Assignment3

1.Recitation Problem:

1.1 Chapter 5

Que 2

a) Given: total number of observations = n
Since bootstrap allows sampling with replacement
- ⇨ Every observation in original sample is independent and has equal probability to appear in each bootstrap observation
- ⇨ Probability that first bootstrap observation is the j^{th} observation from original sample is = $\frac{1}{n}$
- ⇨ Probability that first bootstrap observation is not the j^{th} observation from original sample is = $1 - \frac{1}{n}$

b) Same as above-
Given: total number of observations = n
Since bootstrap allows sampling with replacement
- ⇨ Every observation in original sample has equal probability to appear in each bootstrap observation
- ⇨ Probability that Second bootstrap observation is the j^{th} observation from original sample is = $\frac{1}{n}$
- ⇨ Probability that Second bootstrap observation is not the j^{th} observation from original sample is = $1 - \frac{1}{n}$

c) As discussed above probabilities are independent to each other
- ⇨ Probability that j^{th} observation is not in the bootstrap sample is given by-
P(j^{th} observation is not in the bootstrap sample) = p(j^{th} observation is not in the first bootstrap sample) * p(j^{th} observation is not in the first bootstrap sample) * … p(j^{th} observation is not in the n^{th} bootstrap sample)

Also, we know that
p(j^{th} observation is not in the first bootstrap sample) = $1 - \frac{1}{n}$
p(j^{th} observation is not in the second bootstrap sample) = $1 - \frac{1}{n}$
:
p(j^{th} observation is not in the nth bootstrap sample) = $1 - \frac{1}{n}$

- ⇨ P(j^{th} observation is not in the bootstrap sample) = $(1 - \frac{1}{n}) * (1 - \frac{1}{n}) * \ldots$ n times
$= (1 - \frac{1}{n})^n$

d) By using above formula:
P(j^{th} observation is not in the bootstrap sample) = $(1 - \frac{1}{n})^n$
- ⇨ P(j^{th} observation is in the bootstrap sample) = $1 - (1 - \frac{1}{n})^n$
Given here n = 5
- ⇨ P(j^{th} observation is in the bootstrap sample) = $1 - (1 - \frac{1}{5})^5$
= 0.67

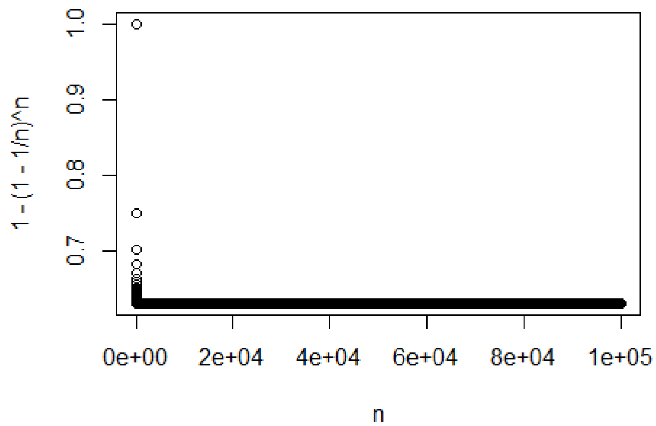
e) Using above formula
P(j^{th} observation is in the bootstrap sample) = $1 - (1 - \frac{1}{n})^n$
Given here n = 100
- ⇨ P(j^{th} observation is in the bootstrap sample) = $1 - (1 - \frac{1}{100})^{100}$
= 0.634

f) Using above formula
P(j^{th} observation is in the bootstrap sample) = $1 - (1 - \frac{1}{n})^n$
Given here n = 1000

Assignment3

⇨ P(j[th] observation is in the bootstrap sample) = 1 - $(1 - \frac{1}{1000})^{1000}$
= 0.632

g)
```
n <-seq(1,100000)
plot(n,1-(1-1/n)^n)
```



h)

```
data <- rep(NA, 10000)
for (i in 1:10000)
{
  data[i] <- sum(sample(1:100, rep = TRUE) == 4) > 0
}
mean(data)
```

```
## [1] 0.632
```

The resulting fraction of 10,000 bootstrap samples that have the 4th observation is close to our predicted probability of 1-(1-1/100)^100 = 63.4%

Que 3

a) Suppose we have 'n' number of observations, then we can perform K-fold cross validation by splitting the data into k equal groups with each of length $\frac{k}{n}$ (approximately). Now, the first group is treated as a validation set and the method is fit on the remaining (k – 1) groups and the mean squared error, MSE_1 is computed. This procedure is repeated k times; each time a different group is considered for validation set. In this way each group will be considered as a validation set only once and considered as a training set for (k-1) times. This process results in k estimates of test error which can be computed by averaging the values and is given by:

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} MSE_i$$

b) i) Validation set Approach:

Advantages:

- This approach is conceptually simple.
- This approach is easy to implement.

Disadvantages:

- Validation estimate of test error rate can be highly variable, depends on which observations are included in training set and which observations are included in validation set.
- In this approach, only observations which are included in training set are used to fit the model and model can perform worse if fewer observations are used for training. Which may also result in overestimation of the test error rate.

ii) **LOOCV:**

Advantages:

- LOOCV cross-validation approach is a special case of k-fold cross-validation in which k=n.
- In this approach each observation will be considered in a validation set only for once and considered in a training set for (n-1) times.
- This approach gives approximately unbiased estimates of the test error since each training set contains (n -1) observations.

Disadvantages:

- It requires fitting the potentially computationally expensive model n times as compared to k-fold cross-validation which requires the model fitting only k times, where k < n.
- this approach has higher variance than k-fold cross-validation (since we are averaging the outputs of n fitted models trained on an almost identical set of observations, these outputs are highly correlated, and the mean of highly correlated quantities has higher variance than less correlated ones).
- So, there is a bias-variance trade-off associated with the choice of kk in k-fold cross-validation; typically using k=5 or k=10 yield test error rate estimates that suffer neither from excessively high bias nor from very high variance.

1.2 Chapter 6

Que 1

a) Best subset selection model will have the least training RSS because in this model the final model will be selected after considering all the possible models with k parameters. But this is not true in case of forward stepwise selection or backward stepwise selection.
b) Best subset selection model will have the high chances of choosing a model with less test RSS as it contains 2^p models whereas as the other two models will consider only $1+\frac{p(p+1)}{2}$ models.
c) i) **TRUE** because in the forward stepwise selection, (k+1)-variable model contains one additional predictor along with all the predictors selected for the k-variable model.
ii) **TRUE** because in the backward stepwise selection, k-variable model is obtained by removing one predictor from (k+1)-variable model which will reduce the RSS of the model.
iii) **FALSE** because both the models follow different criteria. Also, there is not link between the models obtained from forward and backward model.
iv) **FALSE** because both the models follow different criteria. Also, there is not link between the models obtained from forward and backward model.
v) **False** because in the best subset method, the model with (k+1) predictors is obtained by selecting among all possible models with (k+1) predictors. So, it does not guarantee to choose the same predictors for k predictor model.

Que 2

a) Option iii is correct
Because the main aim of regularization is to reduce the test MSE by adding a penalty. It does this by decreasing variance and increasing bias. This penalty shrinks the coefficient and slop gets less steep.
b) Option iii is correct
The reason is same as above because even the ridge regression is the part of regularization techniques. Although

c) Option ii is correct
Because non-linear methods are relatively more flexible than least square that may result in more predication accuracy.

Que 3

a) iv. Steadily decrease
Because as we are increasing **s** from 0, we are reducing the restriction on coefficients, due to which the coefficients will increase to their least square estimates. Hence the model will become more and more flexible which leads a steady decrease in the training RSS.

b) ii. Decrease initially, and then eventually start increasing in a U shape.
Because as we are increasing **s** from 0, we are reducing the restriction on coefficients, due to which the coefficients will increase to their least square estimates. Hence the model will become more and more flexible which results in less test RSS at first which then start increase again to give the U shape curve.

c) iii. Steadily increase.
Because as we are increasing **s** from 0, we are restricting coefficients less and less, due to which the model becomes more and more flexible and as the flexibility of model increases the variance of model will also increases.

d) iv. Steadily decrease
Because as we are increasing **s** from 0, we are restricting coefficients less and less, due to which the model becomes more and more flexible and as the flexibility of model increases the bias of model will decreases.

a) v. Remain Constant
Because irreducible errors are independent of the model. Hence it does not depend on value of **s**.

Que 4

a) iii. Steadily increase
Because as we are increasing λ from 0(i.e. increasing the penalty), we are increasing the restriction on coefficients(i.e. shrinking the coefficient), due to which the coefficients will deviate from their least square estimates. Hence the model will become less and less flexible which results in steady increase in the training RSS.

b) ii. Decrease initially, and then eventually start increasing in a U shape
Because as we are increasing λ from 0(i.e., increasing the penalty), we are increasing the restriction on coefficients(i.e. shrinking the coefficient), due to which the coefficients will deviate from their least square estimates. Hence the model will become less and less flexible which results in less test RSS at first which then start increase again to give the U shape curve.

c) iv. Steadily decrease
Because as we are increasing the penalty, we are shrinking our coefficients and the model becomes less and less flexible and as the flexibility of model decreases the variance of model will also decreases.

d) iii. Steadily increase.
Because as we are increasing the penalty, we are shrinking our coefficients and the model becomes less and less flexible and as the flexibility of model decreases the bias of model will increases.

e) v. Remain Constant
Because irreducible error is independent of the model. Hence it does not depend on value of λ.

Que 5

Assignment3

(a) We know that the general form of ridge regression optimization is given by:-

$$\text{Minimize}: \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_j)^2 + \lambda \sum_{j=1}^{p} \hat{\beta}_j^2$$

here $n = p = 2$ and $\hat{\beta}_0 = 0$

$$\Rightarrow \min [(y_1 - \hat{\beta}_1 x_{11} - \hat{\beta}_2 x_{12})^2 + (y_2 - \hat{\beta}_1 x_{21} - \hat{\beta}_{21} x_{22})^2] + \lambda(\beta_1^2 + \beta_2^2) \quad \text{— (1)}$$

(b) expanding equation (1)

$$\Rightarrow (y_1^2 + \hat{\beta}_1^2 x_{11}^2 + \hat{\beta}_2^2 x_{12}^2 - 2\hat{\beta}_1 x_{11} y_1 - 2\hat{\beta}_2 x_{12} y_1 + 2\hat{\beta}_1\hat{\beta}_2 x_{11} x_{12})$$
$$+ (y_2^2 + \hat{\beta}_1^2 x_{21}^2 + \hat{\beta}_2^2 x_{22}^2 - 2\hat{\beta}_1 x_{21} y_2 - 2\hat{\beta}_2 x_{22} y_2 + 2\hat{\beta}_1\hat{\beta}_2 x_{21} x_{22}$$
$$+ \lambda\beta_1^2 + \lambda\beta_2^2$$

to minimize it we will take its derivative and equate it to zero

$$\frac{\partial}{\partial \hat{\beta}_1} : (2\hat{\beta}_1 x_{11}^2 - 2x_{11}y_1 + 2\hat{\beta}_2 x_{11} x_{12}) + (2\hat{\beta}_1 x_{21}^2 - 2x_{21}y_2 + 2\hat{\beta}_2 x_{21} x_{22}) + 2\lambda\hat{\beta}_1 = 0$$

given

$x_{11} = x_{12} = x_1$ and $x_{21} = x_{22} = x_2$ and divide throughout by 2, we get

$$\Rightarrow (\hat{\beta}_1 x_1^2 - x_1 y_1 + \hat{\beta}_2 x_1^2) + (\hat{\beta}_1 x_2^2 - x_2 y_2 + \hat{\beta}_2 x_2^2) + \lambda\beta_1 = 0$$

$$\Rightarrow \hat{\beta}_1 (x_1^2 + x_2^2) + \beta_2 (x_1^2 + x_2^2) + \lambda\hat{\beta}_1 = x_1 y_1 + x_2 y_2$$

Assignment3

Add $2\hat{\beta}_1 x_1 x_2$ and $2\beta_2 x_1 x_2$ on both sides.

$$= \hat{\beta}_1 (x_1 + x_2)^2 + \beta_2 (x_1 + x_2)^2 + \lambda \hat{\beta}_1 = x_1 y_1 + x_2 y_2$$

As $x_1 + x_2 = 0$ ..

$$\therefore \lambda \hat{\beta}_1 = x_1 y_1 + x_2 y_2 + 2\hat{\beta}_1 x_1 x_2 + 2\hat{\beta}_2 x_1 x_2 \quad \text{— (2)}$$

Similarly taking Partial derivative with resp. to $\hat{\beta}_2$, we get

$$\lambda \hat{\beta}_2 = x_1 y_1 + x_2 y_2 + 2\hat{\beta}_1 x_1 x_2 + 2\hat{\beta}_2 x_1 x_2 \quad \text{— (3)}$$

from equation (2) and (3)

$$\Rightarrow \beta_1 = \beta_2$$

hence proved.

c. $\min \left[(y_1 - \hat{\beta}_1 x_{11} - \hat{\beta}_2 x_{12})^2 + (y_2 - \beta_1 x_{21} - \beta_2 x_{22})^2 \right] + \lambda (|\hat{\beta}_1| + |\hat{\beta}_2|)$

d. Replacing the penalty term from ridge Regression, the derivative term to β is

$$= \frac{\partial}{\partial \beta} (\lambda |\beta|) = \lambda \frac{|\beta|}{\beta}$$

Same like in ridge regression, we get

$$\frac{\lambda |\beta_1|}{\beta_1} = \frac{\lambda |\beta_2|}{\beta_2}$$

provided both β_1 and β_2 are both positive or both negative

Assignment3

==

2. Practicum problem:

2.1 Problem 1

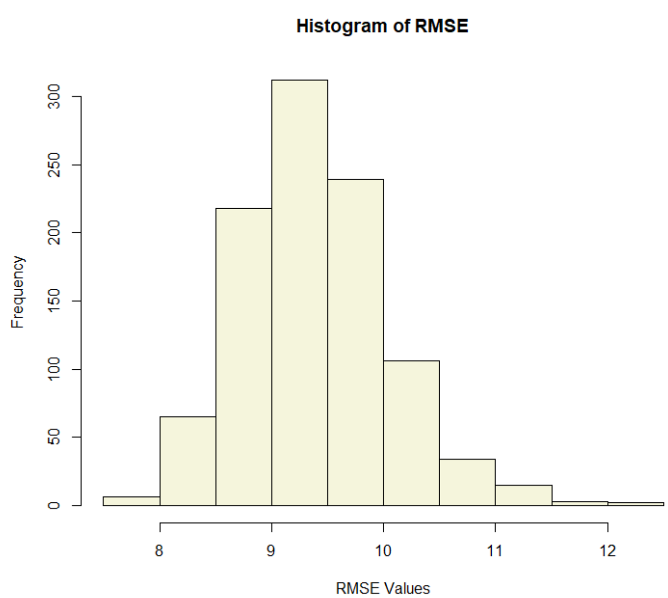
```
library(ggplot2)
library(caret)

## Loading required package: lattice

library(readr)
library(data.table)
#Importing the dataset
URL ="http://archive.ics.uci.edu/ml/machine-learning-
databases/00243/yacht_hydrodynamics.data"
data = fread(URL, header = FALSE)
colnames(data)  <- c("Longitudinal_position","Prismatic_coefficient","Length-
displacement","Beam_drought","Length_beam","Froude_no","Residuary")

#display the dataset
head(data)

##    Longitudinal_position Prismatic_coefficient Length-displacement
## 1:                  -2.3                 0.568                4.78
## 2:                  -2.3                 0.568                4.78
## 3:                  -2.3                 0.568                4.78
## 4:                  -2.3                 0.568                4.78
## 5:                  -2.3                 0.568                4.78
## 6:                  -2.3                 0.568                4.78
##    Beam_drought Length_beam Froude_no Residuary
## 1:         3.99        3.17     0.125      0.11
## 2:         3.99        3.17     0.150      0.27
## 3:         3.99        3.17     0.175      0.47
## 4:         3.99        3.17     0.200      0.78
## 5:         3.99        3.17     0.225      1.18
## 6:         3.99        3.17     0.250      1.82

#dimensions of dataset
nrow(data)

## [1] 308

ncol(data)

## [1] 7

#splitting data into train and test data
partition  <- createDataPartition(data$Longitudinal_position,p=0.8,list=F)
train <- data[partition,]
test  <- data[-partition,]

#Fitting a linear model
```

```
fitted_model  <- lm(Residuary~.,data= train)

#Summary of linear model
summary(fitted_model)

##
## Call:
## lm(formula = Residuary ~ ., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.824  -7.301  -1.895   5.677  30.791
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -4.16490   30.36594  -0.137    0.891
## Longitudinal_position    -0.05712    0.38685  -0.148    0.883
## Prismatic_coefficient    -8.72827   49.16527  -0.178    0.859
## `Length-displacement`     2.87460   15.78962   0.182    0.856
## Beam_drought             -2.00813    6.19047  -0.324    0.746
## Length_beam              -6.44813   15.80220  -0.408    0.684
## Froude_no               119.48358    5.68581  21.014   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.087 on 240 degrees of freedom
## Multiple R-squared:  0.6523, Adjusted R-squared:  0.6436
## F-statistic: 75.05 on 6 and 240 DF,  p-value: < 2.2e-16

#Calculating Training MSE
mean((train$Residuary-predict(fitted_model,train))^2,na.rm=T)

## [1] 80.23318

#Calculating Training RMSE
sqrt(mean((train$Residuary-predict(fitted_model,train))^2,na.rm=T))

## [1] 8.957298

#Finding R2 value
rss  <-  sum((train$Residuary-predict(fitted_model,train))^2,na.rm=T)
tss  <- sum((train$Residuary-mean(train$Residuary,na.rm=T))^2,na.rm=T)
r  <- 1- (rss/tss)
r*100

## [1] 65.23254

#using trainControl method to perform bootstrap
bootstrap  <- trainControl(method="boot",number = 1000)
new_fitted_model <-
train(Residuary~.,data=train,method="lm",trControl=bootstrap,na.action = na.pass)

#checking the fit
new_fitted_model
```

Assignment3

```
## Linear Regression
##
## 247 samples
##   6 predictor
##
## No pre-processing
## Resampling: Bootstrapped (1000 reps)
## Summary of sample sizes: 247, 247, 247, 247, 247, 247, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   9.388165  0.6322668  7.49468
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
#Histogram
hist(new_fitted_model$resample$RMSE,main="Histogram of RMSE",xlab="RMSE
Values",col=blues9)
```




```
#Calculating Training MSE
(mean(new_fitted_model[["resample"]][["RMSE"]]))^2

## [1] 88.13764

#Calculating Training RMSE
mean(new_fitted_model[["resample"]][["RMSE"]])

## [1] 9.388165

#Calculating R2
mean(new_fitted_model[["resample"]][["Rsquared"]])

## [1] 0.6322668
```

Assignment3

```
#Results of linear model on Test data
#Calculating Testing MSE
mean((test$Residuary-predict(fitted_model,test))^2,na.rm=T)

## [1] 75.12459

#Calculating Testing RMSE
sqrt(mean((test$Residuary-predict(fitted_model,test))^2,na.rm=T))

## [1] 8.667444

#calculating TEST R2
rss  <-  sum((test$Residuary-predict(fitted_model,test))^2,na.rm=T)
tss  <- sum((test$Residuary-mean(test$Residuary,na.rm=T))^2,na.rm=T)
r  <- 1- (rss/tss)
r

## [1] 0.6619119

#Results of bootstrap model on Test data
#Calculating Testing MSE
mean((test$Residuary-predict(new_fitted_model,test))^2,na.rm=T)

## [1] 75.12459

#Calculating Testing RMSE
sqrt(mean((test$Residuary-predict(new_fitted_model,test))^2,na.rm=T))

## [1] 8.667444

#calculating TEST R2
rss  <-  sum((test$Residuary-predict(new_fitted_model,test))^2,na.rm=T)
tss  <- sum((test$Residuary-mean(test$Residuary,na.rm=T))^2,na.rm=T)
rsq  <- 1- (rss/tss)
rsq*100

## [1] 66.19119
```

There is no difference in performance of original and bootstrap model on the test set.

==

2.2 Problem 2

```
library(ggplot2)
library(caret)

## Loading required package: lattice

#Importing dataset
URL = "https://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german/german.data-numeric"
data  <-  read.csv(URL,sep='')

#Display the dataset
head(data)
```

Assignment3

```
##   X1 X6 X4 X12 X5 X5.1 X3 X4.1 X1.1 X67 X3.1 X2 X1.2 X2.1 X1.3 X0 X0.1
## 1  2 48  2  60  1    3  2    2    1  22    3  1    1    1    1  0    0
## 2  4 12  4  21  1    4  3    3    1  49    3  1    2    1    1  0    0
## 3  1 42  2  79  1    4  3    4    2  45    3  1    2    1    1  0    0
## 4  1 24  3  49  1    3  3    4    4  53    3  2    2    1    1  1    0
## 5  4 36  2  91  5    3  3    4    4  35    3  1    2    2    1  0    0
## 6  4 24  2  28  3    5  3    4    2  53    3  1    1    1    1  0    0
##   X1.4 X0.2 X0.3 X1.5 X0.4 X0.5 X1.6 X1.7
## 1    1    0    0    1    0    0    1    2
## 2    1    0    0    1    0    1    0    1
## 3    0    0    0    0    0    0    1    1
## 4    1    0    0    0    0    0    1    2
## 5    1    0    0    0    0    1    0    1
## 6    1    0    0    1    0    0    1    1
```

```r
#dimensions of dataset
nrow(data)
```

```
## [1] 999
```

```r
ncol(data)
```

```
## [1] 25
```

```r
colnames(data)[25]  <- c("response")
str(data)
```

```
## 'data.frame':    999 obs. of  25 variables:
##  $ X1      : int  2 4 1 1 4 4 2 4 2 2 ...
##  $ X6      : int  48 12 42 24 36 24 36 12 30 12 ...
##  $ X4      : int  2 4 2 3 2 2 2 2 4 2 ...
##  $ X12     : int  60 21 79 49 91 28 69 31 52 13 ...
##  $ X5      : int  1 1 1 1 5 3 1 4 1 1 ...
##  $ X5.1    : int  3 4 4 3 3 5 3 4 1 2 ...
##  $ X3      : int  2 3 3 3 3 3 3 1 4 2 ...
##  $ X4.1    : int  2 3 4 4 4 4 2 4 2 1 ...
##  $ X1.1    : int  1 1 2 4 4 2 3 1 3 3 ...
##  $ X67     : int  22 49 45 53 35 53 35 61 28 25 ...
##  $ X3.1    : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ X2      : int  1 1 1 2 1 1 1 1 2 1 ...
##  $ X1.2    : int  1 2 2 2 2 1 1 1 1 1 ...
##  $ X2.1    : int  1 1 1 1 2 1 2 1 1 1 ...
##  $ X1.3    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ X0      : int  0 0 0 1 0 0 0 0 1 1 ...
##  $ X0.1    : int  0 0 0 0 0 0 1 0 0 0 ...
##  $ X1.4    : int  1 1 0 1 1 1 1 1 1 1 ...
##  $ X0.2    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ X0.3    : int  0 0 0 0 0 0 1 0 0 1 ...
##  $ X1.5    : int  1 1 0 0 0 1 0 1 1 0 ...
##  $ X0.4    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ X0.5    : int  0 1 0 0 1 0 0 1 0 0 ...
##  $ X1.6    : int  1 0 1 1 0 1 0 0 0 1 ...
##  $ response: int  2 1 1 2 1 1 1 1 2 2 ...
```

Assignment3

```
typeof(data$response)

## [1] "integer"

data$response=factor(data$response)
str(data)

## 'data.frame':    999 obs. of  25 variables:
##  $ X1      : int  2 4 1 1 4 4 2 4 2 2 ...
##  $ X6      : int  48 12 42 24 36 24 36 12 30 12 ...
##  $ X4      : int  2 4 2 3 2 2 2 2 4 2 ...
##  $ X12     : int  60 21 79 49 91 28 69 31 52 13 ...
##  $ X5      : int  1 1 1 1 5 3 1 4 1 1 ...
##  $ X5.1    : int  3 4 4 3 3 5 3 4 1 2 ...
##  $ X3      : int  2 3 3 3 3 3 3 1 4 2 ...
##  $ X4.1    : int  2 3 4 4 4 4 2 4 2 1 ...
##  $ X1.1    : int  1 1 2 4 4 2 3 1 3 3 ...
##  $ X67     : int  22 49 45 53 35 53 35 61 28 25 ...
##  $ X3.1    : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ X2      : int  1 1 1 2 1 1 1 1 2 1 ...
##  $ X1.2    : int  1 2 2 2 2 1 1 1 1 1 ...
##  $ X2.1    : int  1 1 1 1 2 1 2 1 1 1 ...
##  $ X1.3    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ X0      : int  0 0 0 1 0 0 0 0 1 1 ...
##  $ X0.1    : int  0 0 0 0 0 0 1 0 0 0 ...
##  $ X1.4    : int  1 1 0 1 1 1 1 1 1 1 ...
##  $ X0.2    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ X0.3    : int  0 0 0 0 0 0 1 0 0 1 ...
##  $ X1.5    : int  1 1 0 0 0 1 0 1 1 0 ...
##  $ X0.4    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ X0.5    : int  0 1 0 0 1 0 0 1 0 0 ...
##  $ X1.6    : int  1 0 1 1 0 1 0 0 0 1 ...
##  $ response: Factor w/ 2 levels "1","2": 2 1 1 2 1 1 1 1 2 2 ...

#splitting data into train and test data
partition  <- createDataPartition(data$response,p=0.8,list=F)
train <- data[partition,]
test  <- data[-partition,]

#Fitting our logistic model
fitted_model  <- glm(response~.,data=train,family =binomial)

#predicting the results of glm on Train dataset
prob  <- predict(fitted_model,train,type="response")
pred_response  <- rep(1,nrow(train))
pred_response[prob > 0.5]  <- 2
pred_response<-factor(pred_response)
conf<-confusionMatrix(train$response,pred_response)
conf$byClass

##          Sensitivity          Specificity       Pos Pred Value
##            0.8135048            0.6966292            0.9035714
##       Neg Pred Value            Precision               Recall
```

```
##            0.5166667            0.9035714            0.8135048
##                   F1           Prevalence       Detection Rate
##            0.8561760            0.7775000            0.6325000
## Detection Prevalence    Balanced Accuracy
##            0.7000000            0.7550670

#using trainControl and train functions
crossvalidation <- trainControl(method="cv",number = 10)
new_fitted_model <-
train(response~.,data=train,method="glm",family=binomial,trControl=crossvalidation)
new_fitted_model

## Generalized Linear Model
##
## 800 samples
##  24 predictor
##   2 classes: '1', '2'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 720, 720, 720, 720, 720, 720, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.7675    0.4022067

#predicting the results of crossvalidation on Train dataset
new_prob <- predict(new_fitted_model,train,type="prob")
new_pred_response  <- rep(1,nrow(train))
new_pred_response[new_prob[2] > 0.5]  <- 2
new_pred_response<-factor(new_pred_response)
conf<-confusionMatrix(train$response,new_pred_response)
conf$byClass

##          Sensitivity          Specificity       Pos Pred Value
##            0.8135048            0.6966292            0.9035714
##       Neg Pred Value            Precision               Recall
##            0.5166667            0.9035714            0.8135048
##                   F1           Prevalence       Detection Rate
##            0.8561760            0.7775000            0.6325000
## Detection Prevalence    Balanced Accuracy
##            0.7000000            0.7550670
```

```
#predicting the results of glm on Test dataset
prob  <- predict(fitted_model,test,type="response")
pred_response  <- rep(1,nrow(test))
pred_response[prob > 0.5]  <- 2
pred_response<-factor(pred_response)
conf<-confusionMatrix(test$response,pred_response)
conf$byClass
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.8000000            0.6590909            0.8920863
##       Neg Pred Value            Precision               Recall
##            0.4833333            0.8920863            0.8000000
##                   F1           Prevalence       Detection Rate
##            0.8435374            0.7788945            0.6231156
## Detection Prevalence    Balanced Accuracy
##            0.6984925            0.7295455
```

```
#predicting the results of crossvalidation on Test dataset
new_prob <- predict(new_fitted_model,test,type="prob")
new_pred_response  <- rep(1,nrow(test))
new_pred_response[new_prob[2] > 0.5]  <- 2
new_pred_response<-factor(new_pred_response)
conf<-confusionMatrix(test$response,new_pred_response)
conf$byClass
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.8000000            0.6590909            0.8920863
##       Neg Pred Value            Precision               Recall
##            0.4833333            0.8920863            0.8000000
##                   F1           Prevalence       Detection Rate
##            0.8435374            0.7788945            0.6231156
## Detection Prevalence    Balanced Accuracy
##            0.6984925            0.7295455
```

==

2.3 Problem 3

```
#Importing the dataset
car<-data.frame(mtcars)

#display the dataset
head(car)
```

```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
#dimensions of dataset
nrow(car)
```

```
## [1] 32
```

```
ncol(car)
```

```
## [1] 11
```

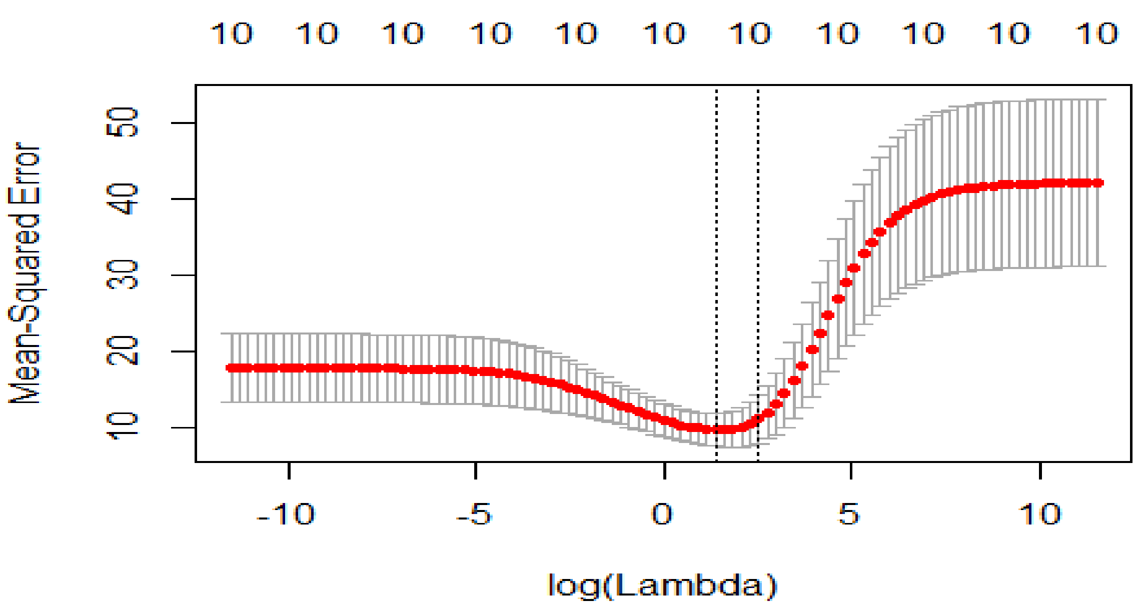
Assignment3

```
#Check Structure of dataset
str(car)

## 'data.frame':    32 obs. of  11 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

set.seed(200)
partition <- createDataPartition(car$am,times=1,p=0.8,list = F)
train <- car[partition,]
test <- car[-partition,]


#fiitting a linear model
model <- lm(mpg~.,data=train)
#MSE on test set
mean((predict(model,test)-test$mpg)^2)

## [1] 10.71549

summary(model)

##
## Call:
## lm(formula = mpg ~ ., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0200 -2.0955 -0.2192  1.3621  4.6315
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -17.79527   34.31617  -0.519   0.6116
## cyl          -0.10885    1.24904  -0.087   0.9317
## disp          0.02193    0.02167   1.012   0.3276
## hp           -0.01242    0.03012  -0.413   0.6858
## drat          0.65269    2.24277   0.291   0.7750
## wt           -5.30058    2.52253  -2.101   0.0529 .
## qsec          2.46523    1.61141   1.530   0.1469
## vs           -2.59087    3.43564  -0.754   0.4625
```

```
## am            2.71842    2.76117   0.985   0.3405
## gear          1.63422    2.10387   0.777   0.4494
## carb          0.07967    1.04162   0.076   0.9400
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.966 on 15 degrees of freedom
## Multiple R-squared:  0.8704, Adjusted R-squared:  0.7841
## F-statistic: 10.08 on 10 and 15 DF,  p-value: 5.523e-05
```

Only attribute wt is seems to be relevant.

```
#Ridge Regression
# Loaging the library
library(glmnet)

# Getting the independent variable
x <- model.matrix(mpg~.,train)[,-1]

# Getting the dependent variable
y <- train$mpg

# Setting the range of lambda values
lambda_seq <- 10^seq(5,-5,by = -.1)

# Using cross validation glmnet
ridge_cv <- cv.glmnet(x, y, alpha = 0,lambda = lambda_seq)

plot(ridge_cv)
```



Assignment3

```
#Best lambda value
best_lambda <- ridge_cv$lambda.min
best_lambda

## [1] 3.981072

# Using glmnet function to build the ridge regression model
fit <- glmnet(x, y, alpha = 0, lambda  = best_lambda)


# Summary
summary(fit)

##           Length Class     Mode
## a0         1     -none-    numeric
## beta      10     dgCMatrix S4
## df         1     -none-    numeric
## dim        2     -none-    numeric
## lambda     1     -none-    numeric
## dev.ratio  1     -none-    numeric
## nulldev    1     -none-    numeric
## npasses    1     -none-    numeric
## jerr       1     -none-    numeric
## offset     1     -none-    logical
## call       5     -none-    call
## nobs       1     -none-    numeric

#for testdatast
xx =  model.matrix(mpg~.,test)[,-1]
model_predict <- predict(fit,s =,newx = xx, type = "response")

#MSE on test data using Ridge
mean((model_predict-test$mpg)^2)

## [1] 1.184656

#coefficients of glm model:
(Intercept)  )
xz
```

We can see that the MSE on test data will decreases from 10.71549 to 1.184656 by performing Ridge Regression.

```
# Coefficient of glm model:
coef(model)
##                        1
## (Intercept) -17.79526837
## cyl           0.108853352
## disp          0.02193177
## hp           -0.01242459
## drat          0.65268664
## wt           -5.30057738
## qsec          2.46523037
## vs           -2.59087201
```

```
## am          2.71842115
## gear        1.63421704
## carb        0.07966846
```

```
# Coefficient of Ridge regression model:
coef(ridge_cv,s="lambda.min")
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept) 19.533705869
## cyl         -0.368008786
## disp        -0.005720897
## hp          -0.011099008
## drat         1.156418468
## wt          -1.109528763
## qsec         0.203566030
## vs           0.804978288
## am           1.520934064
## gear         0.588710051
## carb        -0.497348516
```

Compared to linear fit rigid regularization has shrinked the coefficients and some of them are shrinked close to zero.

==

2.4 Problem 4

```
library(ggplot2)
library(lattice)
library(caret)
#Importing the dataset
data <- data.frame(swiss)

#display the dataset
head(data)
```

```
##              Fertility Agriculture Examination Education Catholic
## Courtelary        80.2        17.0          15        12     9.96
## Delemont          83.1        45.1           6         9    84.84
## Franches-Mnt      92.5        39.7           5         5    93.40
## Moutier           85.8        36.5          12         7    33.77
## Neuveville        76.9        43.5          17        15     5.16
## Porrentruy        76.1        35.3           9         7    90.57
##              Infant.Mortality
## Courtelary               22.2
## Delemont                 22.2
## Franches-Mnt             20.2
## Moutier                  20.3
## Neuveville               20.6
## Porrentruy               26.6
```

```
#dimensions of dataset
nrow(data)
```

Assignment3

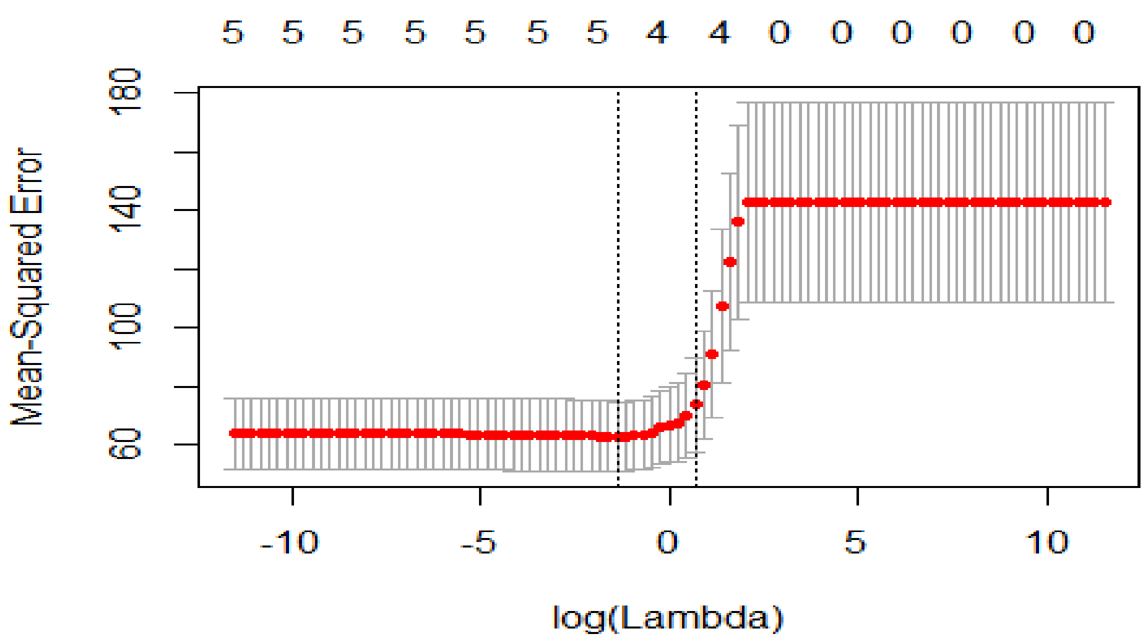
```
## [1] 47

ncol(data)

## [1] 6

#80-20 split using createDataPartition
set.seed(150)
partition <- createDataPartition(data$Fertility,p=0.8,list = F)
train <- data[partition,]
test <- data[-partition,]

#fitting a linear fit
model <- lm(Fertility~.,train)

summary(model)

##
## Call:
## lm(formula = Fertility ~ ., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.014  -5.942   1.329   3.491  15.717
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      66.16966   11.76082   5.626 2.90e-06 ***
## Agriculture      -0.17497    0.07982  -2.192  0.03552 *
## Examination      -0.05176    0.29772  -0.174  0.86303
## Education        -1.06932    0.23606  -4.530 7.32e-05 ***
## Catholic          0.11713    0.03946   2.969  0.00554 **
## Infant.Mortality  1.03247    0.41295   2.500  0.01756 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.167 on 33 degrees of freedom
## Multiple R-squared:  0.6893, Adjusted R-squared:  0.6422
## F-statistic: 14.64 on 5 and 33 DF,  p-value: 1.406e-07
```

Agriculture, Education, Catholic and Infant Mortality are relevant feature with coefficients as -0.17497, 0.05176, 0.11713, 1.03247

```
#calculating test MSE
mean((test$Fertility-predict(model,test))^2)

## [1] 59.91027

#lasso
# Loaging the library
library(Matrix)
library(foreach)
library(glmnet)
```

Assignment3

```
## Loaded glmnet 2.0-18

# Getting the independent variable
x <- model.matrix(Fertility~.,train)[,-1]

# Getting the dependent variable
y <- train$Fertility

# Setting the range of lambda values
lambda_seq <- 10^seq(5,-5,by = -.1)

# Using cross validation glmnet
lasso_cv <- cv.glmnet(x, y, alpha = 1,lambda = lambda_seq)
plot(lasso_cv)
```




```
#Best lambda value
best_lambda <- lasso_cv$lambda.min
best_lambda

## [1] 0.2511886

# Using glmnet function to build the ridge regression model
fit <- glmnet(x, y, alpha = 1, lambda  = best_lambda)

# Checking the model
summary(fit)

##           Length Class     Mode
## a0        1      -none-    numeric
## beta      5      dgCMatrix S4
## df        1      -none-    numeric
```

```
## dim        2      -none-    numeric
## lambda     1      -none-    numeric
## dev.ratio  1      -none-    numeric
## nulldev    1      -none-    numeric
## npasses    1      -none-    numeric
## jerr       1      -none-    numeric
## offset     1      -none-    logical
## call       5      -none-    call
## nobs       1      -none-    numeric

#for testdata
xx =  model.matrix(Fertility~.,test)[,-1]
model_predict <- predict(fit,s =,newx = xx, type = "response")

#MSE on test data
mean((model_predict-test$Fertility)^2)

## [1] 57.83554

# Coefficient of lm model:
coef(model)
## (Intercept)      66.16965921
## Agriculture      -0.17497395
## Examination      -0.05176448
## Education        -1.06932048
## Catholic          0.11713319
## Infant.Mortality  1.03247401

# Coefficient of Lasso regression model:
 coef(lasso_cv)

## 6 x 1 sparse Matrix of class "dgCMatrix"
##                            1
## (Intercept)      60.59242105
## Agriculture       .
## Examination       .
## Education        -0.62205775
## Catholic          0.06463855
## Infant.Mortality  0.69070657
```

Compared to linear fit Lasso regularization has shrinked the coefficients and two of them are shrinked to zero.

==