# STATS 101C - Statistical Models and Data Mining - Homework 4

*Darren Tsang, Discussion 4B*

Produced on Sunday, Nov. 15 2020 @ 03:07:08 AM

## Question 1 (Exercise 2 from Section 6.8)

### Part A (lasso relative to least squares)

The following statement is true: Lasso 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.

Lasso regression aims to drive some predictor coefficients to 0, which means the model will be less flexible than a least squares model. Furthermore, driving coefficients to 0 decreases variance and increases bias. The MSE is a function of variance and squared bias, so we can see that as long as the squared bias does not increase more than the decrease in variance, the MSE will be smaller.

### Part B (ridge regression relative to least squares)

The following statement is true: 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 argument can be made as in Part A. Note that in ridge regression, when compared to lasso regression, the bias will be lower and the variance will be higher.

### Part C (non-linear methods relative to least squares)

The following statement is true: 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.

Non-linear models are more flexible because they can include the squaring, cubing, etc. of predictors. The nature of having non-linear models lead to a higher variance, thus it is only an improvement in prediction accuracy when the decrease in bias is greater than the increase in variance.

# Question 2 (Exercise 5 from Section 6.8)

## Part A

$$\text{minimize } f = \text{minimize}\Big[(y_1 - \hat{\beta}_1 x_{11} - \hat{\beta}_2 x_{12})^2 + (y_2 - \hat{\beta}_1 x_{21} - \hat{\beta}_2 x_{22})^2 + \lambda(\hat{\beta}_1^2 + \hat{\beta}_2^2)\Big]$$

## Part B

From the instructions, we can set $x_{11} = x_{12} = -x_{21} = -x_{22} = a$ and $y_1 = -y_2 = b$. We do some substitution:

$$(b - \hat{\beta}_1 a - \hat{\beta}_2 a)^2 + (-b + \hat{\beta}_1 a + \hat{\beta}_2 a)^2 + \lambda(\hat{\beta}_1^2 + \hat{\beta}_2^2)$$

Factoring out a negative:

$$(b - \hat{\beta}_1 a - \hat{\beta}_2 a)^2 + (-(b - \hat{\beta}_1 a - \hat{\beta}_2 a))^2 + \lambda(\hat{\beta}_1^2 + \hat{\beta}_2^2)$$

Combining like terms:

$$2(b - \hat{\beta}_1 a - \hat{\beta}_2 a)^2 + \lambda(\hat{\beta}_1^2 + \hat{\beta}_2^2) \tag{1}$$

Taking partial derivatives of (1) and setting equal to 0:

$$\frac{\partial f}{\partial \hat{\beta}_1} = -4a(b - \hat{\beta}_1 a - \hat{\beta}_2 a) + 2\lambda\hat{\beta}_1 = 0 \longrightarrow 2\lambda\hat{\beta}_1 = 4a(b - \hat{\beta}_1 a - \hat{\beta}_2 a) \tag{2}$$

$$\frac{\partial f}{\partial \hat{\beta}_2} = -4a(b - \hat{\beta}_1 a - \hat{\beta}_2 a) + 2\lambda\hat{\beta}_2 = 0 \longrightarrow 2\lambda\hat{\beta}_2 = 4a(b - \hat{\beta}_1 a - \hat{\beta}_2 a) \tag{3}$$

Since the right hand side of (2) and (3) are equal:

$$2\lambda\hat{\beta}_1 = 2\lambda\hat{\beta}_2 \longrightarrow \hat{\beta}_1 = \hat{\beta}_2,$$

as desired.

## Part C

I will be using the alternate formulation of Lasso regression found on Page 220 of the textbook:

$$\min_{\beta}\left\{(y_1 - \hat{\beta}_1 x_{11} - \hat{\beta}_2 x_{12})^2 + (y_2 - \hat{\beta}_1 x_{21} - \hat{\beta}_2 x_{22})^2\right\} \text{ subject to } |\hat{\beta}_1| + |\hat{\beta}_2| \leq s$$

More specifically:

$$\min_{\beta}\left\{2(b - a(\hat{\beta}_1 + \hat{\beta}_2))^2\right\} \text{ subject to } |\hat{\beta}_1| + |\hat{\beta}_2| \leq s \tag{4}$$

## Part D

Note: If we plot the constraint in (4) with the horizontal axis being $\hat{\beta}_1$ and the vertical axis being $\hat{\beta}_2$, we get a diamond centered at the origin with side lengths of $s$.

The smallest value of the equation in (4) is 0. Thus, we are trying to solve:

$$2(b - a(\hat{\beta}_1 + \hat{\beta}_2))^2 = 0$$

Doing some algebra:

$$\hat{\beta}_1 + \hat{\beta}_2 = \frac{b}{a} \tag{5}$$

We notice that (5) is a linear line that will be parallel to the side of the diamond that is in the top right quadrant or bottom left quadrant. The solutions of the optimization problem are the points that are on the aforementioned sides of the diamond. In other words, any $\hat{\beta}_1$, $\hat{\beta}_2$ that satisfy $\hat{\beta}_1 + \hat{\beta}_2 = s$, where $\hat{\beta}_1$, $\hat{\beta}_1$ are of the same sign (either both positive or both negative), are solutions to the optimization problem. This shows that there are multiple solutions, as desired.

## Question 3 (Exercise 9 from Section 6.8)

```
library(ISLR)
library(caret)
library(glmnet)
library(pls)
```

### Part A

```
data(College)

set.seed(1234)
indices <- createDataPartition(College$Apps, p = .7, list = FALSE)

college_train <- College[indices, ]
college_test <- College[-indices, ]

dim(college_train)
```

```
## [1] 545  18
```

```
dim(college_test)
```

```
## [1] 232  18
```

## Part B

```
glm.mod <- train(Apps ~ ., data = college_train, method = "glm")
glm.mod
```

```
## Generalized Linear Model
##
## 545 samples
##  17 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 545, 545, 545, 545, 545, 545, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   1258.967  0.9069493  718.3832
```

```
glm.predictions <- predict(glm.mod, newdata = college_test)
glm.test.error <- mean((glm.predictions - college_test$Apps)^2)
glm.test.error
```

```
## [1] 970934.6
```

The test error is $9.7093457 \times 10^5$.

## Part C

```
grid <- 10^seq(10, -2, length = 100)

x <- model.matrix(Apps ~ ., college_train)[,-1]
y <- college_train$Apps

ridge.mod <- glmnet(x, y, family = "gaussian", alpha = 0,
                    lambda = grid, standardize = TRUE)

set.seed(1234)
ridge.cv.output <- cv.glmnet(x, y, family = "gaussian", alpha = 0,
                             lambda = grid, standardize = TRUE,
                             nfolds = 10)

ridge.best.lambda.cv <- ridge.cv.output$lambda.min
ridge.best.lambda.cv
```

```
## [1] 0.01
```

```
ridge.predictions <- predict(ridge.mod,
                             s = ridge.best.lambda.cv,
                             newx = model.matrix(Apps ~ ., college_test)[,-1])

ridge.test.error <- mean((ridge.predictions - college_test$Apps)^2)
ridge.test.error
```

```
## [1] 970790.9
```

The test error is $9.7079093 \times 10^5$ using the best lambda of 0.01.

## Part D

```
grid <- 10^seq(10, -2, length = 100)

x <- model.matrix(Apps ~ ., college_train)[, -1]
y <- college_train$Apps

lasso.mod <- glmnet(x, y, family = "gaussian", alpha = 1,
                    lambda = grid, standardize = TRUE)

set.seed(1234)
lasso.cv.output <- cv.glmnet(x, y, family = "gaussian", alpha = 1,
                             lambda = grid, standardize = TRUE,
                             nfolds = 10)

lasso.best.lambda.cv <- lasso.cv.output$lambda.min
lasso.best.lambda.cv
```

```
## [1] 18.73817
```

```
lasso.coef <- predict(lasso.mod, type = "coefficients", s = lasso.best.lambda.cv)
lasso.coef[,1]
```

```
##   (Intercept)    PrivateYes        Accept        Enroll     Top10perc
## -356.78962342 -409.83228125    1.55692978   -0.39572827   48.34388735
##     Top25perc   F.Undergrad   P.Undergrad      Outstate    Room.Board
##  -11.97804796    0.00000000    0.03798147   -0.05887328    0.11688635
##         Books      Personal           PhD      Terminal     S.F.Ratio
##    0.00000000    0.00000000   -5.41367850   -4.85247883    4.40587592
##   perc.alumni        Expend     Grad.Rate
##    0.00000000    0.07422791    3.91065378
```

```
lasso.predictions <- predict(lasso.mod,
                             s = lasso.best.lambda.cv,
                             newx = model.matrix(Apps ~ ., college_test)[,-1])
lasso.test.error <- mean((lasso.predictions - college_test$Apps)^2)
lasso.test.error
```

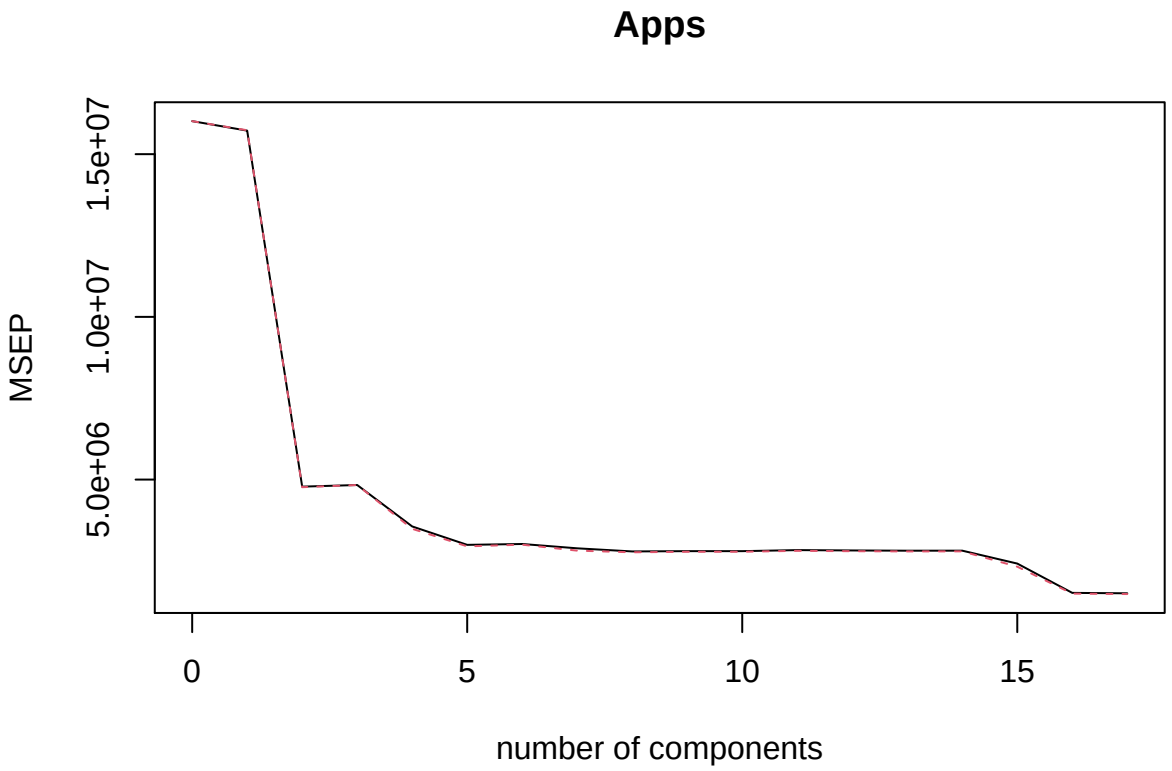
```
## [1] 892542.9
```

The test error is $8.9254288 \times 10^5$ using the best lambda of 18.7381742. There are 13 predictors with non-zero coefficient estimates.

### Part E

```
set.seed(1234)

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




```
pcr.predictions <- predict(pcr.fit,
                           newdata = college_test,
                           ncomp = 16)

pcr.test.error <- mean((pcr.predictions - college_test$Apps)^2)
pcr.test.error
```

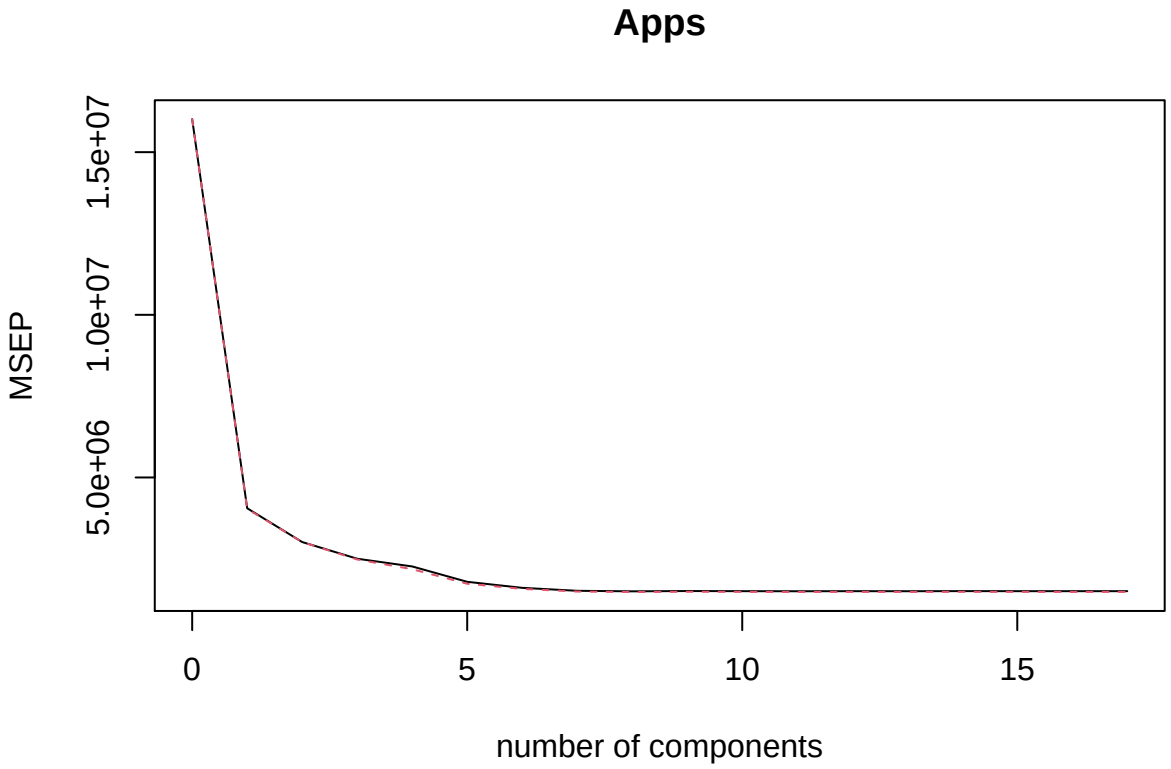
```
## [1] 1058674
```

The test error is $1.0586742 \times 10^6$ using $M = 16$, which was chosen through cross-validation.

### Part F

```
set.seed(1234)

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




```
pls.predictions <- predict(pls.fit,
                           newdata = college_test,
                           ncomp = 7)

pls.test.error <- mean((pls.predictions - college_test$Apps)^2)
pls.test.error
```

```
## [1] 995321
```

The test error is $9.9532104 \times 10^5$ using $M = 7$, which was chosen through cross-validation.

## Part G

The testing errors for the linear model, ridge regression model, lasso model, PCR model, PLS model are $9.7093457 \times 10^5$, $9.7079093 \times 10^5$, $8.9254288 \times 10^5$, $1.0586742 \times 10^6$, and $9.9532104 \times 10^5$ respectively. Note that the lasso model had the lowest, the PCR model had the highest, and the other three are pretty close to one another. Overall, I would not say that there is a huge difference among the test errors. As a result, I would say that there is no model that is clearly better than the others. The difference in the test errors can simply be explained by the randomness of the testing / training set; if we did a different split of the two sets, I would not be surprised if the ordering of the lowest to highest test errors shifted around.

```
test.avg <- mean(college_test$Apps)
test.actual <- college_test$Apps

glm.rsquared <- 1 - sum((test.actual - glm.predictions)^2) / sum((test.actual - test.avg)^2)
ridge.rsquared <- 1 - sum((test.actual - ridge.predictions)^2) / sum((test.actual - test.avg)^2)
lasso.rsquared <- 1 - sum((test.actual - lasso.predictions)^2) / sum((test.actual - test.avg)^2)
pcr.rsquared <- 1 - sum((test.actual - pcr.predictions)^2) / sum((test.actual - test.avg)^2)
pls.rsquared <- 1 - sum((test.actual - pls.predictions)^2) / sum((test.actual - test.avg)^2)


cbind(glm.rsquared, ridge.rsquared, lasso.rsquared, pcr.rsquared, pls.rsquared)
```

```
##      glm.rsquared ridge.rsquared lasso.rsquared pcr.rsquared pls.rsquared
## [1,]    0.9229533      0.9229647      0.9291739    0.9159909    0.9210181
```

Furthermore, the $R^2$ value is calculated and show above for each of the five methods. We can see that all of the $R^2$ values are pretty high (around 0.92), so it is fair to say we can accurately predict the number of college applicants. The lasso $R^2$ is the highest, with a value of 0.9291739, but not by much.