# HW6

*Yigao Li*

*March 24, 2018*

## 5.4 - 8

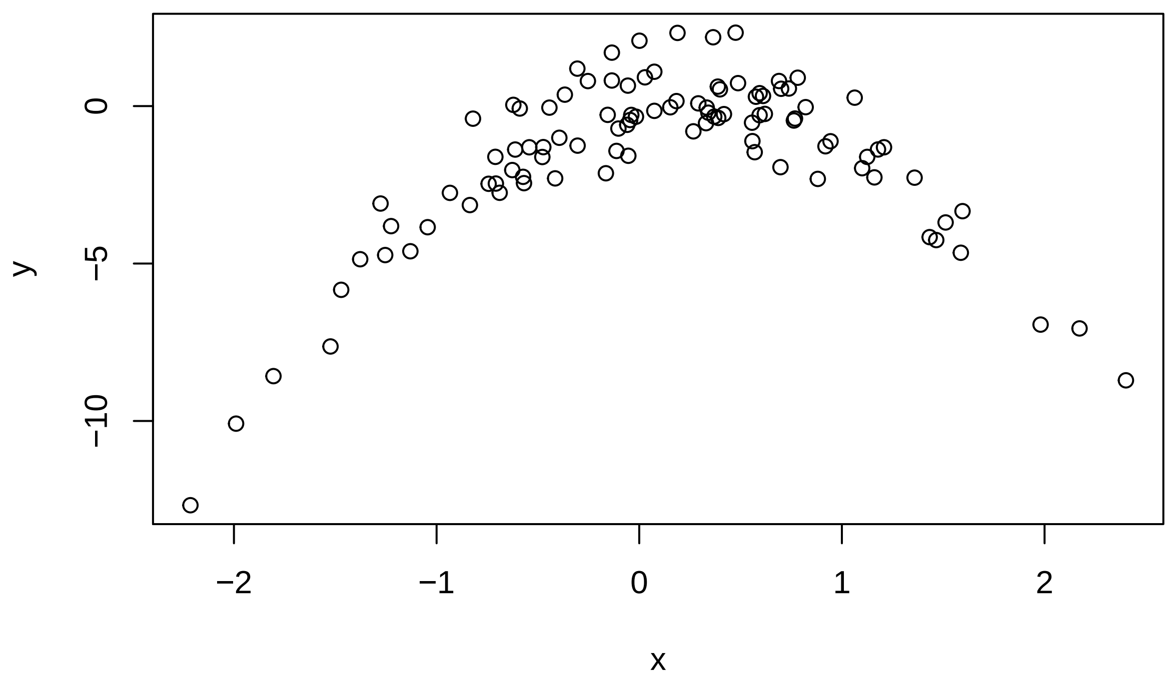
### (a)

```
library(boot)
set.seed(1)
x <- rnorm(100)
y <- x - 2*x^2 + rnorm(100)
```

$n = 100$, $p = 2$. $y = x - 2x^2 + \epsilon$.

### (b)

```
plot(x,y)
```




Scatterplot is a bell shape, quadratic plot.

### (c)

```
simulated.data <- data.frame(x,y)
model.1 <- glm(y ~ x, data = simulated.data)
loocv.1 <- cv.glm(simulated.data, model.1)
loocv.1$delta
```

```
## [1] 7.288162 7.284744
```

```
model.2 <- glm(y ~ poly(x,2), data = simulated.data)
loocv.2 <- cv.glm(simulated.data, model.2)
loocv.2$delta
```

```
## [1] 0.9374236 0.9371789
```

```
model.3 <- glm(y ~ poly(x,3), data = simulated.data)
loocv.3 <- cv.glm(simulated.data, model.3)
loocv.3$delta
```

```
## [1] 0.9566218 0.9562538
```

```
model.4 <- glm(y ~ poly(x,4), data = simulated.data)
loocv.4 <- cv.glm(simulated.data, model.4)
loocv.4$delta
```

```
## [1] 0.9539049 0.9534453
```

### (d)

```
set.seed(6)
x.re <- rnorm(100)
y.re <- x.re - 2*x.re^2 + rnorm(100)
simulated.data.re <- data.frame(x.re,y.re)
model.1.re <- glm(y.re ~ x.re, data = simulated.data.re)
loocv.1.re <- cv.glm(simulated.data.re, model.1.re)
loocv.1.re$delta
```

```
## [1] 7.992172 7.989248
```

```
model.2.re <- glm(y.re ~ poly(x.re,2), data = simulated.data.re)
loocv.2.re <- cv.glm(simulated.data.re, model.2.re)
loocv.2.re$delta
```

```
## [1] 0.8498490 0.8495773
```

```
model.3.re <- glm(y.re ~ poly(x.re,3), data = simulated.data.re)
loocv.3.re <- cv.glm(simulated.data.re, model.3.re)
loocv.3.re$delta
```

```
## [1] 0.8654693 0.8650660
```

```
model.4.re <- glm(y.re ~ poly(x.re,4), data = simulated.data.re)
loocv.4.re <- cv.glm(simulated.data.re, model.4.re)
loocv.4.re$delta
```

```
## [1] 0.8709239 0.8704101
```

The results are the same because LOOCV has low bias.

### (e)

As expected, Model ii has the smallest LOOCV error because our true model is a quadratic function.

### (f)

```r
summary(model.1)
```

```
##
## Call:
## glm(formula = y ~ x, data = simulated.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -9.5161  -0.6800   0.6812   1.5491   3.8183
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.6254     0.2619  -6.205 1.31e-08 ***
## x             0.6925     0.2909   2.380   0.0192 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 6.760719)
##
##     Null deviance: 700.85  on 99  degrees of freedom
## Residual deviance: 662.55  on 98  degrees of freedom
## AIC: 478.88
##
## Number of Fisher Scoring iterations: 2
```

```r
summary(model.2)
```

```
##
## Call:
## glm(formula = y ~ poly(x, 2), data = simulated.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9650  -0.6254  -0.1288   0.5803   2.2700
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.5500     0.0958  -16.18  < 2e-16 ***
## poly(x, 2)1   6.1888     0.9580    6.46 4.18e-09 ***
## poly(x, 2)2 -23.9483     0.9580  -25.00  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.9178258)
##
##     Null deviance: 700.852  on 99  degrees of freedom
## Residual deviance:  89.029  on 97  degrees of freedom
```

```
## AIC: 280.17
##
## Number of Fisher Scoring iterations: 2
```

```r
summary(model.3)
```

```
##
## Call:
## glm(formula = y ~ poly(x, 3), data = simulated.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9765  -0.6302  -0.1227   0.5545   2.2843
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.55002    0.09626 -16.102  < 2e-16 ***
## poly(x, 3)1   6.18883    0.96263   6.429 4.97e-09 ***
## poly(x, 3)2 -23.94830    0.96263 -24.878  < 2e-16 ***
## poly(x, 3)3   0.26411    0.96263   0.274    0.784
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.9266599)
##
##     Null deviance: 700.852  on 99  degrees of freedom
## Residual deviance:  88.959  on 96  degrees of freedom
## AIC: 282.09
##
## Number of Fisher Scoring iterations: 2
```

```r
summary(model.4)
```

```
##
## Call:
## glm(formula = y ~ poly(x, 4), data = simulated.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0550  -0.6212  -0.1567   0.5952   2.2267
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.55002    0.09591 -16.162  < 2e-16 ***
## poly(x, 4)1   6.18883    0.95905   6.453 4.59e-09 ***
## poly(x, 4)2 -23.94830    0.95905 -24.971  < 2e-16 ***
## poly(x, 4)3   0.26411    0.95905   0.275    0.784
## poly(x, 4)4   1.25710    0.95905   1.311    0.193
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.9197797)
##
##     Null deviance: 700.852  on 99  degrees of freedom
## Residual deviance:  87.379  on 95  degrees of freedom
```

```
## AIC: 282.3
##
## Number of Fisher Scoring iterations: 2
```

Except the linear model, the other 3 models have statistically significant coefficients of $x$ and $x^2$, which agrees with conclusions from cross-validation results.

# 6.8 - 1

## (a)

Best subset

## (b)

Best subset

## (c)

i. True
ii. True
iii. False
iv. False
v. False

# 6.8 - 2

## (a)

iii. is correct because LASSO takes penalty into consideration. Model flexibility decreases, and shrinkage of LASSO coefficient leads to decrease in variance and increase in bias.

## (b)

iii. is correct. Same reason with above.

# 6.8 - 9

## (a)

```r
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 3.4.3
```

```r
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.4.3
## Loading required package: Matrix
## Loading required package: foreach
## Warning: package 'foreach' was built under R version 3.4.3
## Loaded glmnet 2.0-13
```

```r
college <- College
Split <- floor(dim(college[1])*0.5)
train <- 1:Split
```

```
## Warning in 1:Split: numerical expression has 2 elements: only the first
## used
```

```r
test <- (Split+1):dim(college)[1]
```

```
## Warning in (Split + 1):dim(college)[1]: numerical expression has 2
## elements: only the first used
```

```r
college.train <- college[train,]
college.test <- college[test,]
```

### (b)

```r
model.5 <- lm(Apps ~ ., data = college.train)
pred.5 <- predict(model.5, college.test)
mean((college.test$Apps - pred.5)^2)
```

```
## [1] 1714500
```

### (c)

```r
xtrain.matrix <- model.matrix(Apps ~ ., data = college.train)
xtest.matrix <- model.matrix(Apps ~ ., data = college.test)
y <- college.train$Apps
ridge.fit <- glmnet(xtrain.matrix, y, alpha = 0)
ridge.cv <- cv.glmnet(xtrain.matrix, y, alpha = 0)
best.lambda.ridge <- ridge.cv$lambda.min
pred.ridge <- predict(ridge.fit, s = best.lambda.ridge, newx = xtest.matrix)
mean((college.test$Apps - pred.ridge)^2)
```

```
## [1] 2712097
```

### (d)

```r
lasso.fit <- glmnet(xtrain.matrix, y, alpha = 1)
lasso.cv <- cv.glmnet(xtrain.matrix, y, alpha = 1)
best.lambda.lasso <- lasso.cv$lambda.min
```

```r
pred.lasso <- predict(lasso.fit, s = best.lambda.lasso, newx = xtest.matrix)
mean((college.test$Apps - pred.lasso)^2)
```

```
## [1] 1725149
```

```r
coef(lasso.cv, s = best.lambda.lasso)
```

```
## 19 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept) -7.873426e+02
## (Intercept)  .
## PrivateYes  -4.902563e+02
## Accept       1.196143e+00
## Enroll       1.055887e-01
## Top10perc    3.845537e+01
## Top25perc   -1.166097e+01
## F.Undergrad  2.939521e-02
## P.Undergrad  4.638486e-03
## Outstate    -2.094698e-02
## Room.Board   1.809135e-01
## Books       -1.693012e-01
## Personal     1.032099e-01
## PhD          .
## Terminal    -9.999517e+00
## S.F.Ratio    1.040948e+01
## perc.alumni -6.467081e+00
## Expend       8.315904e-02
## Grad.Rate    6.938364e+00
```