CS 6301.012

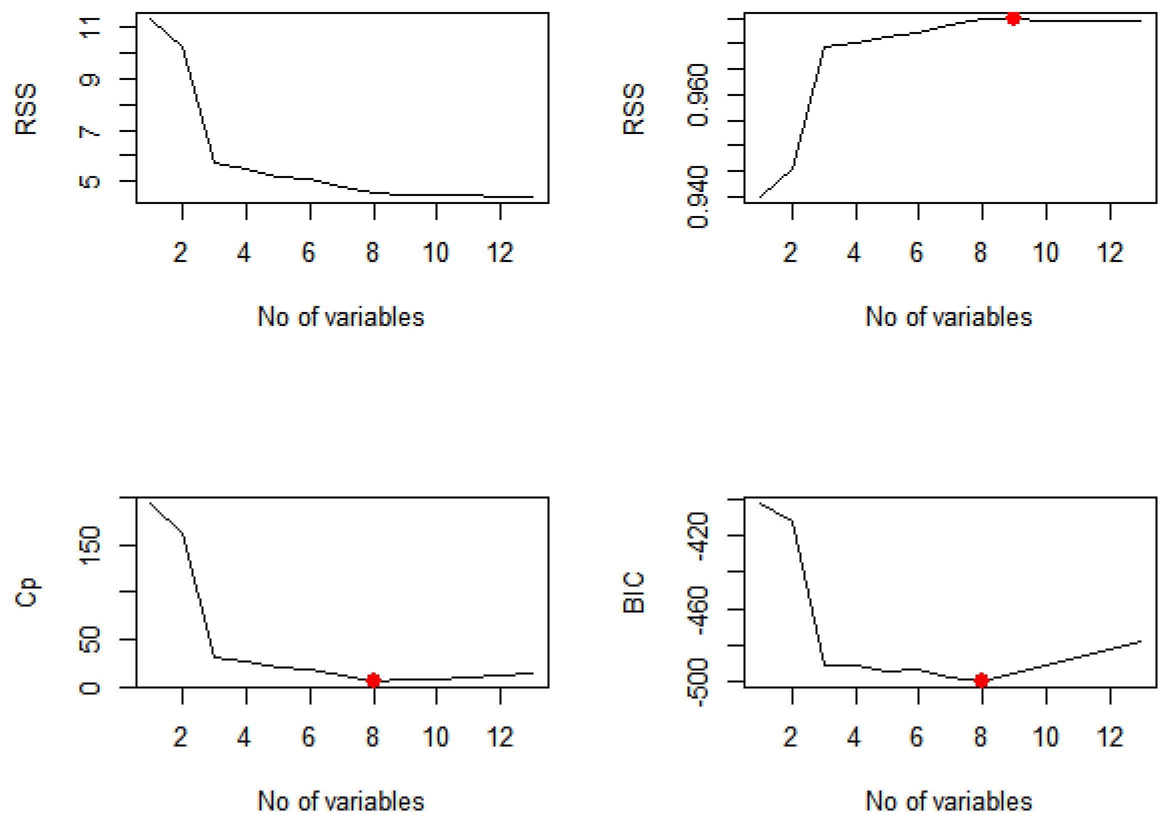
Advanced Computing for Data Science

Assignment VI

Sushmitha Mohan Raj sxm144630

Sreesha Nagaraj sxn146630

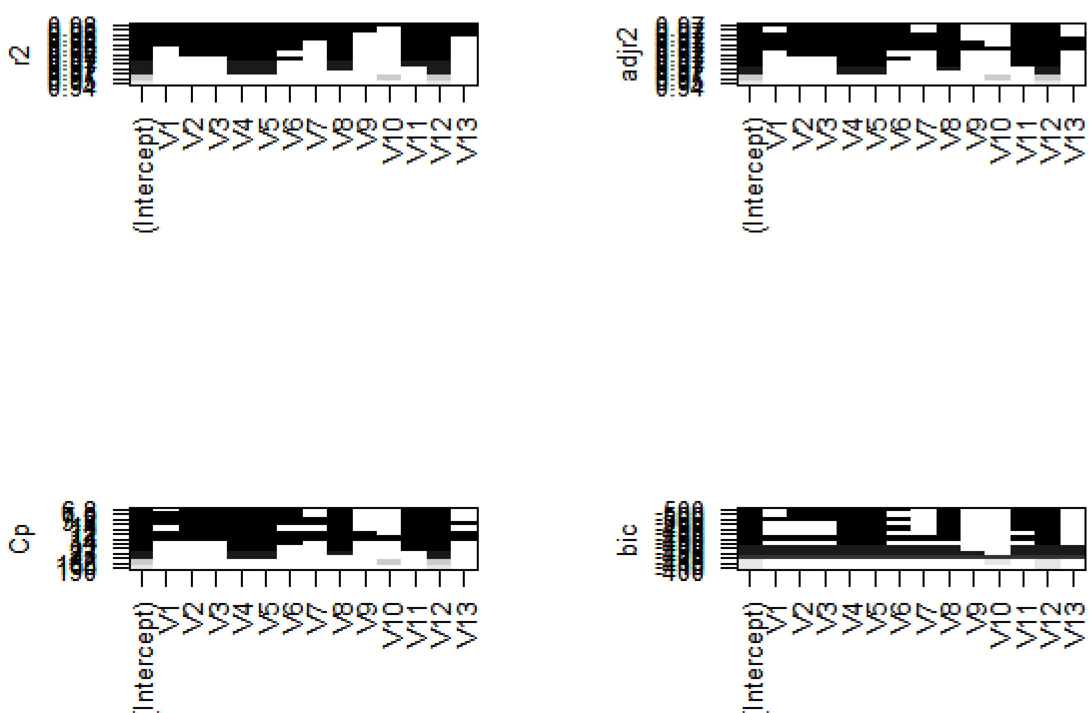
LPGA Winnings:




```
> subsets_full = regsubsets(lpga2009$V14 ~., data=lpga2009)
> subsets_full.summary = summary(subsets_full)
> subsets_full.summary
Subset selection object
Call: regsubsets.formula(lpga2009$V14 ~ ., data = lpga2009)
13 Variables  (and intercept)
    Forced in Forced out
V1      FALSE      FALSE
V2      FALSE      FALSE
V3      FALSE      FALSE
V4      FALSE      FALSE
V5      FALSE      FALSE
V6      FALSE      FALSE
V7      FALSE      FALSE
V8      FALSE      FALSE
V9      FALSE      FALSE
V10     FALSE      FALSE
V11     FALSE      FALSE
V12     FALSE      FALSE
V13     FALSE      FALSE
1 subsets of each size up to 8
Selection Algorithm: exhaustive
         V1  V2  V3  V4  V5  V6  V7  V8  V9  V10 V11 V12 V13
```

```
1  ( 1 ) " " " " " " " " " " " " " " " " " " " " " " "*" " "
2  ( 1 ) " " " " " " " " " " " " " " " " " " "*" " " "*" " "
3  ( 1 ) " " " " " " "*" "*" " " " " " " " " " " " " "*" " "
4  ( 1 ) " " " " " " "*" "*" " " " " "*" " " " " " " "*" " "
5  ( 1 ) " " " " " " "*" "*" " " " " "*" " " " " "*" "*" " "
6  ( 1 ) " " " " " " "*" "*" "*" " " "*" " " " " "*" "*" " "
7  ( 1 ) " " "*" "*" "*" "*" " " " " "*" " " " " "*" "*" " "
8  ( 1 ) " " "*" "*" "*" "*" "*" " " "*" " " " " "*" "*" " "
> subsets_full_13 = regsubsets(lpga2009$V14 ~., data=lpga2009,nvmax = 14)
> subsets_full_13.summary = summary(subsets_full_13)
> subsets_full_13.summary
Subset selection object
Call: regsubsets.formula(lpga2009$V14 ~ ., data = lpga2009, nvmax = 14)
13 Variables  (and intercept)
    Forced in Forced out
V1      FALSE      FALSE
V2      FALSE      FALSE
V3      FALSE      FALSE
V4      FALSE      FALSE
V5      FALSE      FALSE
V6      FALSE      FALSE
V7      FALSE      FALSE
V8      FALSE      FALSE
V9      FALSE      FALSE
V10     FALSE      FALSE
V11     FALSE      FALSE
V12     FALSE      FALSE
V13     FALSE      FALSE
1 subsets of each size up to 13
Selection Algorithm: exhaustive
          V1  V2  V3  V4  V5  V6  V7  V8  V9  V10 V11 V12 V13
1  ( 1 )  " " " " " " " " " " " " " " " " " " " " " " "*" " "
2  ( 1 )  " " " " " " " " " " " " " " " " " " "*" " " "*" " "
3  ( 1 )  " " " " " " "*" "*" " " " " " " " " " " " " "*" " "
4  ( 1 )  " " " " " " "*" "*" " " " " "*" " " " " " " "*" " "
5  ( 1 )  " " " " " " "*" "*" " " " " "*" " " " " "*" "*" " "
6  ( 1 )  " " " " " " "*" "*" "*" " " "*" " " " " "*" "*" " "
7  ( 1 )  " " "*" "*" "*" "*" " " " " "*" " " " " "*" "*" " "
8  ( 1 )  " " "*" "*" "*" "*" "*" " " "*" " " " " "*" "*" " "
9  ( 1 )  "*" "*" "*" "*" "*" "*" " " "*" " " " " "*" "*" " "
10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" " " " " "*" "*" " "
11  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" " " " " "*" "*" "*"
12  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" " " "*" "*" "*"
13  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
> par(mfrow=c(2,2))
> plot(subsets_full_13$rss,xlab = "No of variables", ylab = "RSS", type = "l"
)
> names(subsets_full_13.summary)
[1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
> plot(subsets_full_13.summary$rss,xlab = "No of variables", ylab = "RSS", ty
pe = "l")
> plot(subsets_full_13.summary$rss,xlab = "No of variables", ylab = "RSS", ty
pe = "l")
> par(mfrow=c(2,2))
> plot(subsets_full_13.summary$rss,xlab = "No of variables", ylab = "RSS", ty
pe = "l")
```

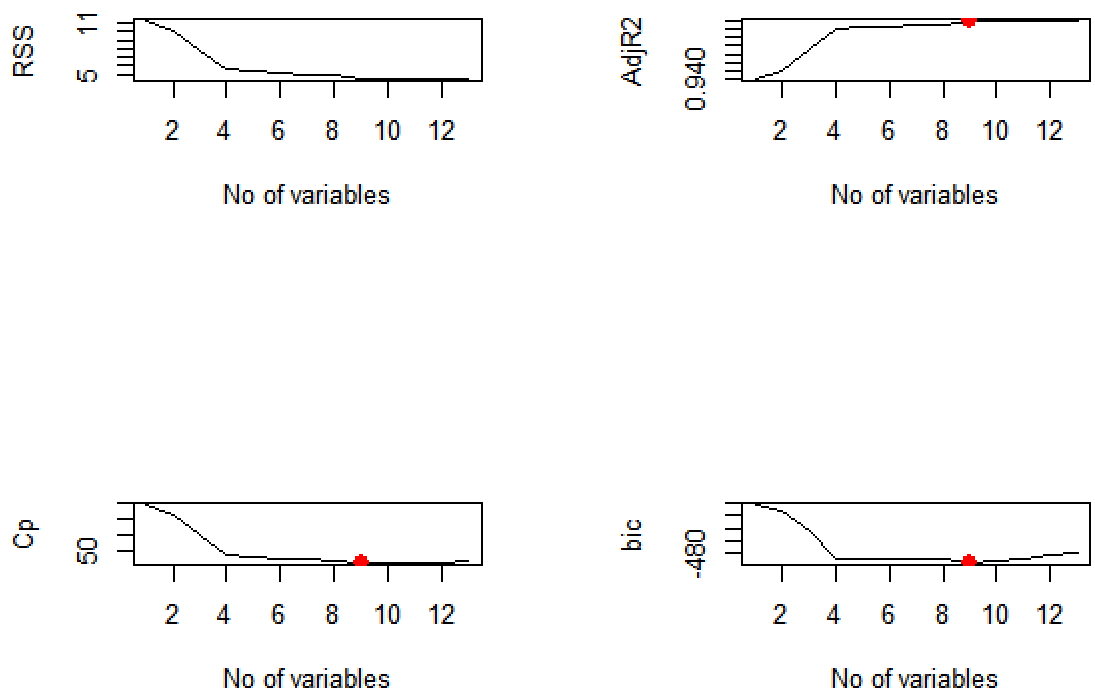
```
> plot(subsets_full_13.summary$adjr2,xlab = "No of variables", ylab = "RSS",
type = "l")
> which.max(subsets_full_13.summary)
Error in which.max(subsets_full_13.summary) :
  (list) object cannot be coerced to type 'double'
> which.max(subsets_full_13.summary$adjr2)
[1] 9
> points(9,subsets_full_13.summary$adjr2[9],col="red",cex=2,pch=20)
> plot(subsets_full_13.summary$cp,xlab = "No of variables", ylab = "Cp", type
= "l")
> which.min(subsets_full_13.summary$cp)
[1] 8
> points(8,subsets_full_13.summary$cp[8],col="red",cex=2,pch=20)
> plot(subsets_full_13.summary$bic,xlab = "No of variables", ylab = "BIC", ty
pe = "l")
> which.min(subsets_full_13.summary$bic)
[1] 8
> points(8,subsets_full_13.summary$bic[8],col="red",cex=2,pch=20)


> plot(subsets_full_13,scale="r2")
> plot(subsets_full_13,scale="adjr2")
> plot(subsets_full_13,scale="Cp")
> plot(subsets_full_13,scale="bic")
```



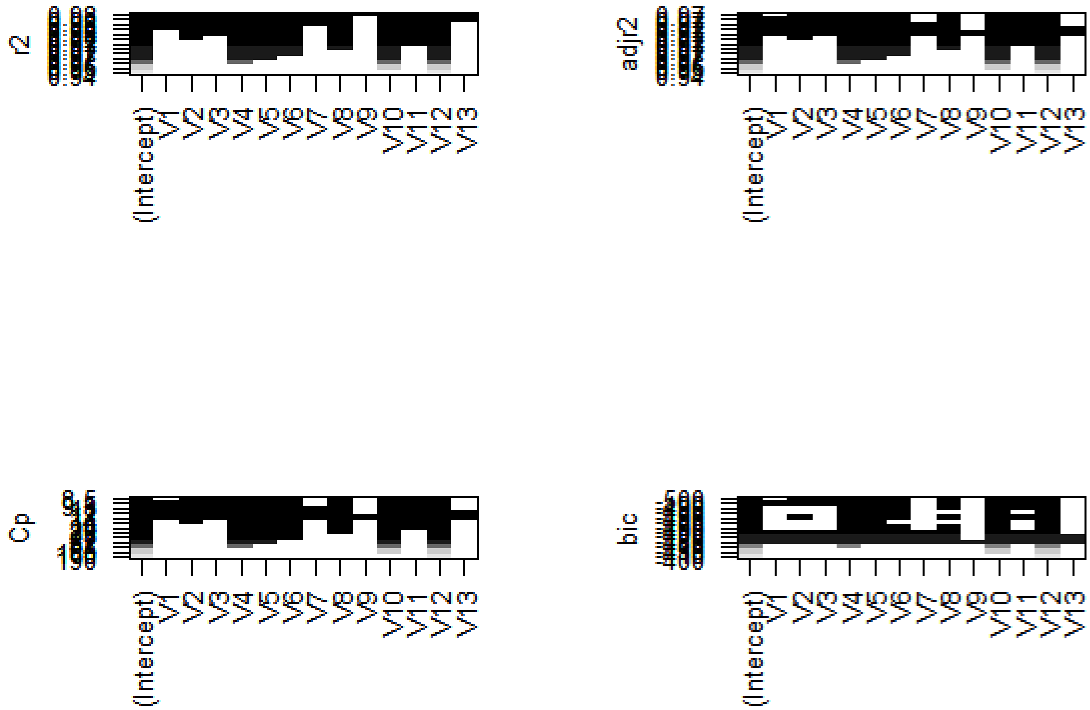
Forward Subset Selection:

```
> subset_fwd = regsubsets(V14~., data=lpga2009, method = "forward",nvmax = 13
)
> subset_fwd.summary = summary(subset_fwd)
> subset_fwd.summary
Subset selection object
Call: regsubsets.formula(V14 ~ ., data = lpga2009, method = "forward",
    nvmax = 13)
13 Variables  (and intercept)
    Forced in Forced out
V1      FALSE      FALSE
V2      FALSE      FALSE
V3      FALSE      FALSE
V4      FALSE      FALSE
V5      FALSE      FALSE
V6      FALSE      FALSE
V7      FALSE      FALSE
V8      FALSE      FALSE
V9      FALSE      FALSE
V10     FALSE      FALSE
V11     FALSE      FALSE
V12     FALSE      FALSE
V13     FALSE      FALSE
1 subsets of each size up to 13
Selection Algorithm: forward
          V1  V2  V3  V4  V5  V6  V7  V8  V9  V10 V11 V12 V13
1  ( 1 )  " " " " " " " " " " " " " " " " " " " " " " "*" " "
2  ( 1 )  " " " " " " " " " " " " " " " " " " "*" " " "*" " "
3  ( 1 )  " " " " " " "*" " " " " " " " " " " "*" " " "*" " "
4  ( 1 )  " " " " " " "*" "*" " " " " " " " " "*" " " "*" " "
5  ( 1 )  " " " " " " "*" "*" "*" " " " " " " "*" " " "*" " "
6  ( 1 )  " " " " " " "*" "*" "*" " " "*" " " "*" " " "*" " "
7  ( 1 )  " " " " " " "*" "*" "*" " " "*" " " "*" "*" "*" " "
8  ( 1 )  " " " " "*" " " "*" "*" "*" " " "*" " " "*" "*" "*" " "
9  ( 1 )  " " "*" "*" "*" "*" "*" " " "*" " " "*" "*" "*" " "
10  ( 1 ) "*" "*" "*" "*" "*" "*" " " "*" " " "*" "*" "*" " "
11  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" " " "*" "*" "*" " "
12  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" " " "*" "*" "*" "*"
13  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"


> par(mfrow=c(2,2))
> plot(subset_fwd.summary$rss,xlab = "No of variables", ylab = "RSS", type =
"l")
> plot(subset_fwd.summary$adjr2,xlab = "No of variables", ylab = "AdjR2", typ
e = "l")
> which.max(subset_fwd.summary$adjr2)
[1] 9
> points(9, subset_fwd.summary$adjr2[9], col="red", cex=2,pch=20)
> plot(subset_fwd.summary$cp,xlab = "No of variables", ylab = "Cp", type = "l
")
> which.min(subset_fwd.summary$cp)
[1] 9
> points(9, subset_fwd.summary$cp[9], col="red", cex=2,pch=20)
> plot(subset_fwd.summary$bic,xlab = "No of variables", ylab = "bic", type =
"l")
```

```
> which.min(subset_fwd.summary$bic)
[1] 9
> points(9, subset_fwd.summary$bic[9], col="red", cex=2,pch=20)
```




```
> plot(subset_fwd,scale = "r2")
> plot(subset_fwd,scale = "adjr2")
> plot(subset_fwd,scale = "Cp")
> plot(subset_fwd,scale = "bic")
```

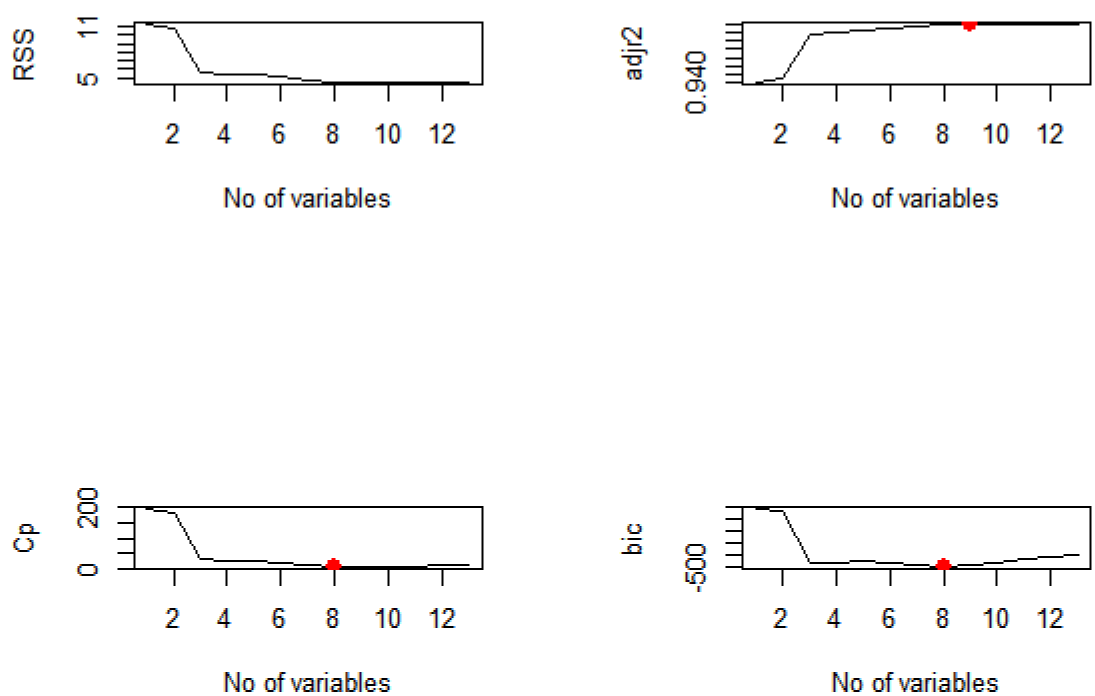


Backward Stepwise Selection

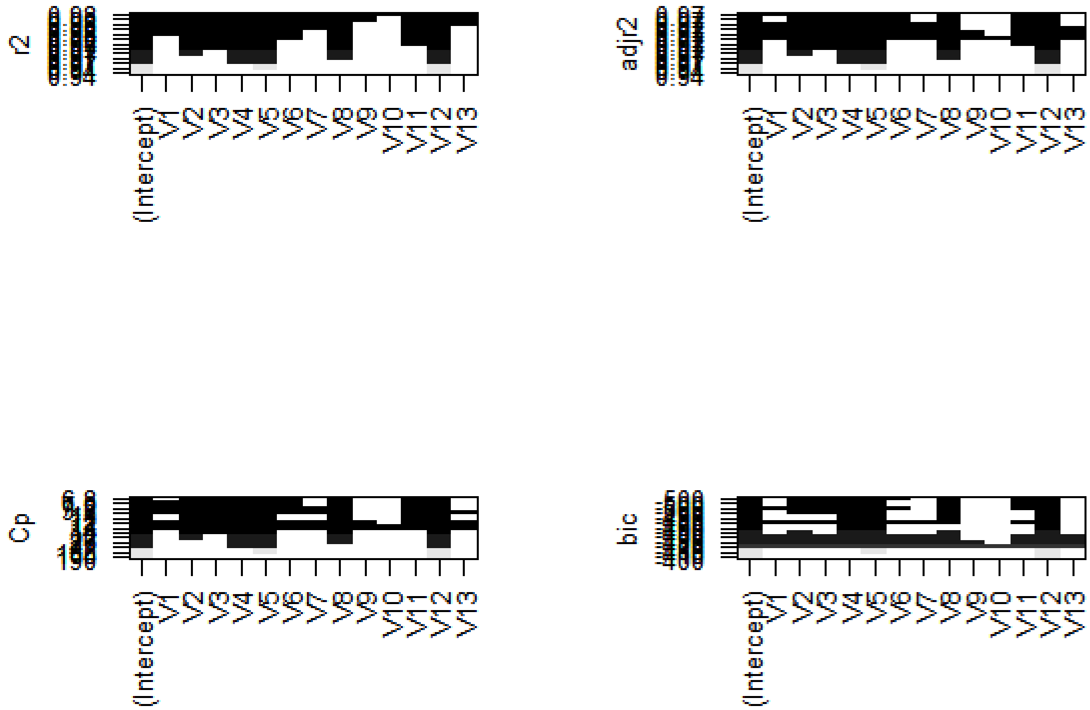
```
subset_bwd = regsubsets(V14~., data=lpga2009, method = "backward",nvmax = 13)
> subset_bwd.summary = summary(subset_bwd)
> subset_bwd.summary
Subset selection object
Call: regsubsets.formula(V14 ~ ., data = lpga2009, method = "backward",
    nvmax = 13)
13 Variables  (and intercept)
    Forced in Forced out
V1      FALSE      FALSE
V2      FALSE      FALSE
V3      FALSE      FALSE
V4      FALSE      FALSE
V5      FALSE      FALSE
V6      FALSE      FALSE
V7      FALSE      FALSE
V8      FALSE      FALSE
V9      FALSE      FALSE
V10     FALSE      FALSE
V11     FALSE      FALSE
V12     FALSE      FALSE
V13     FALSE      FALSE
1 subsets of each size up to 13
```

```
Selection Algorithm: backward
          V1  V2  V3  V4  V5  V6  V7  V8  V9  V10 V11 V12 V13
1  ( 1 )  " " " " " " " " " " " " " " " " " " " " " " "*" " "
2  ( 1 )  " " " " " " " " "*" " " " " " " " " " " " " "*" " "
3  ( 1 )  " " " " " " "*" "*" " " " " " " " " " " " " "*" " "
4  ( 1 )  " " " " " " "*" "*" " " " " "*" " " " " " " "*" " "
5  ( 1 )  " " "*" " " "*" "*" " " " " "*" " " " " " " "*" " "
6  ( 1 )  " " "*" "*" "*" "*" " " " " "*" " " " " " " "*" " "
7  ( 1 )  " " "*" "*" "*" "*" " " " " "*" " " " " "*" "*" " "
8  ( 1 )  " " "*" "*" "*" "*" "*" " " "*" " " " " "*" "*" " "
9  ( 1 )  "*" "*" "*" "*" "*" "*" " " "*" " " " " "*" "*" " "
10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" " " " " "*" "*" " "
11  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" " " " " "*" "*" "*"
12  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" " " "*" "*" "*"
13  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

```
points(9, subset_bwd.summary$adjr2[9], col="red", cex=2,pch=20)
> plot(subset_bwd.summary$rss,xlab = "No of variables", ylab = "RSS", type =
"l")
> plot(subset_bwd.summary$adjr2,xlab = "No of variables", ylab = "adjr2", typ
e = "l")
> which.max(subset_bwd.summary$adjr2)
[1] 9
> points(9, subset_bwd.summary$adjr2[9], col="red", cex=2,pch=20)
> plot(subset_bwd.summary$cp,xlab = "No of variables", ylab = "Cp", type = "l
")
> which.min(subset_bwd.summary$cp)
[1] 8
> points(8, subset_bwd.summary$cp[8], col="red", cex=2,pch=20)
> plot(subset_bwd.summary$bic,xlab = "No of variables", ylab = "bic", type =
"l")
> which.min(subset_bwd.summary$bic)
[1] 8
> points(8, subset_bwd.summary$bic[8], col="red", cex=2,pch=20)
```




```
> plot(subset_bwd, scale = "r2")
> plot(subset_bwd, scale = "adjr2")
> plot(subset_bwd, scale = "Cp")
> plot(subset_bwd, scale = "bic")
```




```
> coef(subsets_full_13,13)
  (Intercept)            V1            V2            V3            V4            
V5
 7.130639e+01 -2.839141e-04 -1.017259e-02 -1.272097e-02 -9.737279e-02  4.5076
19e-01
           V6            V7            V8            V9           V10           
V11
-5.523654e-03 -6.760145e-08 -1.271894e-01 -1.042326e-02  5.079487e-01  9.9188
72e-03
          V12           V13
-3.400786e-02  6.022010e-03
> coef(subset_fwd,13)
  (Intercept)            V1            V2            V3            V4            
V5
 7.130639e+01 -2.839141e-04 -1.017259e-02 -1.272097e-02 -9.737279e-02  4.5076
19e-01
           V6            V7            V8            V9           V10           
V11
-5.523654e-03 -6.760145e-08 -1.271894e-01 -1.042326e-02  5.079487e-01  9.9188
72e-03
          V12           V13
-3.400786e-02  6.022010e-03
> coef(subset_bwd,13)
```

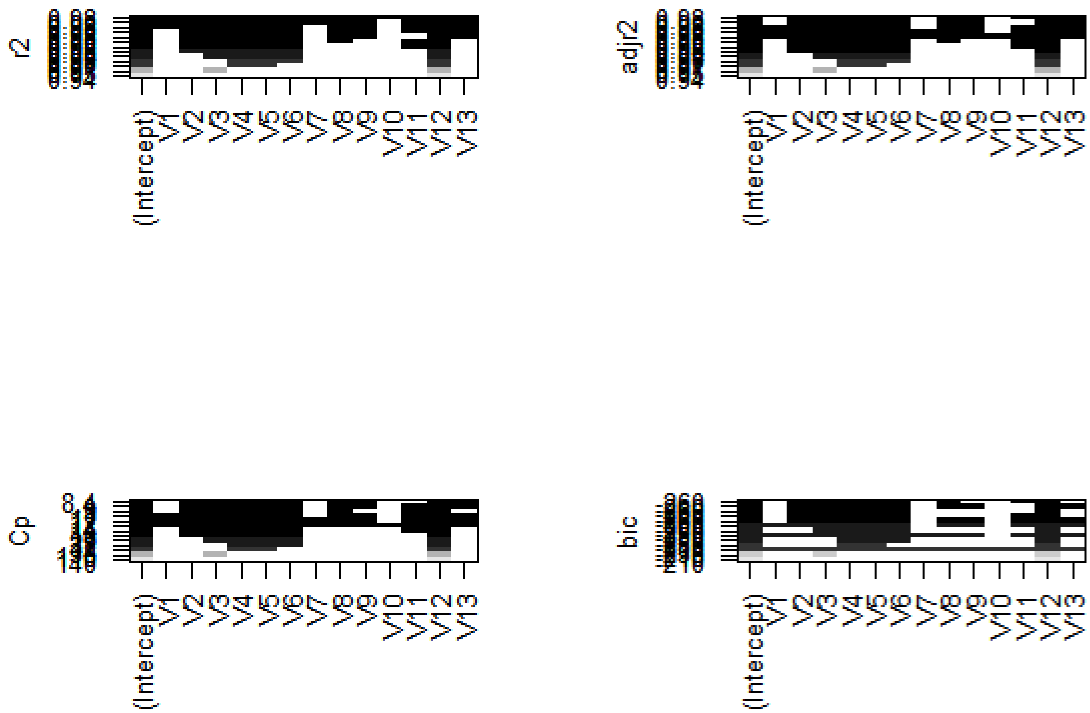
```
  (Intercept)            V1            V2            V3            V4            
V5 
 7.130639e+01 -2.839141e-04 -1.017259e-02 -1.272097e-02 -9.737279e-02  4.5076
19e-01 
           V6            V7            V8            V9           V10           
V11 
-5.523654e-03 -6.760145e-08 -1.271894e-01 -1.042326e-02  5.079487e-01  9.9188
72e-03 
          V12           V13 
-3.400786e-02  6.022010e-03
```

Using Cross Validation Approach

```
> set.seed(1)
> train = sample(c(TRUE,FALSE), nrow(lpga2009),rep=TRUE)
> test = (!train)
> subset_train_full = regsubsets(V14 ~., data=lpga2009[train,],nvmax = 13)
> train_full_Summary = summary(subset_train_full)
> train_full_Summary
Subset selection object
Call: regsubsets.formula(V14 ~ ., data = lpga2009[train, ], nvmax = 13)
13 Variables  (and intercept)
    Forced in Forced out
V1      FALSE      FALSE
V2      FALSE      FALSE
V3      FALSE      FALSE
V4      FALSE      FALSE
V5      FALSE      FALSE
V6      FALSE      FALSE
V7      FALSE      FALSE
V8      FALSE      FALSE
V9      FALSE      FALSE
V10     FALSE      FALSE
V11     FALSE      FALSE
V12     FALSE      FALSE
V13     FALSE      FALSE
1 subsets of each size up to 13
Selection Algorithm: exhaustive
          V1  V2  V3  V4  V5  V6  V7  V8  V9  V10 V11 V12 V13
1  ( 1 )  " " " " " " " " " " " " " " " " " " " " " " "*" " "
2  ( 1 )  " " " " "*" " " " " " " " " " " " " " " " " "*" " "
3  ( 1 )  " " " " " " "*" "*" " " " " " " " " " " " " "*" " "
4  ( 1 )  " " " " " " "*" "*" "*" " " " " " " " " " " "*" " "
5  ( 1 )  " " " " "*" "*" "*" "*" " " " " " " " " " " "*" " "
6  ( 1 )  " " "*" "*" "*" "*" "*" " " " " " " " " " " "*" " "
7  ( 1 )  " " "*" "*" "*" "*" "*" " " " " " " " " "*" "*" " "
8  ( 1 )  " " "*" "*" "*" "*" "*" " " "*" " " " " "*" "*" " "
9  ( 1 )  " " "*" "*" "*" "*" "*" " " "*" "*" " " " " "*" "*"
10  ( 1 ) " " "*" "*" "*" "*" "*" " " "*" "*" " " "*" "*" "*"
11  ( 1 ) "*" "*" "*" "*" "*" "*" " " "*" "*" " " "*" "*" "*"
12  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" " " "*" "*" "*"
13  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

```
> plot(subset_train_full, scale="r2")
> plot(subset_train_full, scale="adjr2")
```

```
> plot(subset_train_full, scale="Cp")
> plot(subset_train_full, scale="bic")
```




```
> test.mat=model.matrix(V14 ~., data= lpga2009[test,])
> val.errors =rep(NA ,13)
> for(i in 1:13){
+ coefi= coef(subset_train_full,id=i)
+ pred = test.mat[,names(coefi)]%*%coefi
+ val.errors[i]=mean((lpga2009$V14[test]-pred)^2)
+ }
> val.errors
 [1] 0.08415571 0.10273877 0.04445451 0.04549853 0.05412799 0.05313093 0.0551
6970
 [8] 0.05017166 0.05133944 0.05329075 0.05279580 0.05452616 0.05458574
> which.min(val.errors)
[1] 3
```

We find that the best model is the one that contains three variables.

```
> coef(subset_train_full,3)
(Intercept)          V4          V5         V12
66.67275719 -0.12176710  0.53265360 -0.03739409
```

Coefficients on the full data set

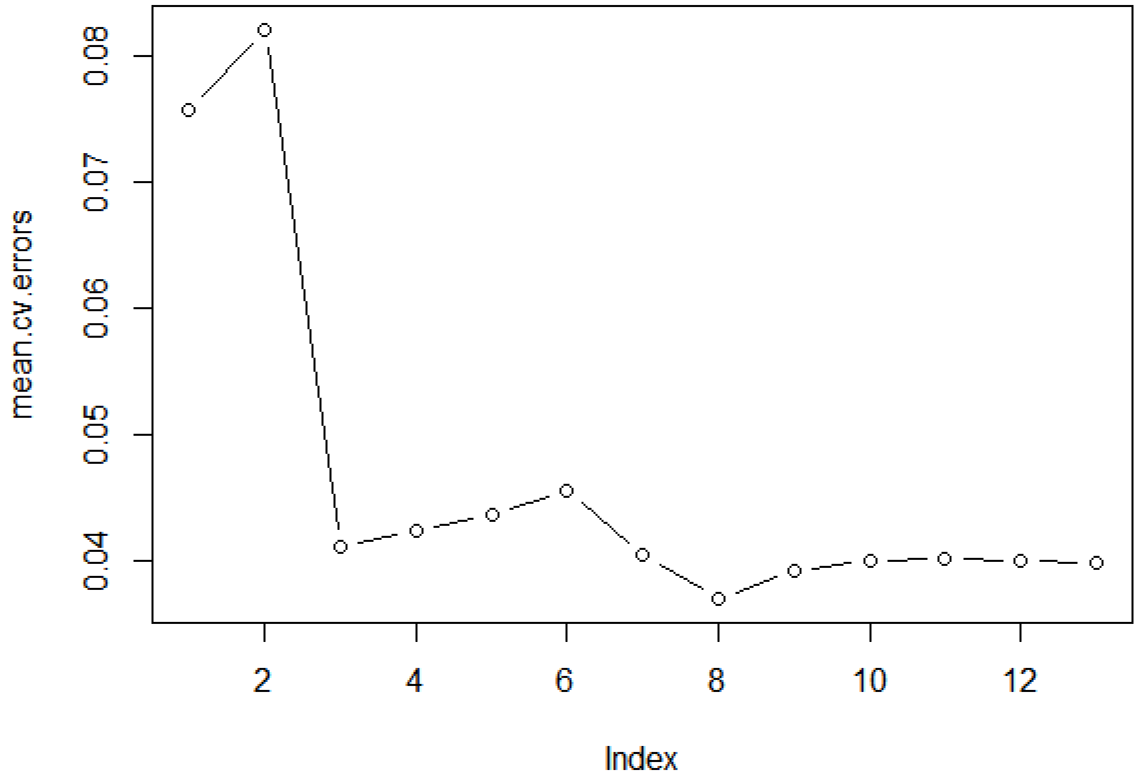
```
> subset_full = regsubsets(V14 ~., data=lpga2009,nvmax = 13)
```

```
> coef(subset_full,3)
(Intercept)          V4          V5         V12
66.88662715 -0.11665632  0.51442503 -0.03802276
```

Cross-Validation

```
k=10
> set.seed(1)
> folds=sample (1:k,nrow(lpga2009),replace =TRUE)
> cv.errors =matrix (NA ,k,13, dimnames =list(NULL , paste (1:13) ))
> for(j in 1:k){
+     best.fit =regsubsets(V14~.,data=lpga2009[folds!=j,], nvmax=13)
+     for(i in 1:13){
+         pred=predict.regsubsets(best.fit,lpga2009[folds==j,],id=i)
+         cv.errors [j,i]=mean((V14[folds==j]-pred)^2)
+     }
+ }
>
> mean.cv.errors =apply(cv.errors ,2, mean)
> mean.cv.errors
         1          2          3          4          5          6          7
0.07574207 0.08217386 0.04109246 0.04235880 0.04366251 0.04562376 0.04044721
         8          9         10         11         12         13
0.03697168 0.03924182 0.04003496 0.04025059 0.03999827 0.03983661
```

The Cross-Validation select model with 8 variables




```
> subset.best = regsubsets(V14~.,data=lpga2009,nvmax = 13)
> coef(subset.best,8)
 (Intercept)           V2           V3           V4           V5           V6
71.993534526 -0.010499992 -0.012789745 -0.096948026  0.464373516 -0.005613287
          V8          V11          V12
-0.129541406  0.019951333 -0.035587780
```

Ridge Regression:

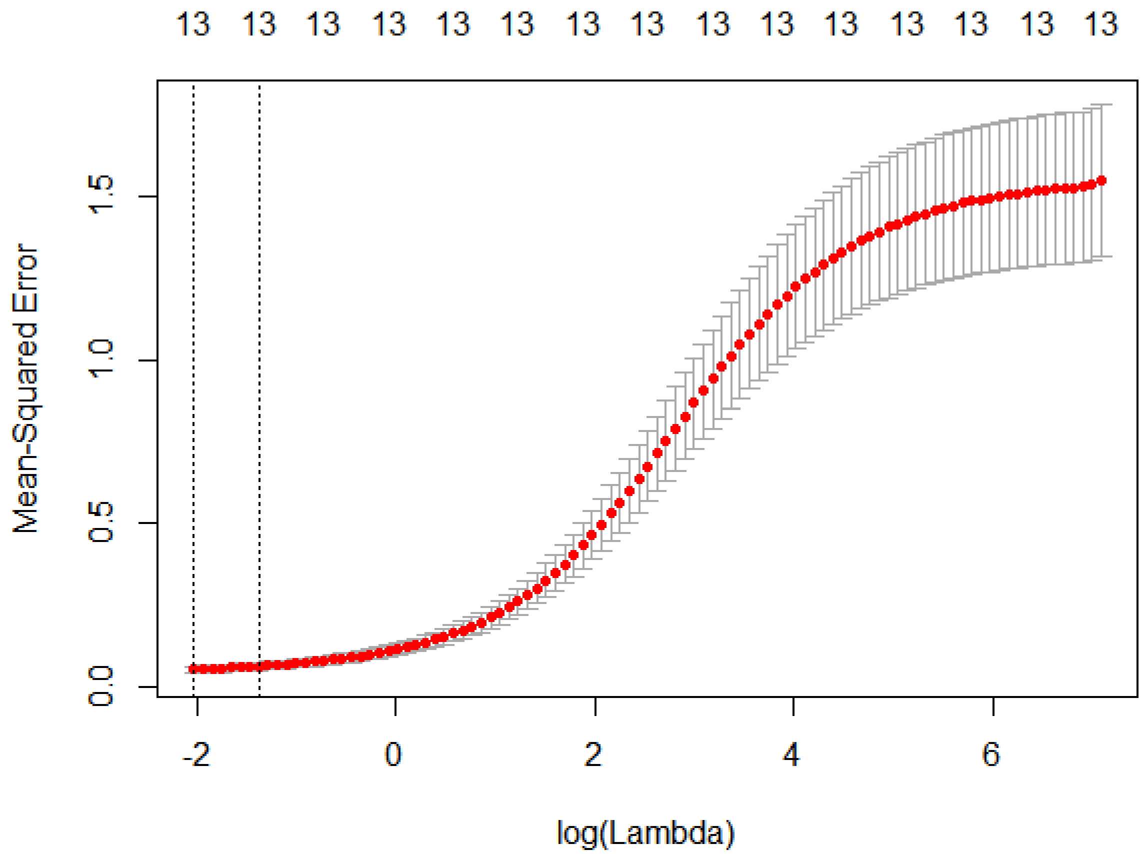
```
> grid =10^ seq (10,-2, length =100)
> ridge.mod =glmnet (x,y,alpha =0, lambda =grid)
> dim(coef(ridge.mod ))
[1]  14 100
> ridge.mod$lambda [50]
[1] 11497.57
> coef(ridge.mod)[,50]
  (Intercept)            V1            V2            V3            V4
V5
 7.267457e+01 -5.163723e-07 -6.036248e-06 -4.671971e-06 -2.094081e-05  7.2719
69e-05
```

```
            V6              V7              V8              V9             V10
V11
-3.193423e-06 -2.445436e-10 -7.333001e-05 -1.610827e-05  2.166955e-03 -1.5160
85e-05
          V12             V13
-7.247062e-06 -4.785361e-06
> sqrt(sum(coef(ridge.mod)[ -1 ,50]^2) )
[1] 0.002169662


> sqrt(sum(coef(ridge.mod)[ -1 ,50]^2) )
[1] 0.002169662
> ridge.mod$lambda [60]
[1] 705.4802
> coef(ridge.mod)[,60]
  (Intercept)             V1              V2              V3              V4
V5
 7.266415e+01 -8.330121e-06 -9.740309e-05 -7.567890e-05 -3.382856e-04  1.1742
84e-03
            V6              V7              V8              V9             V10
V11
-5.148019e-05 -3.945858e-09 -1.183264e-03 -2.592571e-04  3.498474e-02 -2.4450
67e-04
          V12             V13
-1.170228e-04 -7.713420e-05
> sqrt(sum(coef(ridge.mod)[ -1 ,60]^2) )
[1] 0.03502842

> predict(ridge.mod ,s=50, type ="coefficients")[1:14 ,]
  (Intercept)             V1              V2              V3              V4
V5
 7.253018e+01 -1.045822e-04 -1.227944e-03 -1.004749e-03 -4.330192e-03  1.4955
21e-02
            V6              V7              V8              V9             V10
V11
-6.402104e-04 -4.968126e-08 -1.490318e-02 -3.146094e-03  4.437079e-01 -3.0541
22e-03
          V12             V13
-1.487303e-03 -9.553873e-04

set.seed(1)
> train=sample (1: nrow(x), nrow(x)/2)
> test=(- train )
> y.test=y[test]
ridge.mod =glmnet (x[train ,],y[train],alpha =0, lambda =grid ,
+                     thresh =1e-12)
ridge.pred=predict (ridge.mod ,s=4, newx=x[test ,])
> mean(( ridge.pred -y.test)^2)
[1] 0.1745175
```

MSE is 0.1745

```
mean(( mean(y[train ])-y.test)^2)
[1] 1.086542


> ridge.pred=predict (ridge.mod ,s=1e10 ,newx=x[test ,])
> mean(( ridge.pred -y.test)^2)
```

```
[1] 1.086542


> ridge.pred=predict (ridge.mod ,s=0, newx=x[test ,], exact=T)
> mean(( ridge.pred -y.test)^2)
[1] 0.05601155
> lm(y~x, subset =train)

Call:
lm(formula = y ~ x, subset = train)

Coefficients:
(Intercept)          xV1          xV2          xV3          xV4          xV5
  7.506e+01    3.913e-04   -6.025e-03   -1.111e-02   -8.117e-02    4.008e-01
        xV6          xV7          xV8          xV9         xV10         xV11
  7.283e-04   -4.746e-08   -2.245e-01   -7.642e-02   -1.106e+00    1.256e-03
       xV12         xV13
 -4.611e-02    2.861e-02
> predict (ridge.mod ,s=0, exact =T,type="coefficients") [1:14 ,]
  (Intercept)            V1            V2            V3            V4
V5
 7.505955e+01  3.913258e-04 -6.024746e-03 -1.111252e-02 -8.117081e-02  4.0074
47e-01
           V6            V7            V8            V9           V10
V11
 7.283714e-04 -4.745528e-08 -2.245362e-01 -7.642078e-02 -1.105755e+00  1.2554
22e-03
          V12           V13
-4.610965e-02  2.861557e-02


> set.seed (1)
> cv.out =cv.glmnet (x[train ,],y[train],alpha =0)
> plot(cv.out)
```




```
> bestlam =cv.out$lambda.min
> bestlam
[1] 0.1317802
```

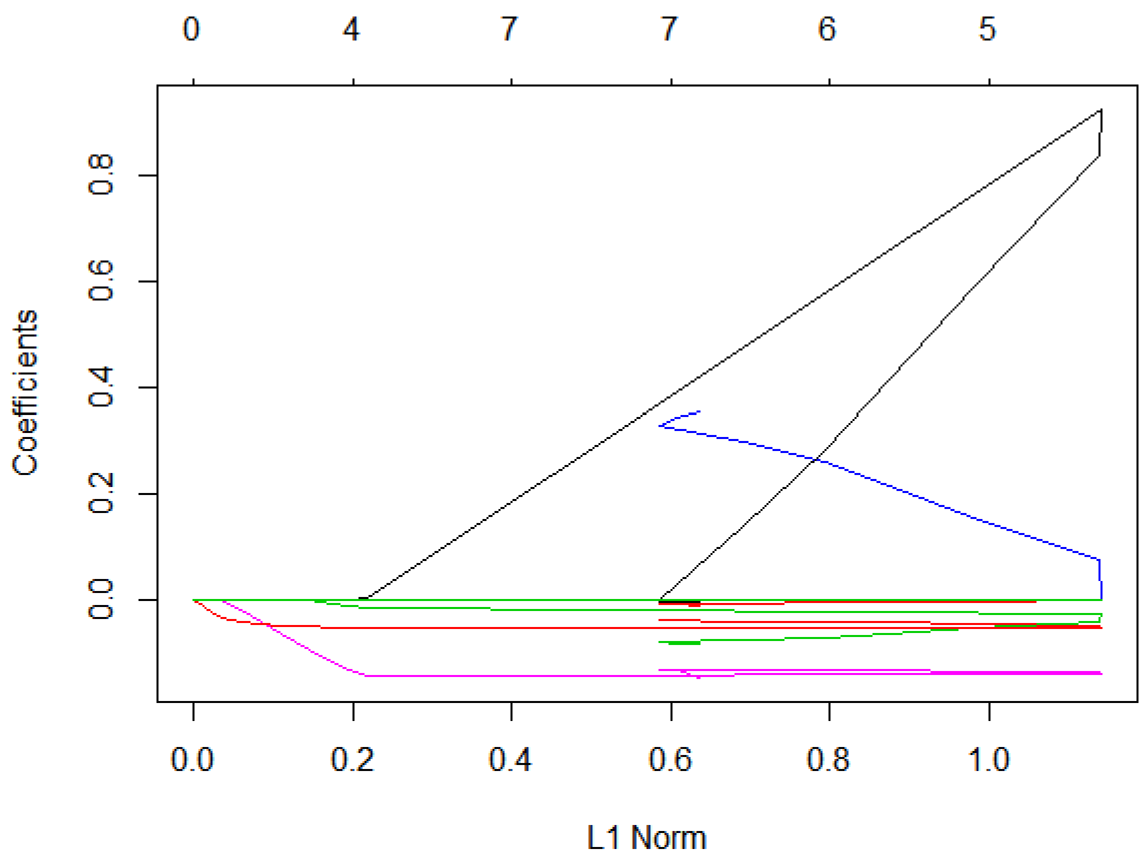
```
ridge.pred=predict (ridge.mod ,s=bestlam ,newx=x[test ,])
> mean(( ridge.pred -y.test)^2)
[1] 0.04282599
```

There is an improvement in the MSE

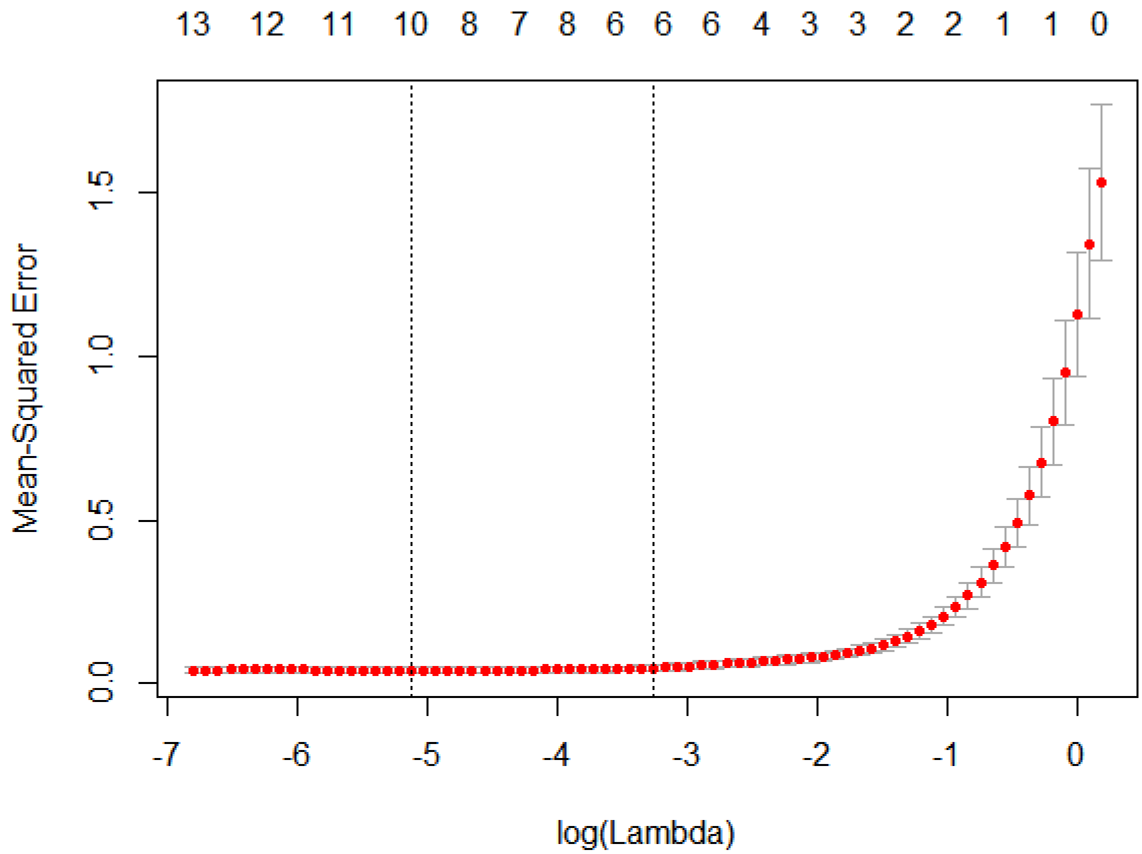
```
> out=glmnet (x,y,alpha =0)
> predict (out ,type="coefficients",s=bestlam )[1:14 ,]
  (Intercept)            V1            V2            V3            V4
V5
 7.016981e+01 -2.894303e-04 -1.042156e-02 -1.622324e-02 -7.532142e-02  2.6534
45e-01
           V6            V7            V8            V9           V10
V11
-5.848681e-03 -2.211937e-07 -1.243824e-01  1.620534e-02  3.278224e+00 -9.3285
98e-03
          V12           V13
-2.114120e-02 -1.256978e-03
```

The Lasso Regression

```
> lasso.mod =glmnet (x[train ,],y[train],alpha =1, lambda =grid)
> plot(lasso.mod)
```

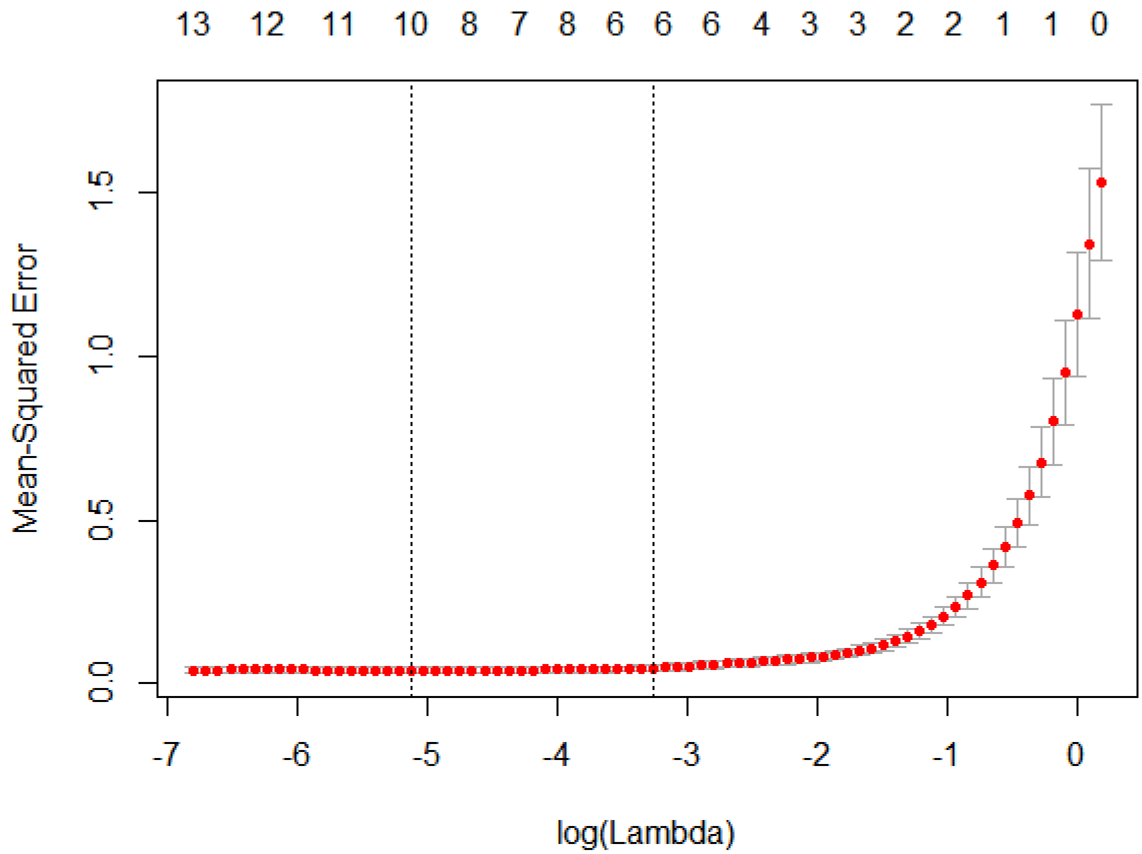



```
> set.seed (1)
> cv.out =cv.glmnet (x[train ,],y[train],alpha =1)
> plot(cv.out)
```




```
> bestlam =cv.out$lambda.min
> bestlam
[1] 0.005976074

> lasso.pred=predict (lasso.mod ,s=bestlam ,newx=x[test ,])
> mean(( lasso.pred -y.test)^2)
[1] 0.04374007
```

This MSE is .001 more than ridge




```
out=glmnet (x,y,alpha =1, lambda =grid)
> lasso.coef=predict(out ,type ="coefficients",s=bestlam )[1:14 ,]
> lasso.coef
  (Intercept)            V1            V2            V3            V4            
V5
 6.948794e+01 -3.036679e-05 -6.106838e-03 -8.053118e-03 -9.695680e-02  3.9741
24e-01
           V6            V7            V8            V9           V10           
V11
-4.303790e-03 -8.161627e-08 -7.266772e-02  4.534686e-03  1.278453e+00  0.0000
00e+00
          V12           V13
-3.164070e-02  0.000000e+00
```

Communities and Crime:

The data set caontains 122 predictors. The response variable is "ViolentCrimesPerPop". The cells with "?" were removed using the following code.

crimeData=read.csv("C:\\Users\\Sushmitha\\Documents\\Third Semester\\Comp Mthds Data Science\\Assignment VI\\Assign_6-Data\\Assign_6-Data\\communitiesCrimeDataset.csv",",", header= TRUE)

idx <- crimeData == "?"

is.na(crimeData) <- idx

crimeData=na.omit(crimeData)

The final data set had 319 rows.

The Best selection model could not be executed due to higher count of predictors. So following are the models built using the "forward" and "backward" models.

#Forward Selection

regfit.fwd=regsubsets(ViolentCrimesPerPop~., crimeData, nvmax = 55, method = "forward")

head(summary(regfit.fwd))

$rsq

```
 [1] 0.5759351 0.6251904 0.6521741 0.6665235 0.6804758 0.6924569 0.7037983 0.7146353
 [9] 0.7272709 0.7378819 0.7460173 0.7532894 0.7596104 0.7651628 0.7705080 0.7758095
[17] 0.7811379 0.7861956 0.7913152 0.7964764 0.8018642 0.8062093 0.8113085 0.8171155
[25] 0.8226798 0.8269001 0.8314047 0.8350404 0.8382924 0.8414197 0.8444722 0.8474111
[33] 0.8502837 0.8533304 0.8562424 0.8592216 0.8622088 0.8646632 0.8674861 0.8700544
[41] 0.8724990 0.8747540 0.8770173 0.8790561 0.8811733 0.8831980 0.8851595 0.8871262
[49] 0.8891868 0.8915737 0.8938489 0.8961034 0.8981283 0.9002118 0.9024843 0.9044487
```

$rss

```
 [1] 10.298648  9.102456  8.447142  8.098659  7.759820  7.468852  7.193421  6.930239
 [9]  6.623375  6.365680  6.168108  5.991503  5.837993  5.703149  5.573338  5.444588
[17]  5.315186  5.192356  5.068023  4.942681  4.811837  4.706314  4.582477  4.441450
[25]  4.306319  4.203826  4.094428  4.006135  3.927158  3.851210  3.777078  3.705704
```

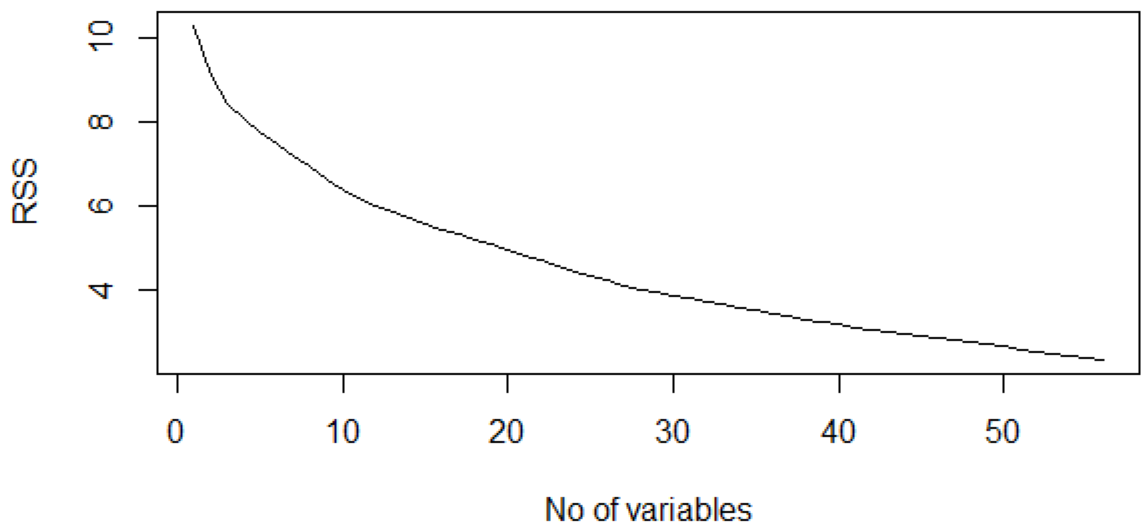
```
[33]  3.635943  3.561952  3.491232  3.418880  3.346335  3.286729  3.218172  3.155800
[41]  3.096433  3.041668  2.986702  2.937190  2.885772  2.836600  2.788965  2.741203
[49]  2.691158  2.633193  2.577938  2.523186  2.474010  2.423412  2.368221  2.320516

$adjr2
 [1] 0.5745974 0.6228182 0.6488615 0.6622754 0.6753716 0.6865426 0.6971314 0.7072710
 [9] 0.7193274 0.7293716 0.7369170 0.7436144 0.7493643 0.7543479 0.7591471 0.7639319
[17] 0.7687769 0.7733674 0.7780543 0.7828171 0.7878546 0.7918059 0.7965969 0.8021862
[25] 0.8075500 0.8114871 0.8157619 0.8191132 0.8220656 0.8249009 0.8276730 0.8303382
[33] 0.8329481 0.8357713 0.8384632 0.8412499 0.8440655 0.8462960 0.8489627 0.8513572
[41] 0.8536270 0.8556948 0.8577873 0.8596344 0.8615864 0.8634447 0.8652425 0.8670597
[49] 0.8690015 0.8713449 0.8735728 0.8757928 0.8777539 0.8798005 0.8820913 0.8840255

$cp
 [1] -14708.212 -13034.435 -12116.579 -11627.546 -11151.989 -10743.338 -10356.400
 [8]  -9986.582  -9555.714  -9193.565  -8915.441  -8666.621  -8450.078  -8259.623
[15]  -8076.200  -7894.262  -7711.412  -7537.747  -7361.982  -7184.806  -6999.940
[22]  -6850.463  -6675.391  -6476.294  -6285.437  -6140.194  -5985.302  -5859.904
[29]  -5747.529  -5639.385  -5533.779  -5432.028  -5332.531  -5227.123  -5126.285
[36]  -5023.167  -4919.779  -4834.476  -4736.662  -4647.492  -4562.521  -4483.983
[43]  -4405.163  -4333.965  -4260.106  -4189.383  -4120.809  -4052.058  -3980.116
[50]  -3897.105  -3817.881  -3739.360  -3668.634  -3595.918  -3516.785  -3448.113

$bic
 [1] -262.1298 -295.7510 -313.8202 -321.4944 -329.3631 -335.7894 -342.0104 -348.1352
 [9] -356.8173 -363.7112 -368.0038 -371.5055 -374.0200 -375.7094 -377.2890 -378.9795
[17] -380.8875 -382.5807 -384.5470 -386.7705 -389.5638 -390.8721 -393.6131 -397.8195
[25] -401.9106 -403.8296 -406.4758 -407.6649 -408.2512 -408.7156 -409.1507 -409.4713
[33] -409.7686 -410.5620 -411.1941 -412.1093 -413.1858 -413.1539 -414.1130 -414.5911
```

[41] -414.8842 -414.8114 -414.8636 -414.4310 -414.2995 -414.0168 -413.6541 -413.3992

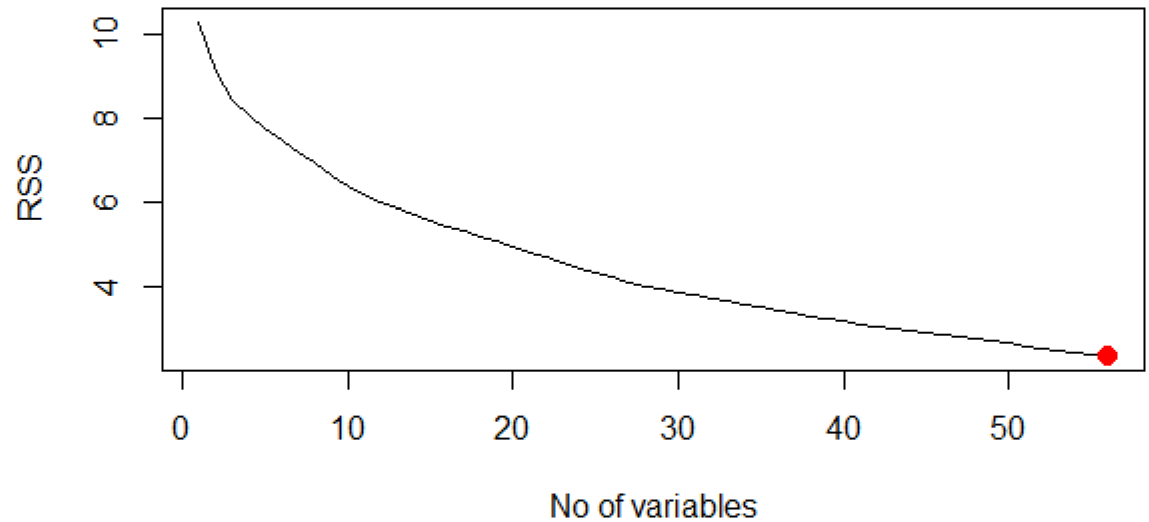
[49] -413.5116 -414.6926 -415.6925 -416.7755 -417.2887 -418.1154 -419.6991 -420.4254

The file output_ForwardSelection.txt attached with the assignment shows the summary of the forward model.

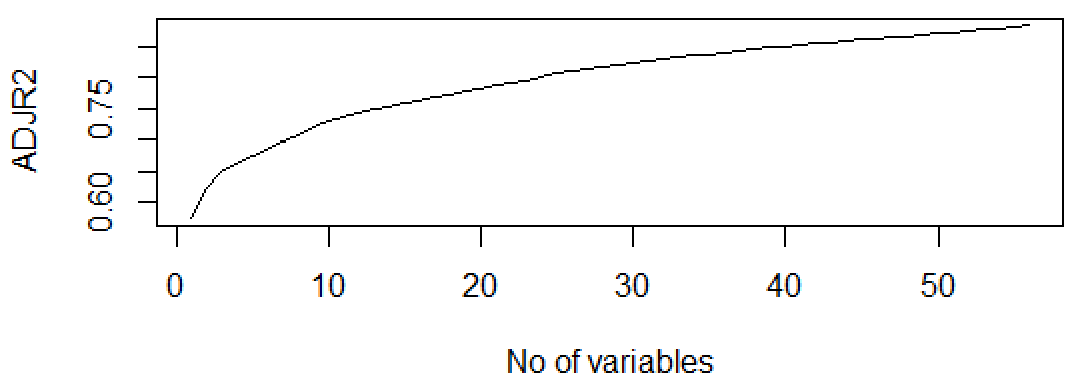
plot(regfit.sum_fwd$rss, xlab = "No of variables", ylab = "RSS" , type = "l")



points(56,regfit.sum_fwd$rss[56], col = "red", cex=2, pch=20)



plot(regfit.sum_fwd$adjr2, xlab = "No of variables", ylab = "ADJR2" , type = "l")



From the above plots we can infer that the model with nvmax=55, the adjusted R square with one predictor gives the best model. And the RSS and adjusted R square are inversely proportional.

#Backward Selection

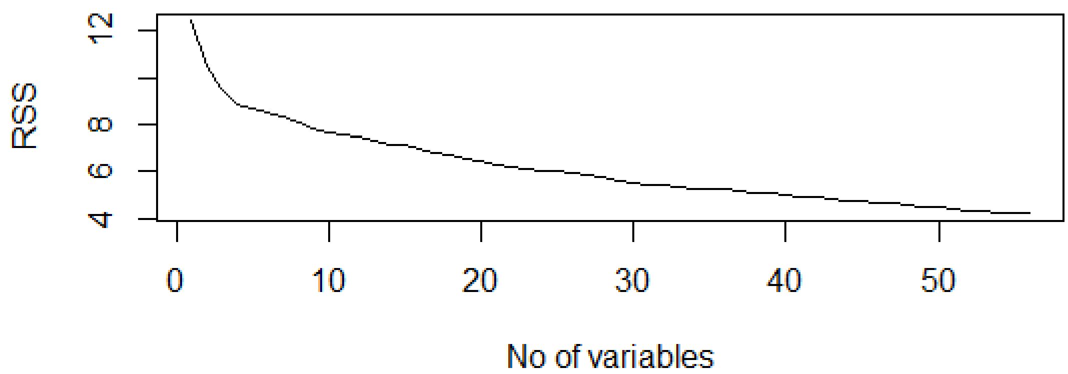
```
regfit.bwd=regsubsets(ViolentCrimesPerPop~., crimeData, nvmax = 55, method = "backward")
sink(file="output_BackwardSelection.txt")
regfit.sum_bwd=summary(regfit.bwd)
head(summary(regfit.bwd))
```

```
$rsq
 [1] 0.4885507 0.5675394 0.6077228 0.6344460 0.6416150 0.6493676 0.6575348 0.6681325
 [9] 0.6772358 0.6849204 0.6898604 0.6903921 0.6976386 0.7057232 0.7067725 0.7140608
[17] 0.7191327 0.7242183 0.7298332 0.7352423 0.7403849 0.7453741 0.7490500 0.7527328
[25] 0.7531287 0.7555004 0.7595096 0.7639418 0.7682604 0.7722317 0.7746182 0.7780789
[33] 0.7799249 0.7830703 0.7835444 0.7847145 0.7872475 0.7899424 0.7913577 0.7941932
[41] 0.7968644 0.7992922 0.8010438 0.8031356 0.8046045 0.8065973 0.8089763 0.8118458
[49] 0.8144644 0.8166573 0.8185472 0.8207094 0.8225997 0.8243216 0.8262531 0.8269956

$rss
 [1] 12.420826 10.502543  9.526666  8.877679  8.703576  8.515300  8.316954  8.059584
 [9]  7.838505  7.651881  7.531911  7.518997  7.343012  7.146674  7.121190  6.944190
[17]  6.821016  6.697508  6.561149  6.429785  6.304894  6.183728  6.094458  6.005019
[25]  5.995406  5.937807  5.840440  5.732803  5.627922  5.531479  5.473522  5.389475
[33]  5.344644  5.268257  5.256742  5.228326  5.166810  5.101364  5.066992  4.998131
[41]  4.933260  4.874299  4.831759  4.780959  4.745287  4.696890  4.639115  4.569427
[49]  4.505834  4.452579  4.406682  4.354170  4.308263  4.266447  4.219538  4.201507

$adjr2
```

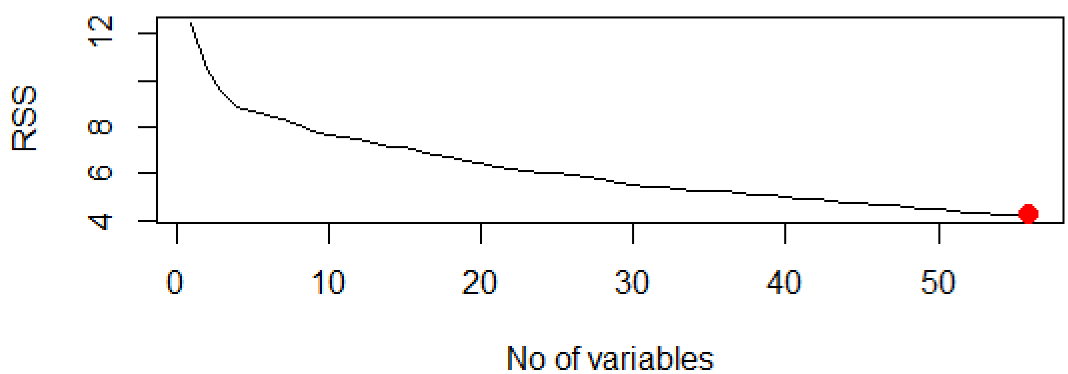
```
 [1] 0.4869373 0.5648023 0.6039868 0.6297893 0.6358900 0.6426246 0.6498266 0.6595682
 [9] 0.6678349 0.6746905 0.6787479 0.6782506 0.6847511 0.6921710 0.6922563 0.6989117
[17] 0.7032698 0.7076714 0.7126654 0.7174734 0.7220283 0.7264492 0.7294844 0.7325478
[25] 0.7320646 0.7337299 0.7371961 0.7411500 0.7450063 0.7485058 0.7502738 0.7532486
[33] 0.7544426 0.7570998 0.7567743 0.7572312 0.7592339 0.7614346 0.7621927 0.7645807
[41] 0.7667973 0.7687497 0.7699343 0.7715224 0.7723965 0.7738895 0.7758467 0.7783962
[49] 0.7806679 0.7824515 0.7838876 0.7856601 0.7871197 0.7883874 0.7899182 0.7900176

$cp
 [1] -17674.131 -14991.173 -13625.303 -12716.290 -12470.966 -12205.836 -11926.631
 [8] -11564.934 -11253.959 -10991.136 -10821.468 -10801.420 -10553.466 -10277.067
[15] -10239.452  -9990.080  -9815.933  -9641.322  -9448.748  -9263.156  -9086.610
[22]  -8915.271  -8788.509  -8661.510  -8646.075  -8563.576  -8425.498  -8273.065
[29]  -8124.487  -7987.699  -7904.699  -7785.236  -7720.581  -7611.824  -7593.731
[36]  -7552.018  -7464.044  -7370.578  -7320.541  -7222.302  -7129.639  -7045.236
[43]  -6983.783  -6910.785  -6858.930  -6789.292  -6706.547  -6607.152  -6516.275
[50]  -6439.847  -6373.702  -6298.313  -6232.153  -6171.712  -6104.153  -6076.953

$bic
 [1] -202.3613 -250.1106 -275.4552 -292.1969 -292.7499 -293.9610 -295.7142 -299.9765
 [9] -303.0839 -305.0055 -304.2814 -299.0636 -300.8535 -303.7339 -299.1082 -301.3721
[17] -301.3161 -301.3799 -302.1765 -302.8630 -303.3549 -303.7799 -302.6534 -301.6044
[25] -296.3503 -293.6646 -293.1736 -293.3424 -293.4673 -293.2160 -290.8109 -289.9820
[33] -286.8814 -285.7083 -280.6412 -276.6050 -274.6154 -272.9167 -269.3081 -267.9079
[41] -266.3101 -264.3805 -261.4116 -259.0180 -255.6420 -253.1469 -251.3299 -250.3931
[49] -249.0987 -247.1262 -244.6663 -242.7253 -240.3413 -237.6874 -235.4490 -231.0499
```

The file output_ForwardSelection.txt attached with the assignment shows the summary of the forward model.

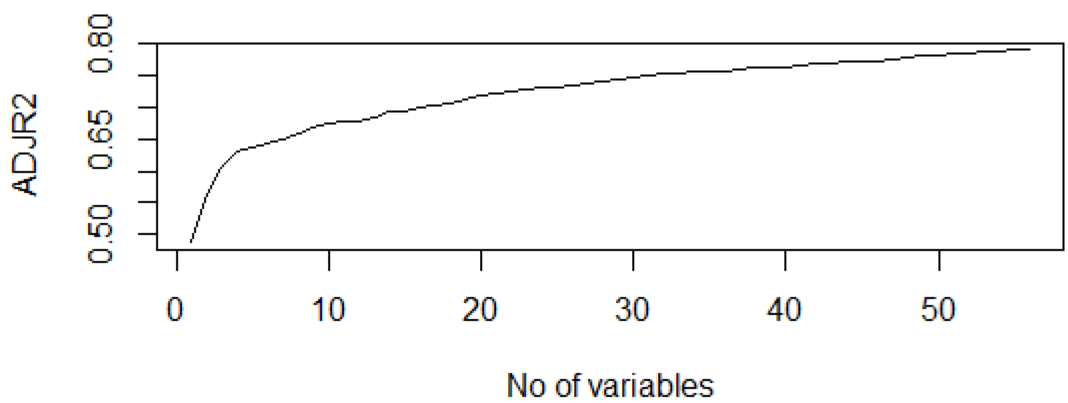
plot(regfit.sum_bwd$rss, xlab = "No of variables", ylab = "RSS" , type = "l")



points(56,regfit.sum_bwd$rss[56], col = "red", cex=2, pch=20)



plot(regfit.sum_bwd$adjr2, xlab = "No of variables", ylab = "ADJR2" , type = "l")



From the above plot we can infer that the model with nvmax=55, the adjusted R square with one predictor gives the best model. And the RSS and adjusted R square are inversely proportional.