# STATS 415 - Homework 6

*Marian L. Schmidt*

*March 11, 2016*

## Question 1

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

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

$$\text{subject to} \quad \sum_{j=1}^{p}|\beta_j| \leq s$$

Figure 1:

**for a particular value of s. For parts (a) through (d), indicate which of i. through v. is correct.**

**i. Increase initially, and then eventually start decreasing in an inverted U shape.**
**ii. Decrease initially, and then eventually start increasing in a U shape.**
**iii. Steadily increase.**
**iv. Steadily decrease.**
**v. Remain constant.**

**(a) As we increase s from 0, the training RSS will:**
*iv. Steadily decreases. As we increase s from 0, the model becomes more flexible and j is constrained and will increase to the least squares estimate.*

**(b) Repeat (a) for test RSS.**
*ii. Deacrease intially, and then eventually start increase in a U shape. As we increase s from 0, at first j is constrained and forced close to 0 due to overfitting and then increases again as coefficients are removed from the model.*

**(c) Repeat (a) for variance.**
*iii. Steadily increases. When s = 0, the model has no variance. However, as we increase s from 0, the model incorporates more s, which increases the variance.*

**(d) Repeat (a) for (squared) bias.** *iv. Steadily decreases. When s = 0, the model is highly biased. However, as we increase s from 0, the s in the model move away from 0 and decrease the bias.*

## Question 2

**In this exercise, we will predict the number of applications received using the other variables in the `College` data set.**

*Let's load the college dataset...*

```r
# Look at what the data structure looks like
# How many rows and columns?
dim(College) # Number of Rows, Columns
```

```
## [1] 777  18
```

```r
# What type of data do we have?
str(College)
```

```
## 'data.frame':    777 obs. of  18 variables:
##  $ Private    : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Apps       : num  1660 2186 1428 417 193 ...
##  $ Accept     : num  1232 1924 1097 349 146 ...
##  $ Enroll     : num  721 512 336 137 55 158 103 489 227 172 ...
##  $ Top10perc  : num  23 16 22 60 16 38 17 37 30 21 ...
##  $ Top25perc  : num  52 29 50 89 44 62 45 68 63 44 ...
##  $ F.Undergrad: num  2885 2683 1036 510 249 ...
##  $ P.Undergrad: num  537 1227 99 63 869 ...
##  $ Outstate   : num  7440 12280 11250 12960 7560 ...
##  $ Room.Board : num  3300 6450 3750 5450 4120 ...
##  $ Books      : num  450 750 400 450 800 500 500 450 300 660 ...
##  $ Personal   : num  2200 1500 1165 875 1500 ...
##  $ PhD        : num  70 29 53 92 76 67 90 89 79 40 ...
##  $ Terminal   : num  78 30 66 97 72 73 93 100 84 41 ...
##  $ S.F.Ratio  : num  18.1 12.2 12.9 7.7 11.9 9.4 11.5 13.7 11.3 11.5 ...
##  $ perc.alumni: num  12 16 30 37 2 11 26 37 23 15 ...
##  $ Expend     : num  7041 10527 8735 19016 10922 ...
##  $ Grad.Rate  : num  60 56 54 59 15 55 63 73 80 52 ...
```

```r
#How many variables do we have in the data?
ncol(College)
```

```
## [1] 18
```

```r
# What variables do we have in the data?
colnames(College)
```

```
##  [1] "Private"     "Apps"        "Accept"      "Enroll"      "Top10perc"  
##  [6] "Top25perc"   "F.Undergrad" "P.Undergrad" "Outstate"    "Room.Board" 
## [11] "Books"       "Personal"    "PhD"         "Terminal"    "S.F.Ratio"  
## [16] "perc.alumni" "Expend"      "Grad.Rate"
```

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

```r
## set the seed to make your partition reproductible
set.seed(123)
# Randomly pick observations from the data for the test data
train <-  sample(1:dim(College)[1], dim(College)[1] / 2)
test <- -train
```

```r
# Create the training and testing data
train_college <- College[train, ]
test_college <- College[test, ] # Remove the training data est data

# How many observations in the training and testing data?
nrow(train_college)
```

```
## [1] 388
```

```r
nrow(test_college)
```

```
## [1] 389
```

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

```r
# Run the linear model
linear_model <- lm(Apps ~ ., data = train_college)
summary(linear_model)
```

```
##
## Call:
## lm(formula = Apps ~ ., data = train_college)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5735.4  -379.7     6.5   327.6  7642.4
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -7.663e+02  5.255e+02  -1.458 0.145623
## PrivateYes  -1.567e+02  1.787e+02  -0.877 0.381124
## Accept       1.742e+00  4.591e-02  37.949  < 2e-16 ***
## Enroll      -1.306e+00  2.260e-01  -5.779 1.60e-08 ***
## Top10perc    5.078e+01  7.638e+00   6.648 1.07e-10 ***
## Top25perc   -1.823e+01  6.291e+00  -2.898 0.003976 **
## F.Undergrad  8.652e-02  3.782e-02   2.287 0.022733 *
## P.Undergrad  5.629e-02  3.993e-02   1.410 0.159482
## Outstate    -9.529e-02  2.479e-02  -3.844 0.000142 ***
## Room.Board   1.579e-01  6.731e-02   2.345 0.019539 *
## Books        3.267e-03  2.869e-01   0.011 0.990921
## Personal     1.654e-01  9.347e-02   1.769 0.077684 .
## PhD         -1.556e+01  5.913e+00  -2.631 0.008869 **
## Terminal     4.628e+00  6.545e+00   0.707 0.479930
## S.F.Ratio    2.417e+01  1.680e+01   1.438 0.151173
## perc.alumni  2.514e+00  5.785e+00   0.435 0.664137
## Expend       7.056e-02  1.709e-02   4.128 4.52e-05 ***
## Grad.Rate    6.034e+00  3.955e+00   1.526 0.127958
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 994.6 on 370 degrees of freedom
## Multiple R-squared:  0.9503, Adjusted R-squared:  0.948
## F-statistic: 416.1 on 17 and 370 DF,  p-value: < 2.2e-16
```

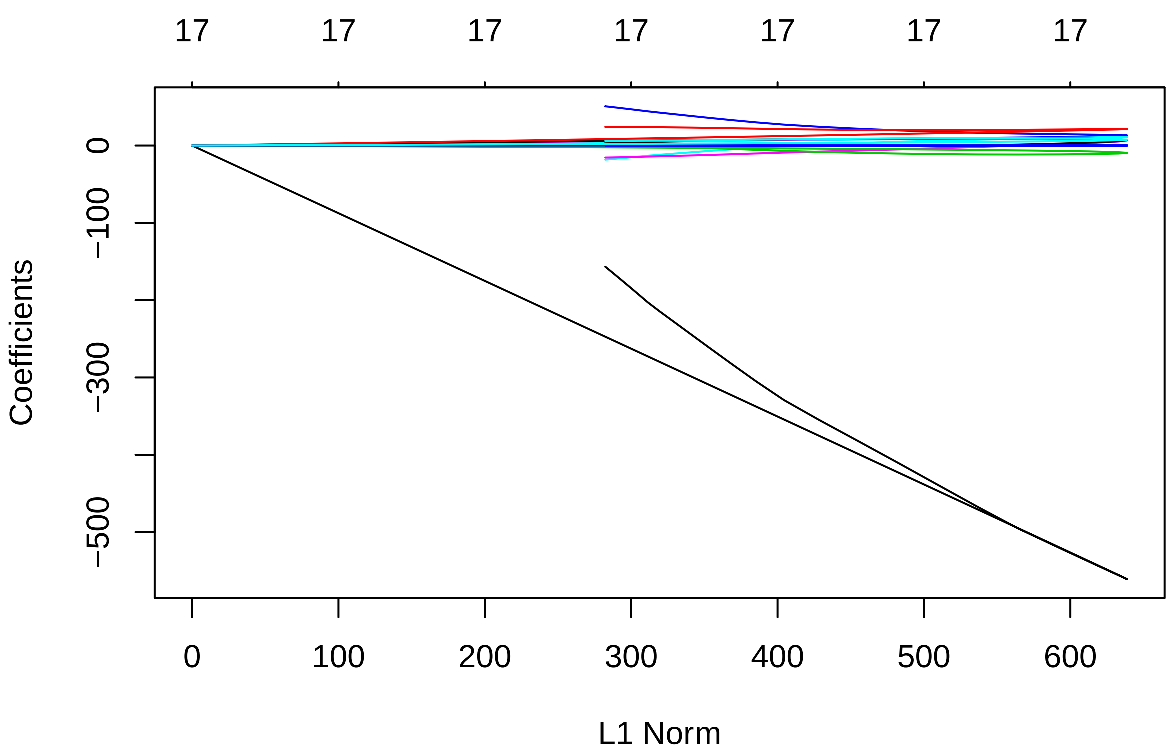
```
linear_prediction <- predict(linear_model, test_college)
linear_MSE <- mean((linear_prediction - test_college$Apps)^2); linear_MSE
```

```
## [1] 1277961
```

*The mean squared error or MSE is* $1.2779606 \times 10^6$*, which is extremely large.*

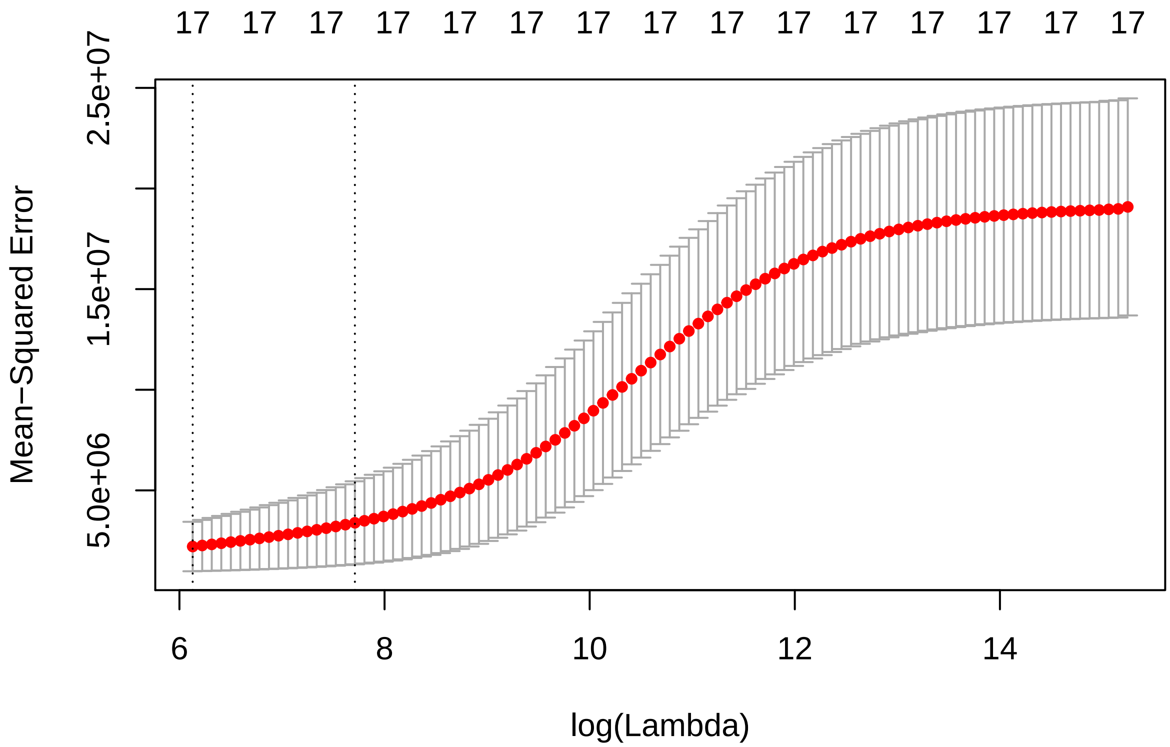
**(c) Fit a ridge regression model on the training set, with   chosen by cross-validation. Report the test error obtained.**

```
# Create a model matrix from the train and test data
train_matrix <- model.matrix(Apps ~ ., data = train_college)
test_matrix <- model.matrix(Apps ~ ., data = test_college)
grid = 10^seq(10, -2, length=100)
# With lambda = grid we will implement a ridge regression over a a grid ov values ranging from 10^10 to
# When alpha = 0 we fit a ridge regression
ridge <- glmnet(train_matrix, train_college$Apps, alpha = 0, lambda = grid, thresh = 1e-12)
plot(ridge)
```




```
# Run a k=fold cross validation for the ridge regression model

cv_ridge  <- cv.glmnet(train_matrix, train_college[, "Apps"], alpha=0)
plot(cv_ridge)
```




```
best_lambda_ridge <- cv_ridge$lambda.min
best_lambda_ridge # The best lambda value!
```

```
## [1] 459.0958
```

*From above, we see that the value of  that results in the smallest cross validation error for ridge regression is* 459.0958225*.*

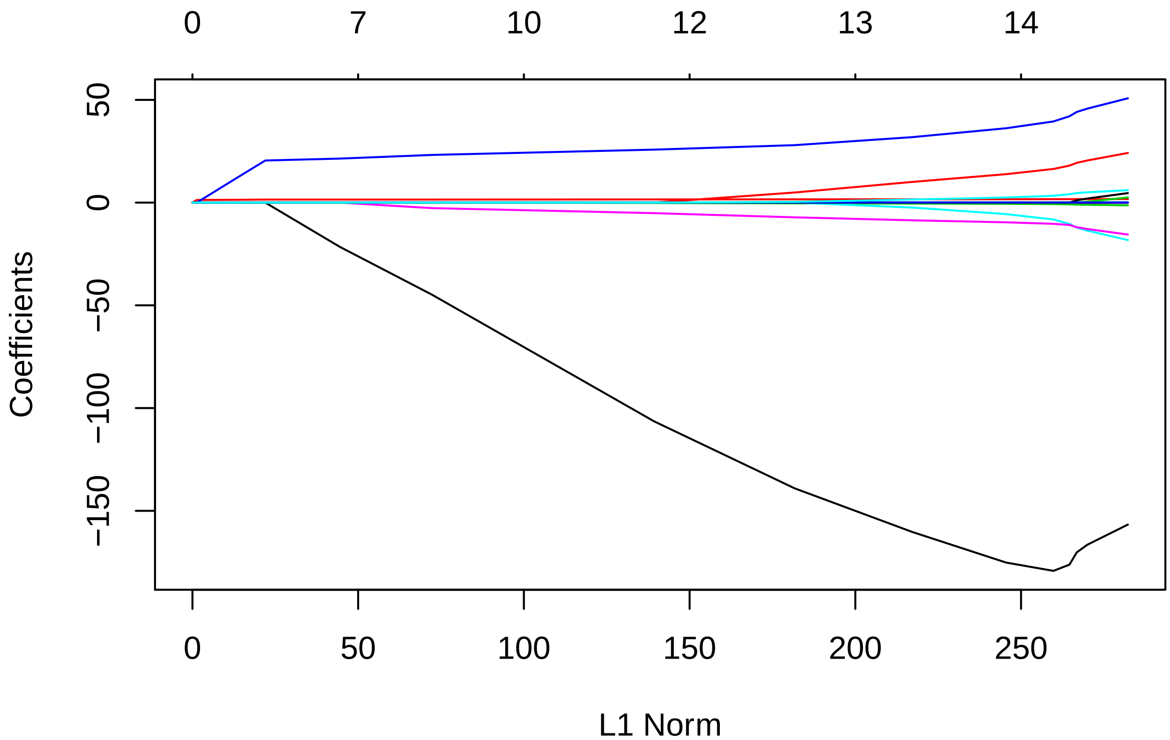
```
# What is the test MSE associated with this best value of  for ridge regression?
ridge_prediction <- predict(ridge, newx = test_matrix, s = best_lambda_ridge)
ridge_MSE <- mean((ridge_prediction - test_college[, "Apps"])^2); ridge_MSE
```

```
## [1] 1131181
```

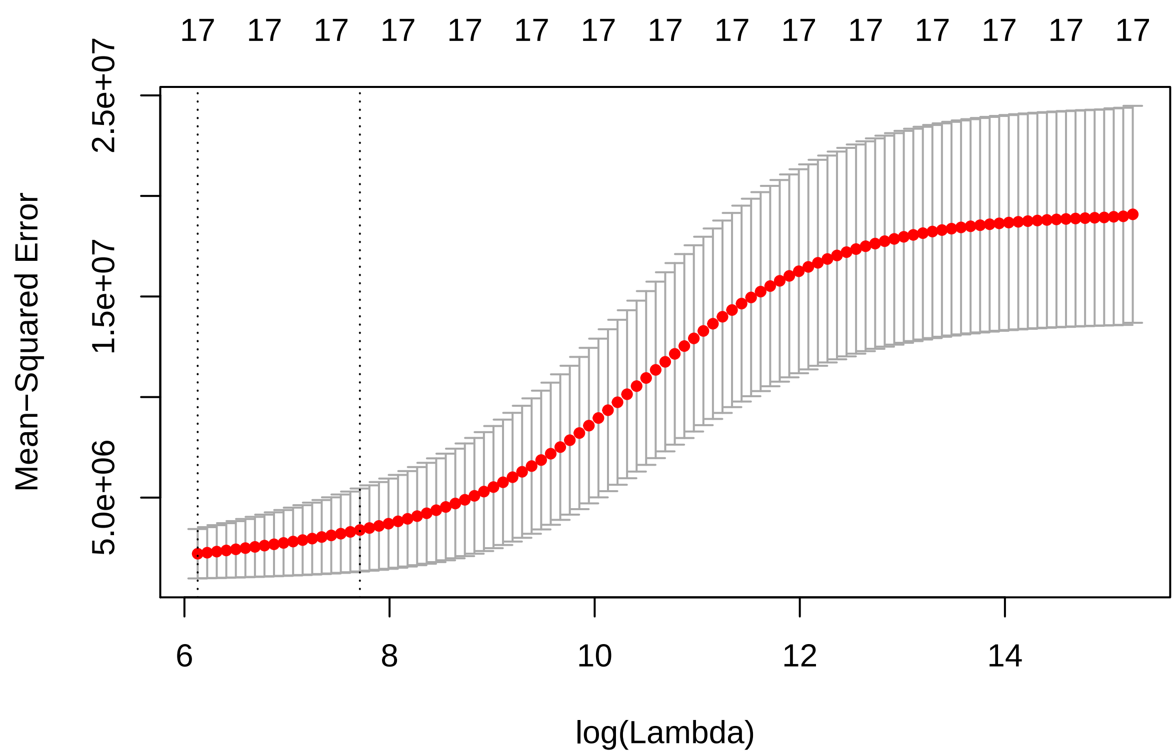
*The test MSE is* $1.1311808 \times 10^6$*.*

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

```
# When alpha = 1 we fit a lasso
lasso <- glmnet(train_matrix, train_college$Apps, alpha = 1, lambda = grid, thresh = 1e-12)
plot(lasso)
```




```r
# Run a k=fold cross validation for the ridge regression model
# When alpha = 0 we fit a ridge regression
cv_lasso  <- cv.glmnet(train_matrix, train_college[, "Apps"], alpha=1)
plot(cv_ridge)
```



```r
best_lambda_lasso <- cv_lasso$lambda.min
best_lambda_lasso # The best lambda value!
```

```
## [1] 2.688934
```

*From above, we see that the value of that results in the smallest cross validation error for lasso is* 2.6889338*.*

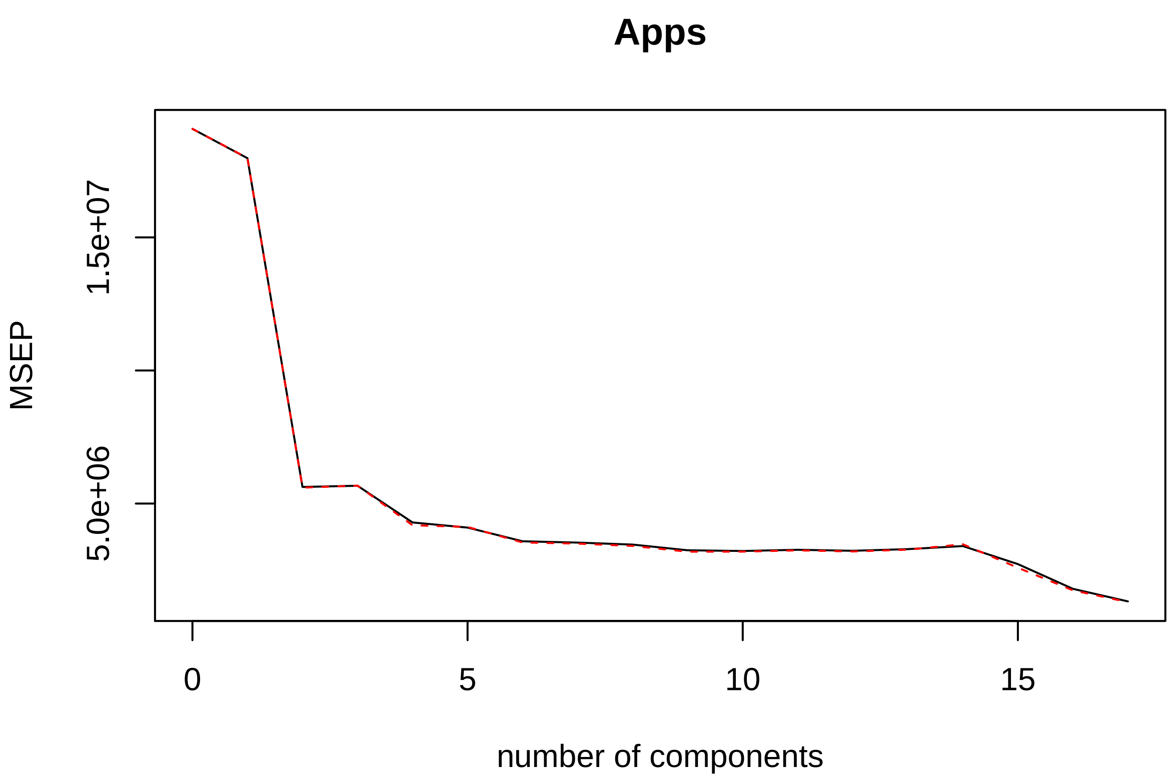
```r
# What is the test MSE associated with this best value of  for lasso?
lasso_prediction <- predict(lasso, newx = test_matrix, s = best_lambda_lasso)
lasso_MSE <- mean((lasso_prediction - test_college[, "Apps"])^2); lasso_MSE
```

```
## [1] 1265727
```

*The test MSE for lasso is* $1.265727 \times 10^6$*.*

**(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.**

```r
# Fit a PCR Model
pcr_fit <- pcr(Apps ~ ., data = train_college, scale = TRUE, validation = "CV")
validationplot(pcr_fit, val.type = "MSEP")
```



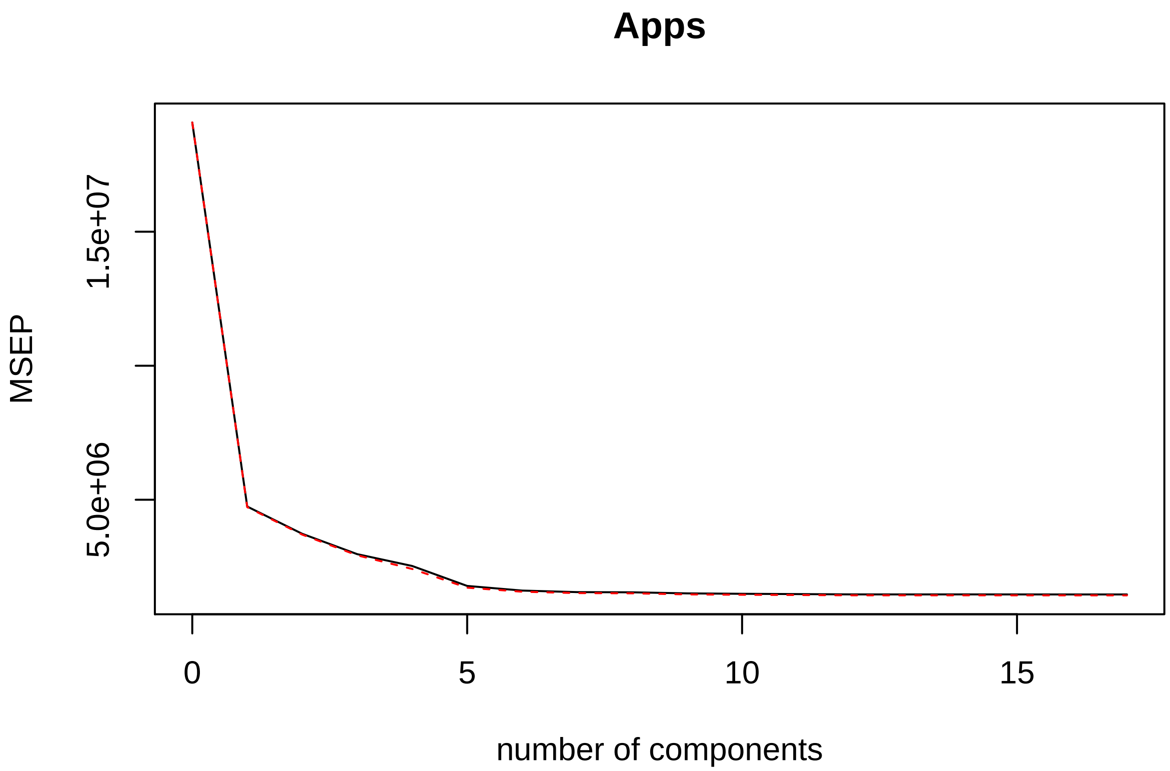

```r
# What is the test MSE associated with this best value of  for PCR?
pcr_prediction <- predict(pcr_fit, test_college, ncomp = 7)
pcr_MSE <- mean((test_college[, "Apps"] - data.frame(pcr_prediction))^2); pcr_MSE
```

```
## [1] 1729401
```

*The test MSE for PCR is* $1.7294013 \times 10^6$.

**(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.**

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




```
pls_prediction <- predict(pls_fit, test_college, ncomp = 7)
pls_MSE <- mean((test_college[, "Apps"] - data.frame(pls_prediction))^2); pls_MSE
```

```
## [1] 1249630
```

*The test MSE for PLS is* $1.2496297 \times 10^6$.

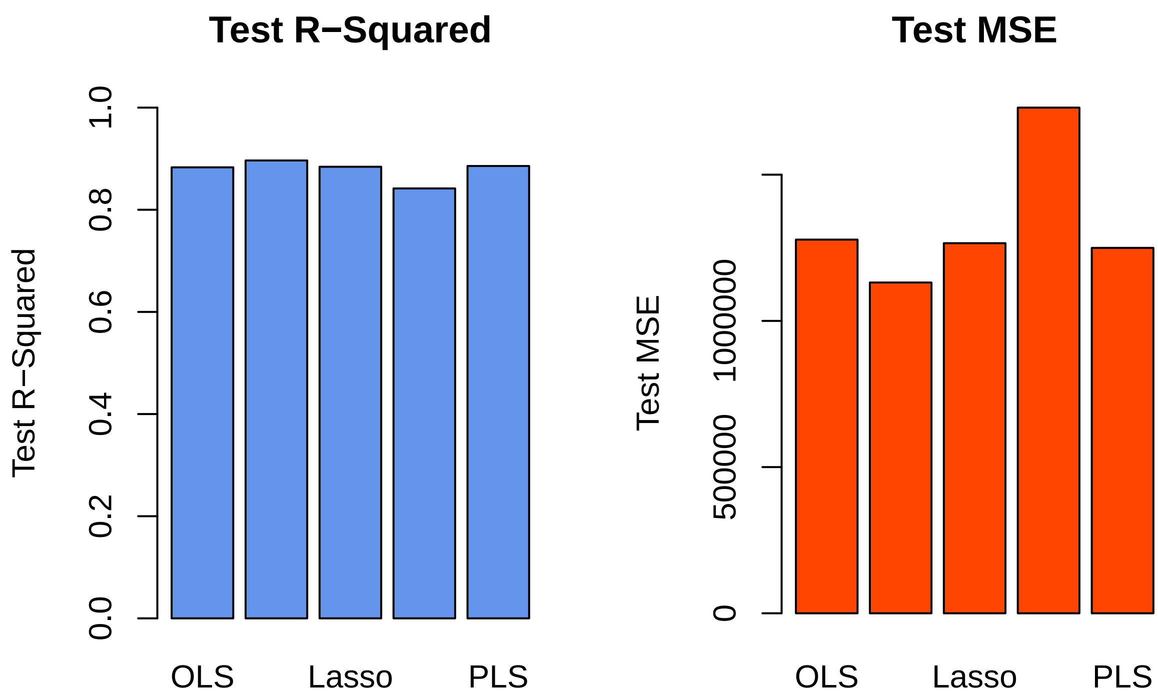
**(g) Comment on the results obtained. How accurately can we predict the number of college applications received? Is there much difference among the test errors resulting from these five approaches?**

```
# To answer this, let's create a plot of the R-squared values and the MSE values
average_test <- mean(test_college[, "Apps"])
linear_r2 = 1 - mean((test_college[, "Apps"] - linear_prediction)^2)/mean((test_college[, "Apps"] - ave
ridge_r2 = 1 - mean((test_college[, "Apps"] -ridge_prediction)^2) /mean((test_college[, "Apps"] - averag
lasso_r2 = 1 - mean((test_college[, "Apps"] -lasso_prediction)^2) /mean((test_college[, "Apps"] - averag
pcr_r2 = 1 - mean((test_college[, "Apps"] -data.frame(pcr_prediction))^2) /mean((test_college[, "Apps"]
pls_r2 = 1 - mean((test_college[, "Apps"] -data.frame(pls_prediction))^2) /mean((test_college[, "Apps"]

par(mfrow = c(1,2))
# Let's create a bar plot of the R2 values to visualize any differences
```

```
barplot(c(linear_r2, ridge_r2, lasso_r2, pcr_r2, pls_r2), col="cornflowerblue",
        names.arg=c("OLS","Ridge", "Lasso", "PCR", "PLS"), main = "Test R-Squared",
        ylab = "Test R-Squared", ylim = c(0,1))

barplot(c(linear_MSE, ridge_MSE, lasso_MSE, pcr_MSE, pls_MSE), col="orangered",
        names.arg=c("OLS","Ridge", "Lasso", "PCR", "PLS"), main = "Test MSE",
        ylab = "Test MSE")
```




*For R-squared the results across the tests are very comparable but PCR describes the smallest amount of variation in the data (with the lowest accuracy). This is also reflected in the test MSE which is the highest for PCR.*