679 HW4

Xinyi Wang

2/15/2019

Q8

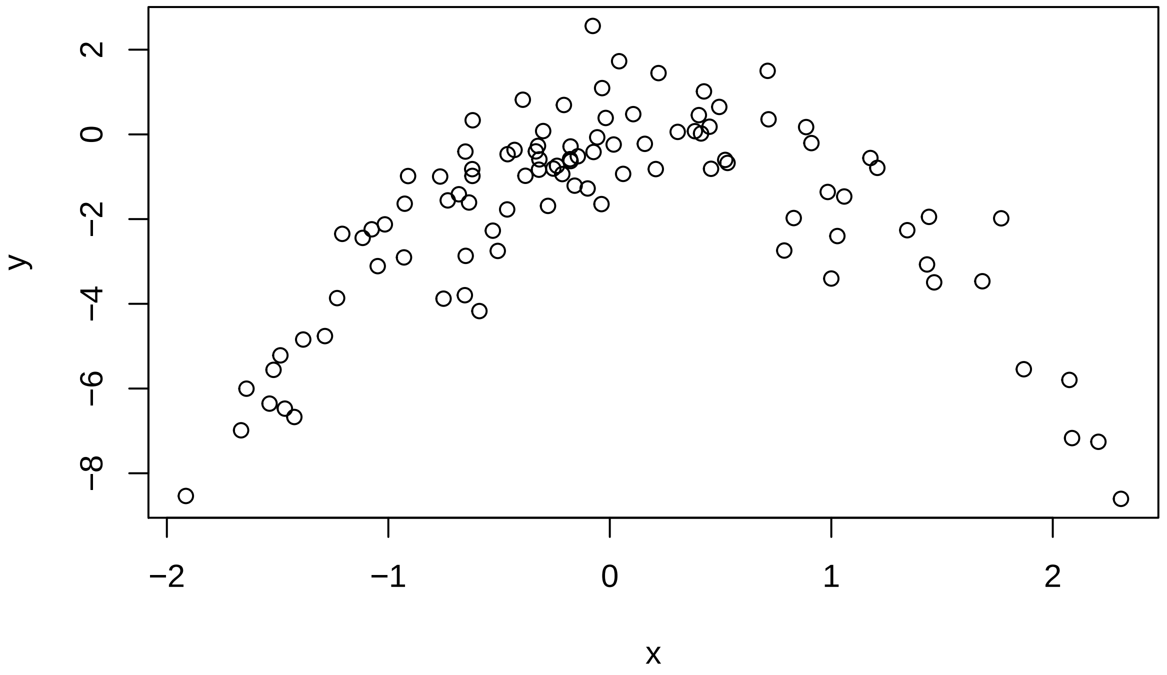
a.Generate a simulated data set as follows :

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

n=100, p=2

b.Create a scatterplot of X against Y. Comment on what you find.

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



####We can find that there may be some relationships between x and y. But the relationship is not linear.

c. Set a random seed, and then compute the LOOCV errors that result from fitting the following four models using least squares :

```
library(boot)
set.seed(1)
Data <- data.frame(x, y)
fit.glm.1 <- glm(y ~ x)
cv.glm(Data, fit.glm.1)$delta[1]
```

```
## [1] 5.890979
```

```
fit.glm.2 <- glm(y ~ poly(x, 2))
cv.glm(Data, fit.glm.2)$delta[1]
```

```
## [1] 1.086596
```

```
fit.glm.3 <- glm(y ~ poly(x, 3))
cv.glm(Data, fit.glm.3)$delta[1]
```

```
## [1] 1.102585
```

```
fit.glm.4 <- glm(y ~ poly(x, 4))
cv.glm(Data, fit.glm.4)$delta[1]
```

```
## [1] 1.114772
```

d.Repeat (c) using another random seed, and report your results. Are your results the same as what you got in (c) ? Why ?

```
set.seed(10)
fit.glm.1 <- glm(y ~ x)
cv.glm(Data, fit.glm.1)$delta[1]
```

```
## [1] 5.890979
```

```
fit.glm.2 <- glm(y ~ poly(x, 2))
cv.glm(Data, fit.glm.2)$delta[1]
```

```
## [1] 1.086596
```

```
fit.glm.3 <- glm(y ~ poly(x, 3))
cv.glm(Data, fit.glm.3)$delta[1]
```

```
## [1] 1.102585
```

```
fit.glm.4 <- glm(y ~ poly(x, 4))
cv.glm(Data, fit.glm.4)$delta[1]
```

```
## [1] 1.114772
```

The results above are identical to the results obtained in (c) since LOOCV evaluates n folds of a single observation.

Which of the models in (c) had the smallest LOOCV error ? Is this what you expected ? Explain your answer.

We may see that the LOOCV estimate for the test MSE is minimum for "fit.glm.2", this is not surprising since we saw clearly in (b) that the relation between "x" and "y" is quadratic.

f comment on the statistical significance of the coefficient estimates that results from fitting each of the models in (c) using least squares. Do these results agree with the conclusions drawn based on the cross-validation results ?

```r
summary(fit.glm.4)
```

```
##
## Call:
## glm(formula = y ~ poly(x, 4))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8914  -0.5244   0.0749   0.5932   2.7796
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.8277     0.1041 -17.549   <2e-16 ***
## poly(x, 4)1   2.3164     1.0415   2.224   0.0285 *
## poly(x, 4)2 -21.0586     1.0415 -20.220   <2e-16 ***
## poly(x, 4)3  -0.3048     1.0415  -0.293   0.7704
## poly(x, 4)4  -0.4926     1.0415  -0.473   0.6373
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 1.084654)
##
##     Null deviance: 552.21  on 99  degrees of freedom
## Residual deviance: 103.04  on 95  degrees of freedom
## AIC: 298.78
##
## Number of Fisher Scoring iterations: 2
```

The p-values show that the linear and quadratic terms are statistically significants and that the cubic and 4th degree terms are not statistically significants. This agree strongly with our cross-validation results which were minimum for the quadratic model.

Q6

For parts (a) through (c), indicate which of i. through iv. is correct. Justify your answer.

The lasso, relative to least squares, is : The lasso is less flexible and will give improved prediction accuracy when its increase in bias is less than its decrease in variance.

Repeat (a) for ridge regression relative to least squares. Same as lasso, ridge regression is less flexible and will give improved prediction accuracy when its increase in bias is less than its decrease in variance.

Repeat (a) for non-linear methods relative to least squares. Non-linear methods are more flexible and will give improved prediction accuracy when their increase in variance are less than their decrease in bias.

Q10

a

```r
set.seed(1)
x <- matrix(rnorm(1000 * 20), 1000, 20)
b <- rnorm(20)
b[3] <- 0
b[4] <- 0
```

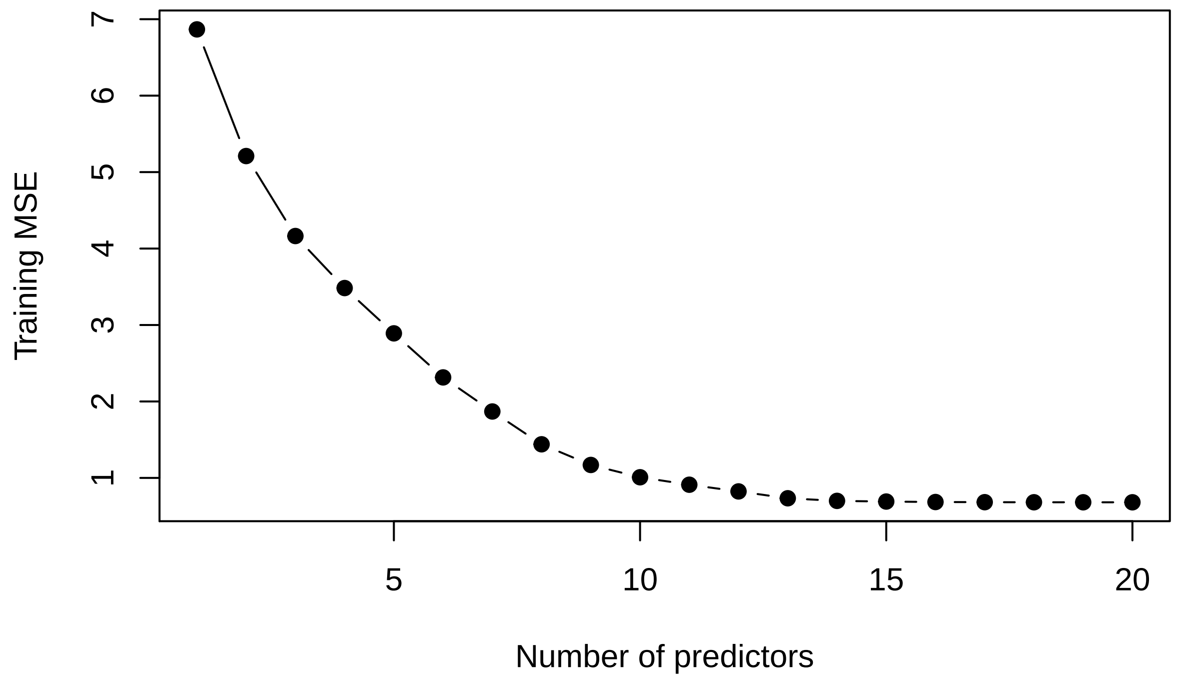
```r
b[9] <- 0
b[19] <- 0
b[10] <- 0
eps <- rnorm(1000)
y <- x %*% b + eps
```

b

```r
train <- sample(seq(1000), 100, replace = FALSE)
test <- -train
x.train <- x[train, ]
x.test <- x[test, ]
y.train <- y[train]
y.test <- y[test]
```

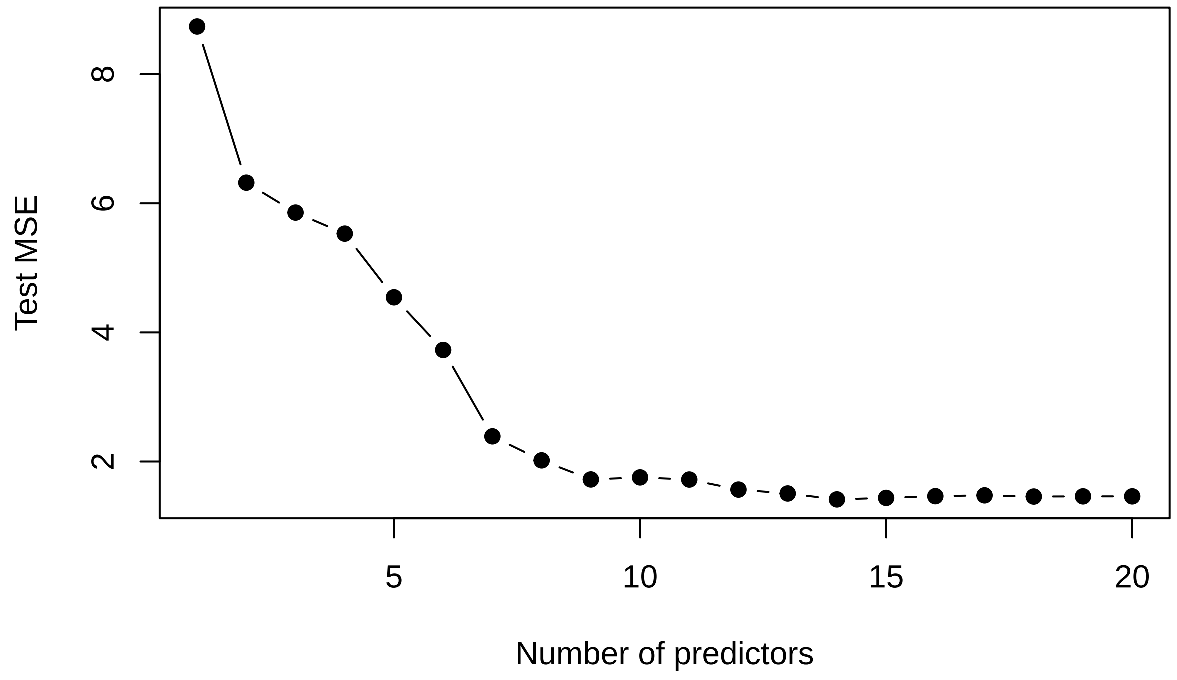
c

```r
library(leaps)
data.train <- data.frame(y = y.train, x = x.train)
regfit.full <- regsubsets(y ~ ., data = data.train, nvmax = 20)
train.mat <- model.matrix(y ~ ., data = data.train, nvmax = 20)
val.errors <- rep(NA, 20)
for (i in 1:20) {
    coefi <- coef(regfit.full, id = i)
    pred <- train.mat[, names(coefi)] %*% coefi
    val.errors[i] <- mean((pred - y.train)^2)
}
plot(val.errors, xlab = "Number of predictors", ylab = "Training MSE", pch = 19, type = "b")
```



d

```r
data.test <- data.frame(y = y.test, x = x.test)
test.mat <- model.matrix(y ~ ., data = data.test, nvmax = 20)
val.errors <- rep(NA, 20)
for (i in 1:20) {
    coefi <- coef(regfit.full, id = i)
    pred <- test.mat[, names(coefi)] %*% coefi
    val.errors[i] <- mean((pred - y.test)^2)
}
plot(val.errors, xlab = "Number of predictors", ylab = "Test MSE", pch = 19, type = "b")
```



e

```r
which.min(val.errors)
```

```
## [1] 14
```

The 14-variables model has the smallest test MSE.

f

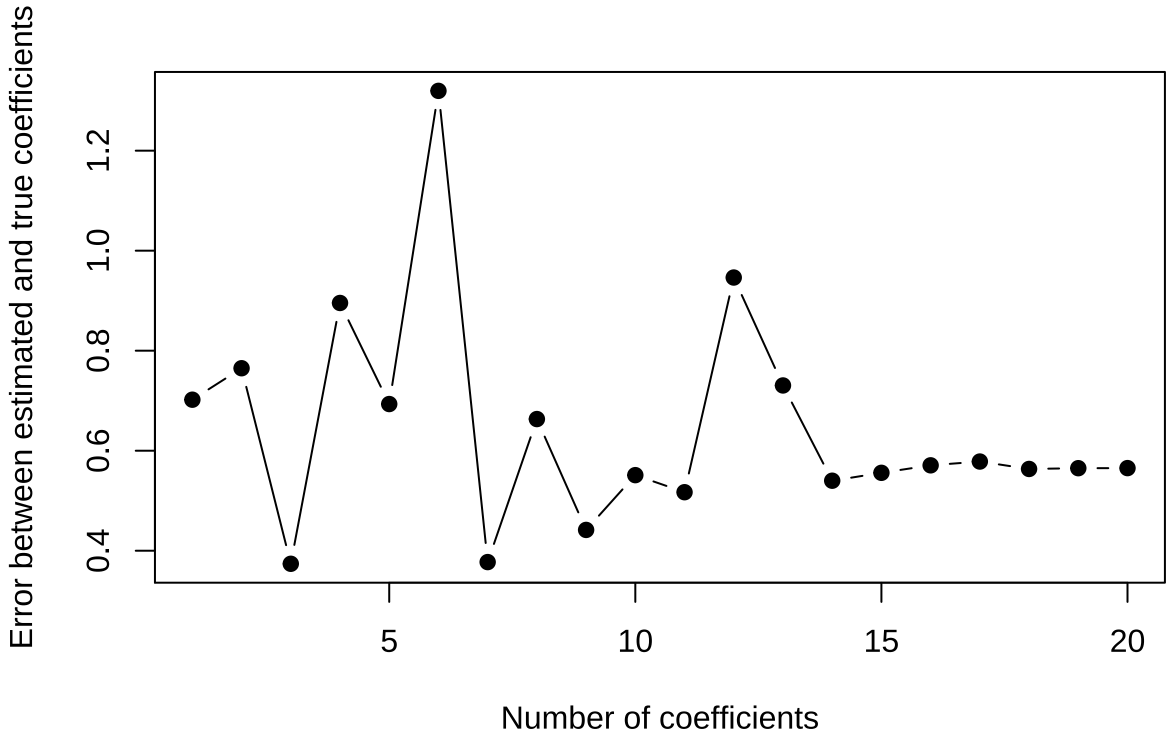
```r
coef(regfit.full, which.min(val.errors))
```

```
## (Intercept)         x.1         x.2         x.5         x.7         x.8
##   0.1630514   0.3707851   0.3176740   1.0424379  -1.2895997   0.8308835
##        x.11        x.12        x.13        x.14        x.15        x.16
##   0.6919104   0.5638802  -0.3641320  -0.8346409  -0.5667810  -0.1959694
##        x.17        x.18        x.20
##   0.3128194   1.5567459  -0.7831598
```

The best model caught all zeroed out coefficients.

g

```r
val.errors <- rep(NA, 20)
x_cols = colnames(x, do.NULL = FALSE, prefix = "x.")
for (i in 1:20) {
    coefi <- coef(regfit.full, id = i)
    val.errors[i] <- sqrt(sum((b[x_cols %in% names(coefi)] - coefi[names(coefi) %in% x_cols])^2) + sum(
}
plot(val.errors, xlab = "Number of coefficients", ylab = "Error between estimated and true coefficients
```


Error between estimated and true coefficients
Number of coefficients