Annand Module 05 Lab 01

Joseph Annand

2023-11-25

Load Libraries

```r
library(ISLR2)
library(MASS)
```

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

## The following object is masked from 'package:ISLR2':
##
##     Boston
```

```r
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 4.3.2
```

```r
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 4.3.2

## Loading required package: Matrix

## Loaded glmnet 4.1-8
```

```r
library(pls)
```

```
## Warning: package 'pls' was built under R version 4.3.2

##
## Attaching package: 'pls'

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

Question 9

Part A - Prepare the Data

```r
# Load Dataset
college.data <- College
college.data <- na.omit(college.data)

# Create vector half the size of college.data that contains random set of indices
set.seed(1)
train <- sample(nrow(college.data), 0.8 * nrow(college.data))

# Initialize training and test data
college.train <- college.data[train, ]
college.test <- college.data[-train, ]
```

Part B - Least Squares Regression

```r
lm.college <- lm(Apps ~ ., data = college.data, subset = train)

summary(lm.college)
```

```
##
## Call:
## lm(formula = Apps ~ ., data = college.data, subset = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5555.2  -404.6    19.9   310.3  7577.7
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -630.58238  435.56266  -1.448 0.148209
## PrivateYes  -388.97393  148.87623  -2.613 0.009206 **
## Accept         1.69123    0.04433  38.153  < 2e-16 ***
## Enroll        -1.21543    0.20873  -5.823 9.41e-09 ***
## Top10perc     50.45622    5.88174   8.578  < 2e-16 ***
## Top25perc    -13.62655    4.67321  -2.916 0.003679 **
## F.Undergrad    0.08271    0.03632   2.277 0.023111 *
## P.Undergrad    0.06555    0.03367   1.947 0.052008 .
## Outstate      -0.07562    0.01987  -3.805 0.000156 ***
## Room.Board     0.14161    0.05130   2.760 0.005947 **
## Books          0.21161    0.25184   0.840 0.401102
## Personal       0.01873    0.06604   0.284 0.776803
## PhD           -9.72551    4.91228  -1.980 0.048176 *
## Terminal      -0.48690    5.43302  -0.090 0.928620
## S.F.Ratio     18.26146   13.83984   1.319 0.187508
## perc.alumni    1.39008    4.39572   0.316 0.751934
## Expend         0.05764    0.01254   4.595 5.26e-06 ***
## Grad.Rate      5.89480    3.11185   1.894 0.058662 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 993.8 on 603 degrees of freedom
## Multiple R-squared:  0.9347, Adjusted R-squared:  0.9328
```

```
## F-statistic: 507.5 on 17 and 603 DF,  p-value: < 2.2e-16
```

```r
lm.predict <- predict(lm.college, college.test)

lm.mse <- mean((lm.predict - college.test$Apps)^2)
lm.mse
```

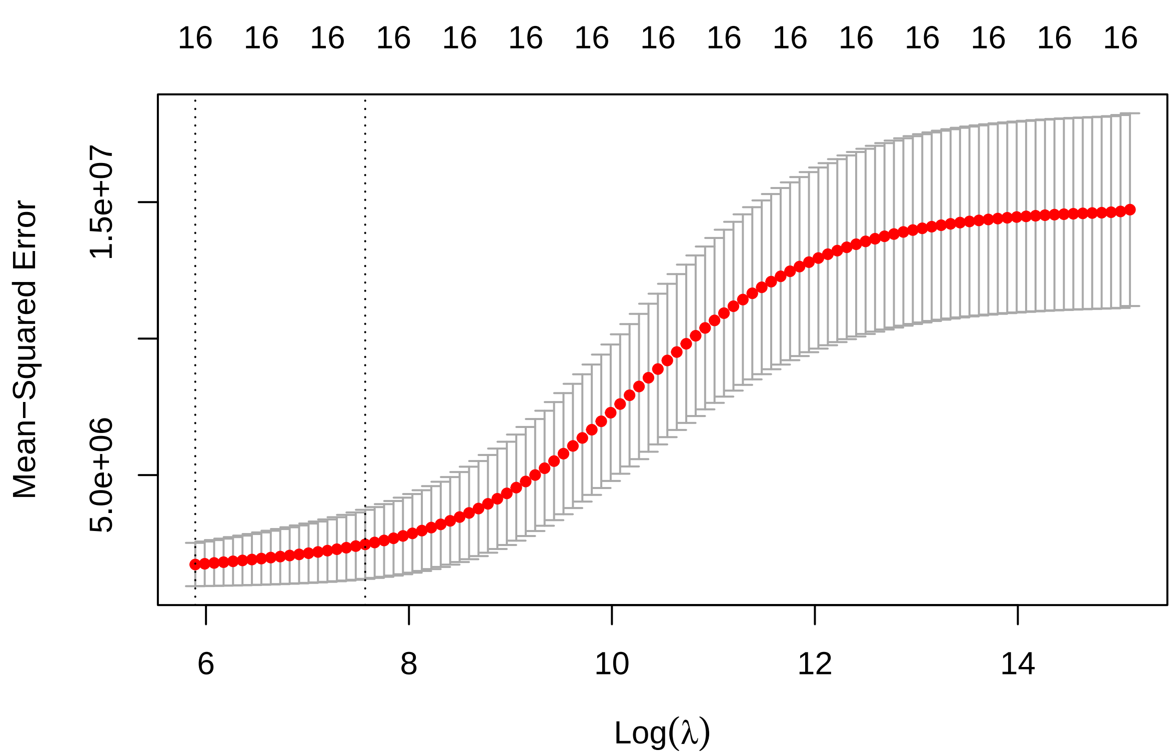
```
## [1] 1567324
```

Part C - Ridge Regression

```r
# Create matrix of x, the predictors, and vector of y, the response
x <- model.matrix(Apps ~ ., college.data)[, -2]
y <- college.data$Apps

# Create a lambda grid and use it to form ridge regression model
lambda.grid <- 10^seq(10, -2, length = 100)
ridge.mod <- glmnet(x[train, ], y[train], alpha = 0, lambda = lambda.grid, thresh = 1e-12)

# Determine best lambda, or tuning parameter, using cross-validation
set.seed(2)
cv.out <- cv.glmnet(x[train, ], y[train], alpha = 0)
plot(cv.out)
```



```r
bestlam <- cv.out$lambda.min
bestlam
```

```
## [1] 362.9786
```

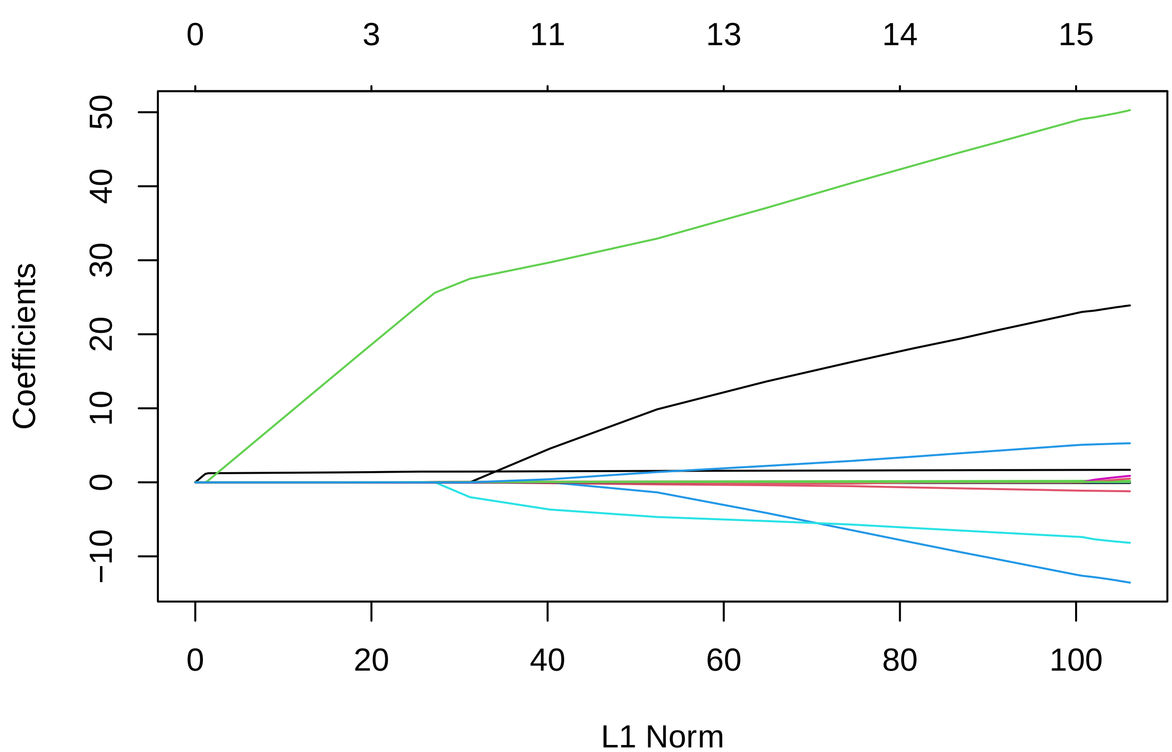
```r
# Predict response of test data using ridge regression and calculate MSE
ridge.pred <- predict(ridge.mod, s = bestlam, newx = x[-train, ])
ridge.mse <- mean((ridge.pred - y[-train])^2)
ridge.mse
```

```
## [1] 1508918
```

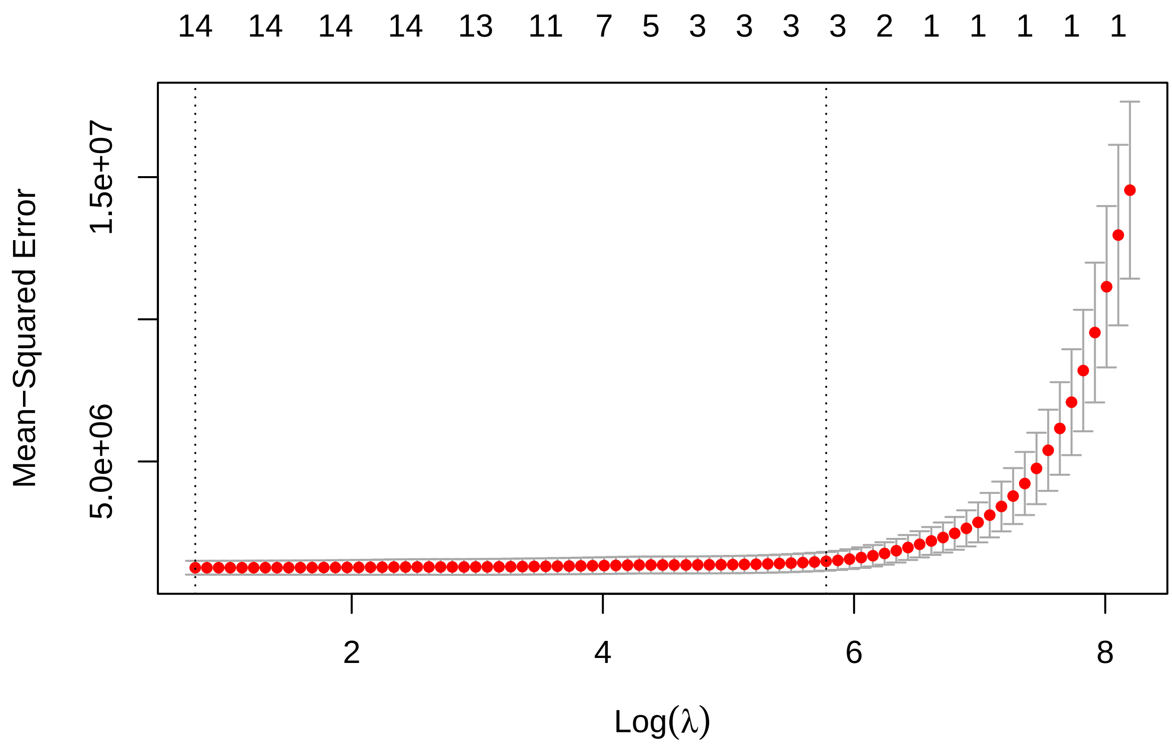
Part D - The Lasso

```r
# Create lasso model on the college dataset
lasso.mod <- glmnet(x[train, ], y[train], alpha = 1, lambda = lambda.grid)
plot(lasso.mod)
```

```
## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values
```



```r
# Perform cross-validation to determine best lambda or tuning parameter
set.seed(3)
cv.out <- cv.glmnet(x[train, ], y[train], alpha = 1)
plot(cv.out)
```



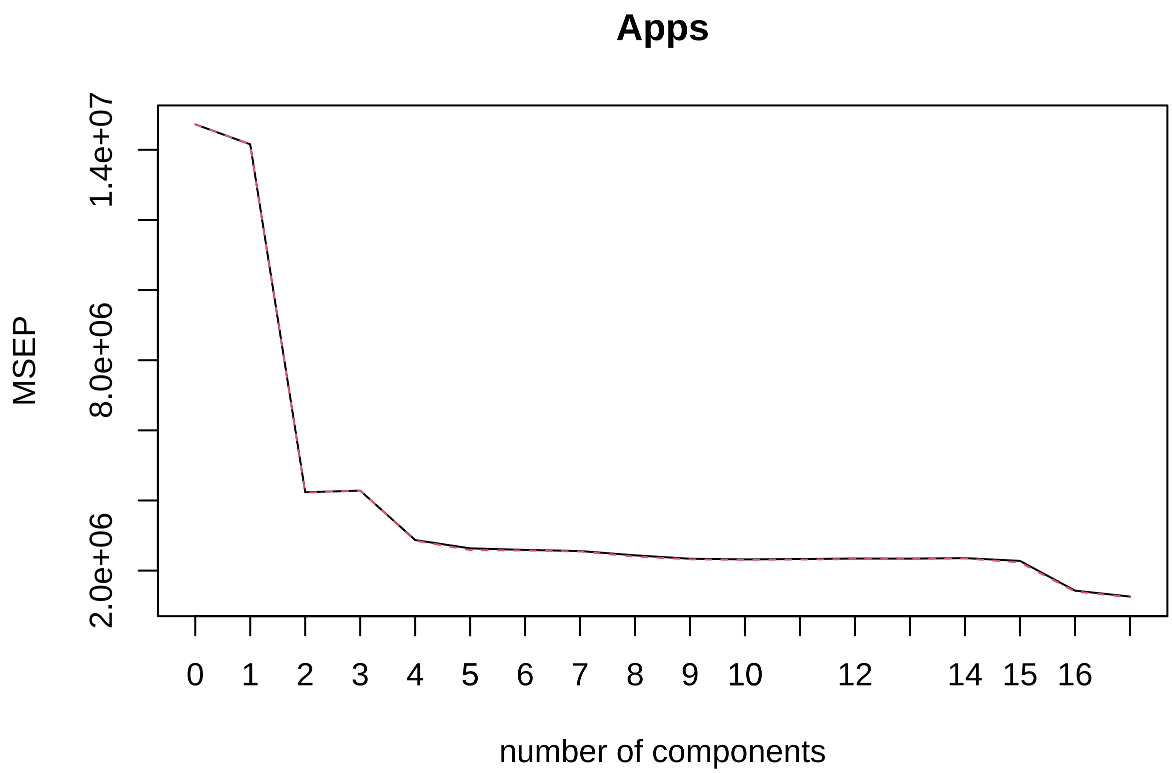

```r
bestlam2<- cv.out$lambda.min

# Predict response of test data and calculate MSE
lasso.pred <- predict(lasso.mod, s = bestlam2, newx = x[-train, ])
lasso.mse <- mean((lasso.pred - y[-train])^2)
lasso.mse
```

```
## [1] 1593402
```

Part E - PCR

```r
# Create PCR model on training data
set.seed(4)
pcr.fit <- pcr(Apps ~ ., data = college.data, subset = train, scale = T,
               validation = "CV")
validationplot(pcr.fit, val.type = "MSEP")
axis(side=1, at=seq(1, 20, by=1))
```



Using the cross-validation method, the lowest error occurs when M=17.

```
# Predict the number of applications using the PCR model
pcr.pred <- predict(pcr.fit, x[-train, ], ncomp = 17)
pcr.mse <- mean((pcr.pred - y[-train])^2)
pcr.mse
```

```
## [1] 1598902
```

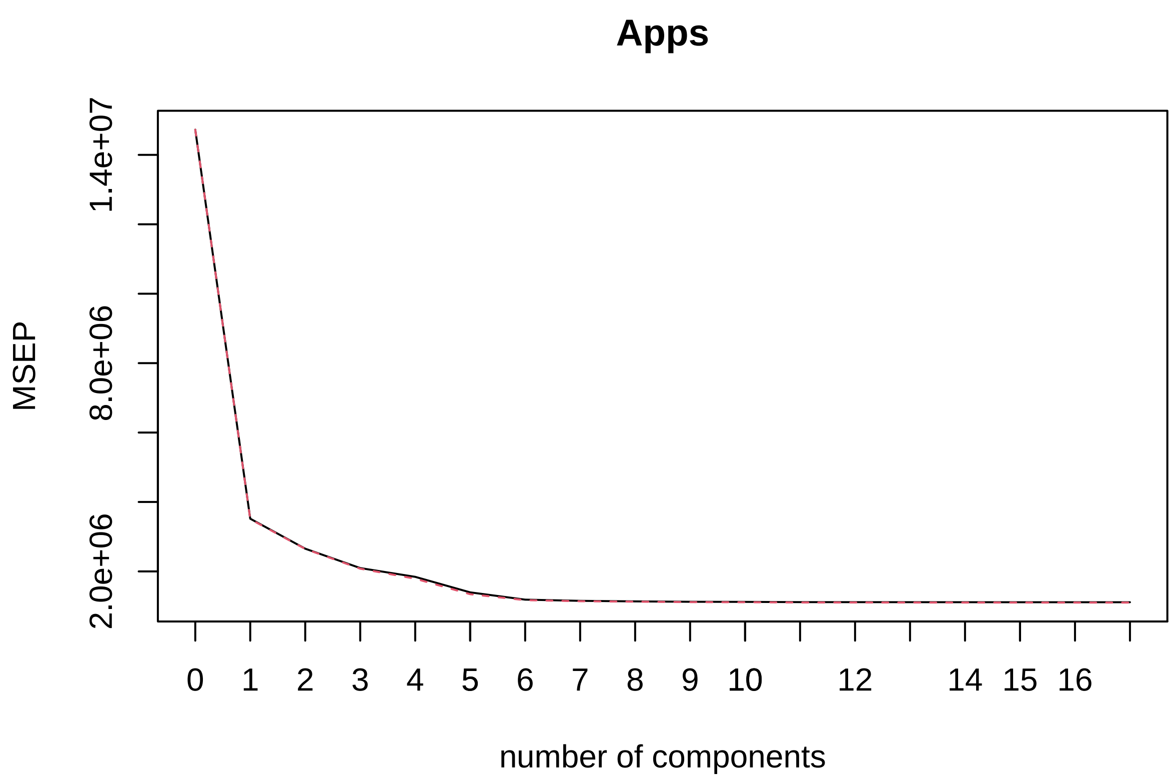
Part F - Partial Least Squares

```
# Create PLS model on the college data
set.seed(5)
pls.fit <- plsr(Apps ~ ., data = college.data, subset = train, scale = T,
                validation = "CV")
summary(pls.fit)
```

```
## Data:    X dimension: 621 17
##  Y dimension: 621 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            3837     1876     1629     1448     1358     1181     1089
## adjCV         3837     1872     1627     1441     1338     1159     1082
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        1073     1066     1060      1059      1056      1056      1055
## adjCV     1068     1062     1056      1054      1051      1051      1050
##        14 comps  15 comps  16 comps  17 comps
## CV         1055      1054      1054      1054
## adjCV      1050      1050      1050      1050
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       25.55    45.38    62.59    65.08    67.55    72.02    75.93    80.46
## Apps    77.30    83.57    87.51    90.88    92.88    93.15    93.24    93.31
##       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X       82.51     85.43     87.83     91.09     92.73     95.12     96.95
## Apps    93.39     93.42     93.45     93.46     93.47     93.47     93.47
##       16 comps  17 comps
## X        97.97    100.00
## Apps     93.47     93.47
```

```r
validationplot(pls.fit, val.type = "MSEP")
axis(side=1, at=seq(1, 20, by=1))
```



```r
# Predict number of applications using PLS
pls.pred <- predict(pls.fit, x[-train, ], ncomp = 7)
pls.mse <- mean((pls.pred - y[-train])^2)
pls.mse
```

```
## [1] 1528394
```

Part G

```r
mse.models <- data.frame(
  model = c("least.squares", "ridge.regression", "lasso", "pcr", "pls"),
  mse = c(lm.mse, ridge.mse, lasso.mse, pcr.mse, pls.mse),
  stringsAsFactors = F
)
mse.models
```

```
##              model     mse
## 1    least.squares 1567324
## 2 ridge.regression 1508918
## 3            lasso 1593402
## 4              pcr 1598902
## 5              pls 1528394
```

Our models adequately explain the relationship between the response and the predictors. the initial least squares regression had an adjusted R-squared of 0.94 and a F-statistic over 500, which indicates that the least squares regression can be used to predict the number of applications received. Several different approaches yield a smaller test error than least squares, meaning that these approaches are even better at predicting the number of applications. The test error for PCR is significantly lower than those of the other four approaches. The other four approaches yield similar test errors to each other.

Question 11

```r
boston.data <- Boston
boston.data <- na.omit(boston.data)

# Create vector half the size of college.data that contains random set of indices
set.seed(7)
train <- sample(c(TRUE, FALSE), nrow(boston.data), replace = T)
test <- (!train)
```

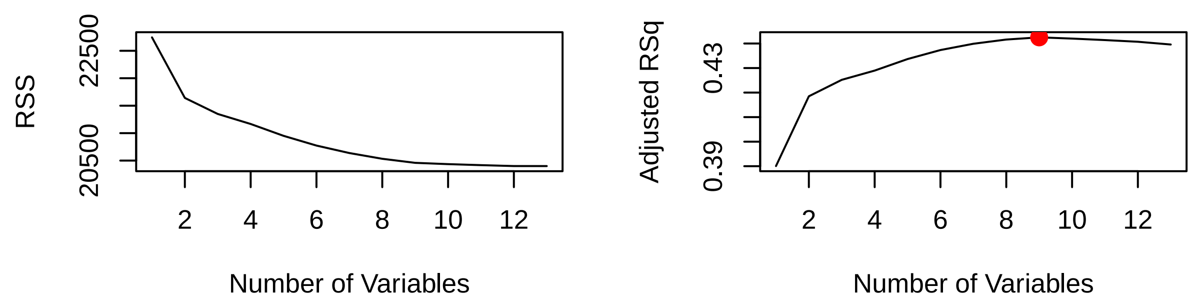
Part A - Explore Different Models with Boston data

```r
# Best Subset Selection
regfit.full <- regsubsets(crim ~ ., data = boston.data, nvmax = 13)
reg.summary <- summary(regfit.full)
```

```r
par(mfrow = c(2,2))
plot(reg.summary$rss, xlab = "Number of Variables", ylab = "RSS", type = "l")
plot(reg.summary$adjr2, xlab = "Number of Variables", ylab = "Adjusted RSq",
     type = "l")
which.max(reg.summary$adjr2)
```

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

```r
points(9, reg.summary$adjr2[9], col = "red", cex = 2, pch = 20)
```




```r
coef(regfit.full, 9)
```

```
##   (Intercept)            zn         indus           nox           dis
##  19.124636156   0.042788127  -0.099385948 -10.466490364  -1.002597606
##           rad       ptratio         black         lstat          medv
##   0.539503547  -0.270835584  -0.008003761   0.117805932  -0.180593877
```

```r
# Ridge Regression

x2 <- model.matrix(crim ~ ., boston.data)[, -1]
y2 <- boston.data$crim

set.seed(8)
```

```r
cv.out <- cv.glmnet(x2, y2, alpha = 0)
bestlam3 <- cv.out$lambda.min

out <- glmnet(x2, y2, alpha = 0)
predict(out, type = "coefficients", s = bestlam3)[1:13,]
```

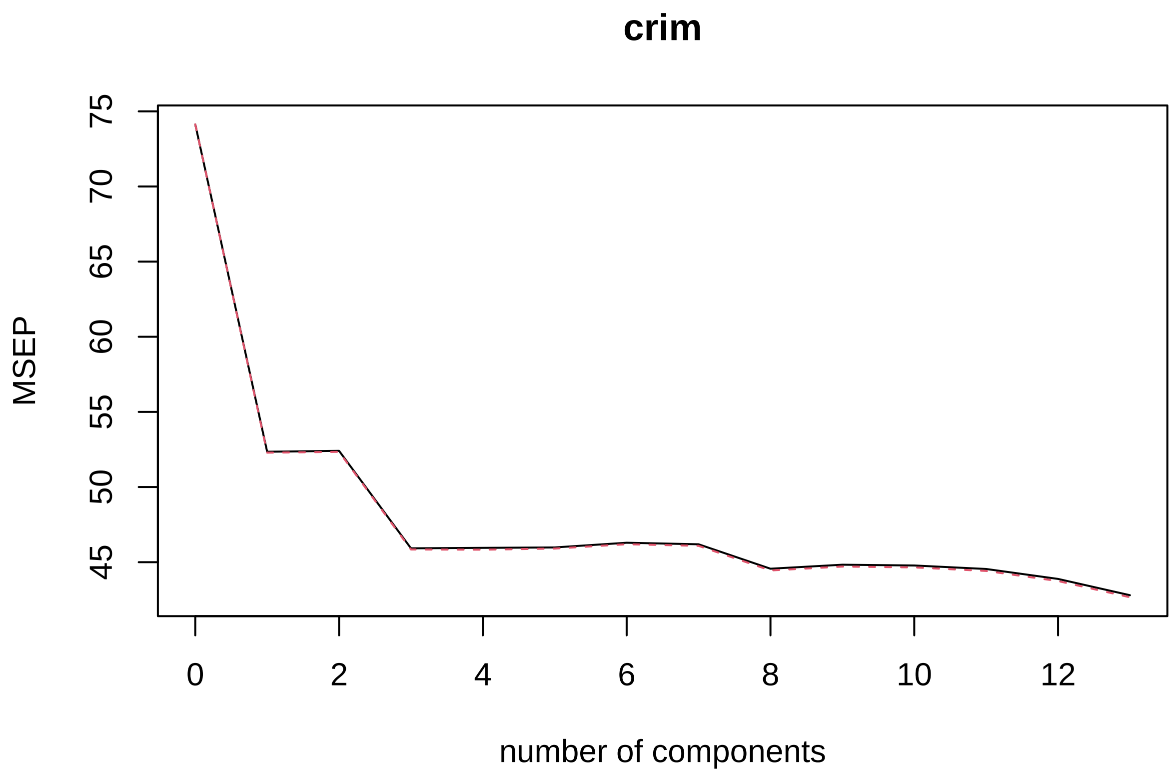
```
##  (Intercept)           zn        indus         chas          nox           rm
##  9.063048666  0.033002416 -0.082046152 -0.737684583 -5.393098508  0.335972073
##          age          dis          rad          tax      ptratio        black
##  0.001962473 -0.702123643  0.422779055  0.003400607 -0.135911587 -0.008483285
##        lstat
##  0.142613436
```

```r
# The Lasso
set.seed(9)
cv.out <- cv.glmnet(x2, y2, alpha = 1)
bestlam4 <- cv.out$lambda.min

out <- glmnet(x2, y2, alpha = 1, lambda = lambda.grid)
lasso.coef <- predict(out, type = "coefficients", s = bestlam4)[1:13,]
lasso.coef
```

```
##   (Intercept)            zn         indus          chas           nox
## 13.3232201680  0.0372265059 -0.0727645312 -0.6003296373 -7.5712246355
##            rm           age           dis           rad           tax
##  0.2670289551  0.0000000000 -0.8227122775  0.5195543611 -0.0001276602
##       ptratio         black         lstat
## -0.2029308099 -0.0075454075  0.1258873002
```

```r
# PCR
set.seed(11)
pcr.boston <- pcr(crim ~ ., data = boston.data, scale = T,
                  validation = "CV")
validationplot(pcr.boston, val.type = "MSEP")
```




```
# Fit PCR to entire data set using M = 8
pcr.boston <- pcr(y2 ~ x2, scale = T, ncomp = 5)
summary(pcr.boston)
```

```
## Data:    X dimension: 506 13
##  Y dimension: 506 1
## Fit method: svdpc
## Number of components considered: 5
## TRAINING: % variance explained
##     1 comps  2 comps  3 comps  4 comps  5 comps
## X     47.70    60.36    69.67    76.45    82.99
## y2    30.69    30.87    39.27    39.61    39.61
```

Part B - Propose Model for Predicting Crime Rate

```
# Use Validation-Set Approach to Determine Best Subset Selection Model
regfit.best <- regsubsets(crim ~ ., data = boston.data[train, ], nvmax = 13)

# Create test matrix
test.mat <- model.matrix(crim ~ ., data = boston.data[test, ])

# Compute test MSE for all possible amounts of variables used in the model
val.errors <- rep(NA, 13)
```

```r
for (i in 1:13) {
  coefi <- coef(regfit.best, id = i)
  pred <- test.mat[, names(coefi)] %*% coefi
  val.errors[i] <- mean((boston.data$crim[test] - pred)^2)
}

# Get coefficient estimates for model with best subset of variables
best.subset <- which.min(val.errors)
coef(regfit.best, best.subset)
```

```
##   (Intercept)            zn           nox           age           dis
##  21.130421882   0.064159542 -10.950157410   0.030010983  -1.034916464
##           rad           tax       ptratio         black          medv
##   0.705005168  -0.007878952  -0.306715346  -0.005776214  -0.262236529
```

Using the validation set approach on a best subset selection method, a model containing seven predictors was determined to have the lowest MSE of all combinations.

Part C

The chosen model does not involve all features because the best subset selection method was used and with a validation set approach, we determined that the model with the lowest test MSE used only seven of the thirteen possible predictors of crime rate per capita.