)
boston.lasso = glmnet(x =boston.X.train, y=boston.train$crim, lambda=boston.lasso$lambda.min)

#test rmse
boston.lasso.preds = predict(boston.lasso, newx=boston.X.test)
test.rmse[3] = RMSE(boston.lasso.preds,boston.test$crim)

#PCR
set.seed(340)
boston.pcr = pcr(crim ~., data = boston.train,
                 scale=TRUE,
                 validation="CV",
                 segments = 10)

boston.pcr.preds = predict(boston.pcr, newdata=boston.test)
test.rmse[4] = RMSE(boston.pcr.preds,boston.test$crim)

#PCR
set.seed(450)
boston.pls = plsr(crim ~., data = boston.train,
                  scale=TRUE,
                  validation="CV",
                  segments = 10)

boston.pls.preds = predict(boston.pcr, newdata=boston.test)
test.rmse[5] = RMSE(boston.pcr.preds,boston.test$crim)
```

(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.

```
test.rmse = data.frame(matrix(round(test.rmse,3), nrow =1, ncol=5))
colnames(test.rmse) = c("Best Subset", "Ridge", "Lasso", "PCR", "PLS")
knitr::kable(test.rmse, caption = "Model Performance")
```

Best Subset	Ridge	Lasso	PCR	PLS

Table 3: Model Performance

Best Subset	Ridge	Lasso	PCR	PLS
10.514	10.486	10.428	10.682	10.682

The model with the lowerst test root mean squared error is the lasso regression model.

(c) Does your chosen model involve all of the features in the data set? Why or why not?

```
coefficients(boston.lasso)
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                        s0
## (Intercept) 14.9493571544
## zn           0.0315132756
## indus       -0.0706752460
## chas        -0.0937541455
## nox         -7.1350548359
## rm          -0.4910409277
## age          0.0003287303
## dis         -0.7175219549
## rad          0.4745362124
## tax          .
## ptratio     -0.2244818733
## black       -0.0071540697
## lstat        0.1733545096
## medv        -0.0675067058
```

The chosen lasso model has removed the tax variable. This is because the model uses penalized regression to shrink paramters to zero that are not predictive of the response variable.