회귀분석2를 위한 보충 R 코드와 결과

서울대학교 통계학과

August 16, 2018

Contents

1 전역옵션들 **1**

2 최적변수선택 **1**

3 전진선택법, 후진선택법 **9**

4 검증자료 방법을 이용한 변수선택 **14**

5 K 겹 교차검증을 이용한 변수 선택 **15**

6 능선회귀와 라쏘 **17**

6.1 능선회귀 17

6.2 라쏘 22

1 전역옵션들

```
opts_chunk$set(eval=TRUE, cache=TRUE, fig.width=7, fig.height=4)
```

2 최적변수선택

```r
library(ISLR)
#fix(Hitters)
names(Hitters)
```

```
##  [1] "AtBat"     "Hits"      "HmRun"     "Runs"      "RBI"
##  [6] "Walks"     "Years"     "CAtBat"    "CHits"     "CHmRun"
## [11] "CRuns"     "CRBI"      "CWalks"    "League"    "Division"
## [16] "PutOuts"   "Assists"   "Errors"    "Salary"    "NewLeague"
```

```r
dim(Hitters)
```

```
## [1] 322  20
```

```r
str(Hitters)
```

```
## 'data.frame': 322 obs. of  20 variables:
##  $ AtBat    : int  293 315 479 496 321 594 185 298 323 401 ...
##  $ Hits     : int  66 81 130 141 87 169 37 73 81 92 ...
##  $ HmRun    : int  1 7 18 20 10 4 1 0 6 17 ...
##  $ Runs     : int  30 24 66 65 39 74 23 24 26 49 ...
##  $ RBI      : int  29 38 72 78 42 51 8 24 32 66 ...
##  $ Walks    : int  14 39 76 37 30 35 21 7 8 65 ...
##  $ Years    : int  1 14 3 11 2 11 2 3 2 13 ...
##  $ CAtBat   : int  293 3449 1624 5628 396 4408 214 509 341 5206 ...
##  $ CHits    : int  66 835 457 1575 101 1133 42 108 86 1332 ...
##  $ CHmRun   : int  1 69 63 225 12 19 1 0 6 253 ...
##  $ CRuns    : int  30 321 224 828 48 501 30 41 32 784 ...
##  $ CRBI     : int  29 414 266 838 46 336 9 37 34 890 ...
##  $ CWalks   : int  14 375 263 354 33 194 24 12 8 866 ...
##  $ League   : Factor w/ 2 levels "A","N": 1 2 1 2 2 1 2 1 2 1 ...
##  $ Division : Factor w/ 2 levels "E","W": 1 2 2 1 1 2 1 2 2 1 ...
##  $ PutOuts  : int  446 632 880 200 805 282 76 121 143 0 ...
##  $ Assists  : int  33 43 82 11 40 421 127 283 290 0 ...
##  $ Errors   : int  20 10 14 3 4 25 7 9 19 0 ...
##  $ Salary   : num  NA 475 480 500 91.5 750 70 100 75 1100 ...
##  $ NewLeague: Factor w/ 2 levels "A","N": 1 2 1 2 2 1 1 1 2 1 ...
```

Hitters 자료를 살펴본다. str의 결과를 보면 Salary에 NA가 있다는 것을 알수있다.

```
sum(is.na(Hitters$Salary))

## [1] 59

Hitters=na.omit(Hitters)
dim(Hitters)

## [1] 263  20

sum(is.na(Hitters))

## [1] 0
```

NA를 다 제거한다.

```
library(leaps)
regfit.full=regsubsets(Salary~., Hitters)
summary(regfit.full)

## Subset selection object
## Call: regsubsets.formula(Salary ~ ., Hitters)
## 19 Variables  (and intercept)
##            Forced in Forced out
## AtBat          FALSE      FALSE
## Hits           FALSE      FALSE
## HmRun          FALSE      FALSE
## Runs           FALSE      FALSE
## RBI            FALSE      FALSE
## Walks          FALSE      FALSE
## Years          FALSE      FALSE
## CAtBat         FALSE      FALSE
## CHits          FALSE      FALSE
## CHmRun         FALSE      FALSE
## CRuns          FALSE      FALSE
## CRBI           FALSE      FALSE
## CWalks         FALSE      FALSE
## LeagueN        FALSE      FALSE
```

```
## DivisionW      FALSE      FALSE
## PutOuts        FALSE      FALSE
## Assists        FALSE      FALSE
## Errors         FALSE      FALSE
## NewLeagueN     FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: exhaustive
##          AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun CRuns
## 1  ( 1 ) " "   " "  " "   " "  " " " " " "   " "   " "   " "    " "    " "
## 2  ( 1 ) " "   "*"  " "   " "  " " " " " "   " "   " "   " "    " "    " "
## 3  ( 1 ) " "   "*"  " "   " "  " " " " " "   " "   " "   " "    " "    " "
## 4  ( 1 ) " "   "*"  " "   " "  " " " " " "   " "   " "   " "    " "    " "
## 5  ( 1 ) "*"   "*"  " "   " "  " " " " " "   " "   " "   " "    " "    " "
## 6  ( 1 ) "*"   "*"  " "   " "  " " " " "*"   " "   " "   " "    " "    " "
## 7  ( 1 ) " "   "*"  " "   " "  " " " " "*"   " "   "*"   "*"    "*"    " "
## 8  ( 1 ) "*"   "*"  " "   " "  " " " " "*"   " "   " "   " "    "*"    "*"
##          CRBI CWalks LeagueN DivisionW PutOuts Assists Errors NewLeagueN
## 1  ( 1 ) "*"  " "    " "     " "       " "     " "     " "    " "
## 2  ( 1 ) "*"  " "    " "     " "       " "     " "     " "    " "
## 3  ( 1 ) "*"  " "    " "     " "       "*"     " "     " "    " "
## 4  ( 1 ) "*"  " "    " "     "*"       "*"     " "     " "    " "
## 5  ( 1 ) "*"  " "    " "     "*"       "*"     " "     " "    " "
## 6  ( 1 ) "*"  " "    " "     "*"       "*"     " "     " "    " "
## 7  ( 1 ) " "  " "    " "     "*"       "*"     " "     " "    " "
## 8  ( 1 ) " "  "*"    " "     "*"       "*"     " "     " "    " "
```

regsubsets는 변수의 개수에 따른 최적의 모형을 반환한다. regsubsets의 옵션 중 force.in과 force.out은 반드시 모형에 들어가야하는 혹은 빠져야 하는 변수들의 인덱스를 지정한다. summary에서 모형의 크기가 8개까지인 것만 보여주는데 이것을 바꾸려면 nvmax 옵션을 쓰면 된다.

```
regfit.full=regsubsets(Salary~.,data=Hitters,nvmax=19)
reg.summary=summary(regfit.full)
names(reg.summary)

## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"

reg.summary$rsq
```

```
##  [1] 0.3215 0.4252 0.4514 0.4754 0.4908 0.5087 0.5141 0.5286 0.5346 0.5405
## [11] 0.5426 0.5436 0.5445 0.5452 0.5455 0.5458 0.5460 0.5461 0.5461
```

변수의 개수가 19개까지 최적의 모형을 반환한다. reg.summary$rsq는 변수의 개수에 따른 최적의 모형의 R^2값을 갖고 있다.

```
par(mfrow=c(2,2))
plot(reg.summary$rss,xlab="Number of Variables",ylab="RSS",type="l")
plot(reg.summary$adjr2,xlab="Number of Variables",ylab="Adjusted RSq",type="l")
which.max(reg.summary$adjr2)
```

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

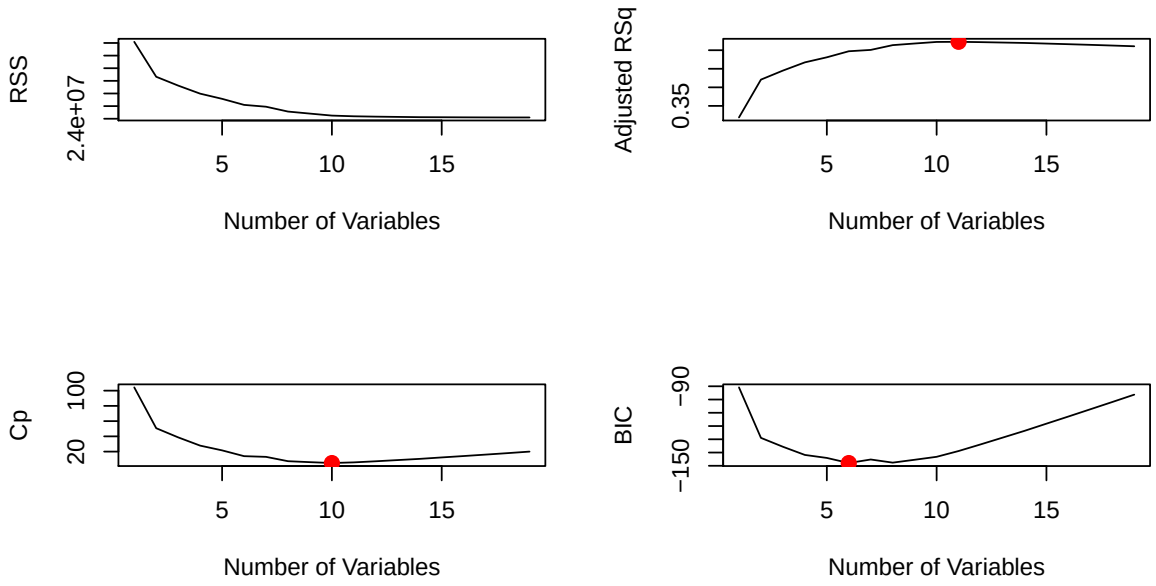
```
points(11,reg.summary$adjr2[11], col="red",cex=2,pch=20)
plot(reg.summary$cp,xlab="Number of Variables",ylab="Cp",type='l')
which.min(reg.summary$cp)
```

```
## [1] 10
```

```
points(10,reg.summary$cp[10],col="red",cex=2,pch=20)
which.min(reg.summary$bic)
```

```
## [1] 6
```

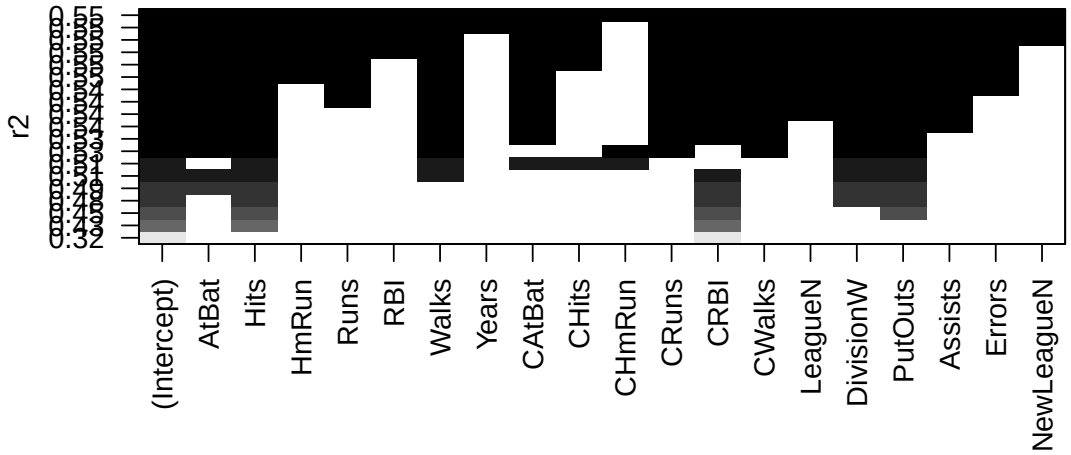
```
plot(reg.summary$bic,xlab="Number of Variables",ylab="BIC",type='l')
points(6,reg.summary$bic[6],col="red",cex=2,pch=20)
```



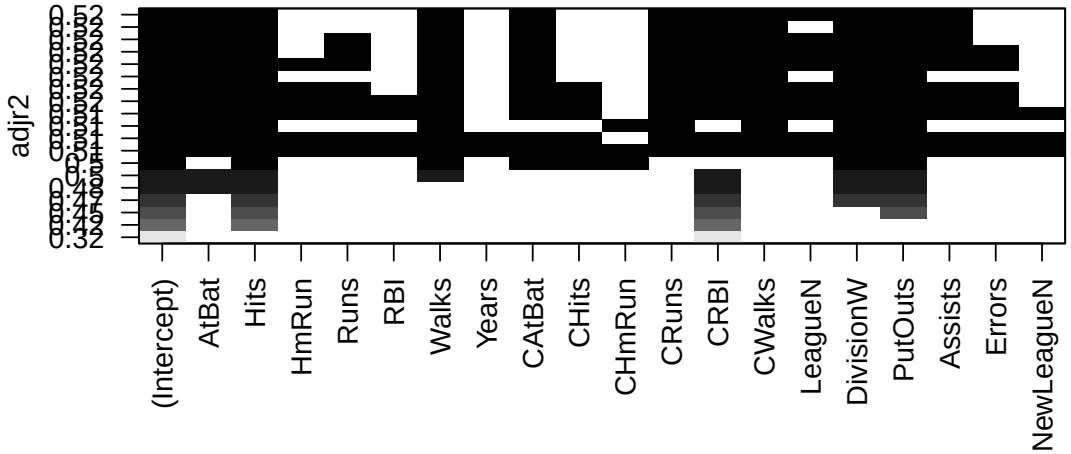

```
par(mfrow=c(1,1))
```

변수의 개수에 따른 최적 모형의 RSS, adjusted R^2, Cp, BIC 값을 그림으로 그리고 최적의 모형을 표시했다.

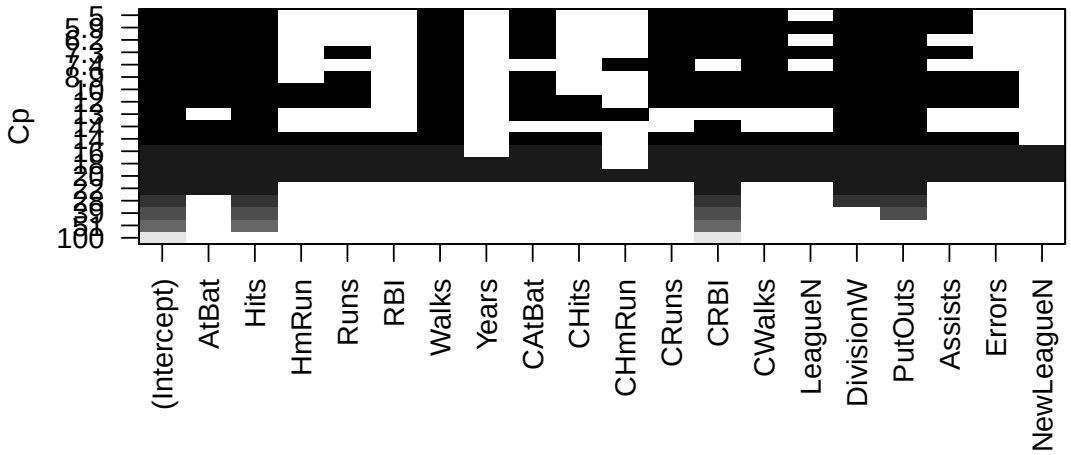
```
plot(regfit.full,scale="r2")
```



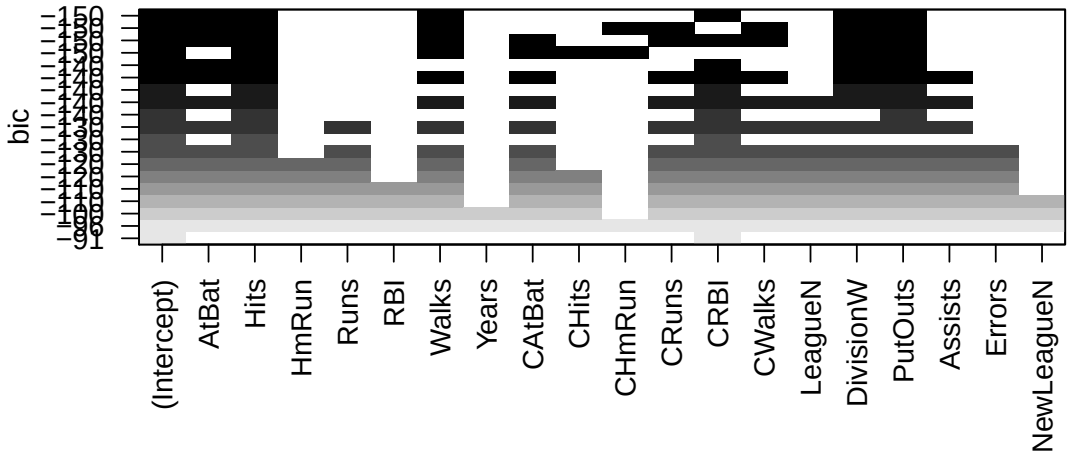

```
plot(regfit.full,scale="adjr2")
```




```
plot(regfit.full,scale="Cp")
```




```
plot(regfit.full,scale="bic")
```




```
coef(regfit.full,6)

## (Intercept)       AtBat        Hits       Walks        CRBI   DivisionW
```

```
##     91.5118     -1.8686      7.6044      3.6976      0.6430   -122.9515
##     PutOuts
##      0.2643
```

이 그림들은 각 기준에 따라 들어가는 변수들을 검은색 박스로 표현했다. 선택되지 않은 변수는 흰색박스로 표시된다. bic의 기준으로 최적의 모형은 6개의 변수(절편을 포함하면 7개)를 포함하는 모형이다. 이 모형의 회귀계수를 알아보기 위해 마지막 명령어를 썼다.

3 전진선택법, 후진선택법

```
regfit.fwd=regsubsets(Salary~.,data=Hitters,nvmax=19,method="forward")
summary(regfit.fwd)

## Subset selection object
## Call: regsubsets.formula(Salary ~ ., data = Hitters, nvmax = 19, method = "forward")
## 19 Variables  (and intercept)
##            Forced in Forced out
## AtBat          FALSE      FALSE
## Hits           FALSE      FALSE
## HmRun          FALSE      FALSE
## Runs           FALSE      FALSE
## RBI            FALSE      FALSE
## Walks          FALSE      FALSE
## Years          FALSE      FALSE
## CAtBat         FALSE      FALSE
## CHits          FALSE      FALSE
## CHmRun         FALSE      FALSE
## CRuns          FALSE      FALSE
## CRBI           FALSE      FALSE
## CWalks         FALSE      FALSE
## LeagueN        FALSE      FALSE
## DivisionW      FALSE      FALSE
## PutOuts        FALSE      FALSE
## Assists        FALSE      FALSE
## Errors         FALSE      FALSE
```

```
## NewLeagueN     FALSE      FALSE
## 1 subsets of each size up to 19
## Selection Algorithm: forward
##           AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun CRuns
## 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 ) "*"   "*"  " "   "*"  " " " " "*"   " "   "*"    " "   " "    "*"
## 14  ( 1 ) "*"   "*"  "*"   "*"  " " " " "*"   " "   "*"    " "   " "    "*"
## 15  ( 1 ) "*"   "*"  "*"   "*"  " " " " "*"   " "   "*"    "*"   " "    "*"
## 16  ( 1 ) "*"   "*"  "*"   "*"  "*" "*" "*"   " "   "*"    "*"   " "    "*"
## 17  ( 1 ) "*"   "*"  "*"   "*"  "*" "*" "*"   " "   "*"    "*"   " "    "*"
## 18  ( 1 ) "*"   "*"  "*"   "*"  "*" "*" "*"   "*"   "*"    "*"   " "    "*"
## 19  ( 1 ) "*"   "*"  "*"   "*"  "*" "*" "*"   "*"   "*"    "*"   "*"    "*"
##           CRBI CWalks LeagueN DivisionW PutOuts Assists Errors NewLeagueN
## 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 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    " "
## 14  ( 1 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    " "
## 15  ( 1 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    " "
## 16  ( 1 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    " "
## 17  ( 1 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    "*"
## 18  ( 1 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    "*"
## 19  ( 1 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    "*"
```

전진선택법을 이용할 때 선택되는 변수들을 *로 표시한다.

```
regfit.bwd=regsubsets(Salary~.,data=Hitters,nvmax=19,method="backward")
summary(regfit.bwd)

## Subset selection object
## Call: regsubsets.formula(Salary ~ ., data = Hitters, nvmax = 19, method = "backward")
## 19 Variables  (and intercept)
##            Forced in Forced out
## AtBat          FALSE      FALSE
## Hits           FALSE      FALSE
## HmRun          FALSE      FALSE
## Runs           FALSE      FALSE
## RBI            FALSE      FALSE
## Walks          FALSE      FALSE
## Years          FALSE      FALSE
## CAtBat         FALSE      FALSE
## CHits          FALSE      FALSE
## CHmRun         FALSE      FALSE
## CRuns          FALSE      FALSE
## CRBI           FALSE      FALSE
## CWalks         FALSE      FALSE
## LeagueN        FALSE      FALSE
## DivisionW      FALSE      FALSE
## PutOuts        FALSE      FALSE
## Assists        FALSE      FALSE
```

```
## Errors         FALSE      FALSE
## NewLeagueN     FALSE      FALSE
## 1 subsets of each size up to 19
## Selection Algorithm: backward
##           AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun CRuns
## 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 ) "*"   "*"  " "   "*"  " " " "*"   " "   "*"    " "   " "    "*"
## 14  ( 1 ) "*"   "*"  "*"   "*"  " " " "*"   " "   "*"    " "   " "    "*"
## 15  ( 1 ) "*"   "*"  "*"   "*"  " " " "*"   " "   "*"    "*"   " "    "*"
## 16  ( 1 ) "*"   "*"  "*"   "*"  "*" "*"   " "   "*"    "*"   " "    "*"
## 17  ( 1 ) "*"   "*"  "*"   "*"  "*" "*"   " "   "*"    "*"   " "    "*"
## 18  ( 1 ) "*"   "*"  "*"   "*"  "*" "*"   "*"   "*"    "*"   " "    "*"
## 19  ( 1 ) "*"   "*"  "*"   "*"  "*" "*"   "*"   "*"    "*"   "*"    "*"
##           CRBI CWalks LeagueN DivisionW PutOuts Assists Errors NewLeagueN
## 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 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    " "
## 14  ( 1 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    " "
## 15  ( 1 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    " "
## 16  ( 1 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    " "
## 17  ( 1 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    "*"
## 18  ( 1 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    "*"
## 19  ( 1 ) "*"  "*"    "*"     "*"       "*"     "*"     "*"    "*"
```

후진선택법으로 변수를 선택할 때의 결과를 보여준다.

```
coef(regfit.full,7)

## (Intercept)        Hits       Walks      CAtBat       CHits      CHmRun
##     79.4509      1.2834      3.2274     -0.3752      1.4957      1.4421
##   DivisionW     PutOuts
##   -129.9866      0.2367

coef(regfit.fwd,7)

## (Intercept)       AtBat        Hits       Walks        CRBI      CWalks
##    109.7873     -1.9589      7.4499      4.9131      0.8538     -0.3053
##   DivisionW     PutOuts
##   -127.1224      0.2533

coef(regfit.bwd,7)

## (Intercept)       AtBat        Hits       Walks       CRuns      CWalks
##    105.6487     -1.9763      6.7575      6.0559      1.1293     -0.7163
##   DivisionW     PutOuts
##   -116.1692      0.3029
```

all subset selection, 전진선택법, 후진선택법을 이용할 때 변수 7개의 최적 모형이 다 다르다. 변수 6개까지의 모형은 세 가지 방법이 모두 같다.

참고 책에서는 용어를 forward stepwise, backward stepwise라고 썼는데 옳지 않은 용어사용인 것 같다. stepwise라는 말이 들어가면 변수가 포함되었다가도 뺄 수 있어야하는데, 그런 것을 의미하지는 않는다.

4 검증자료 방법을 이용한 변수선택

```
set.seed(1)
train=sample(c(TRUE,FALSE), nrow(Hitters),rep=TRUE)
test=(!train)
```

주어진 자료를 훈련자료와 시험자료로 나눌수 있도록 인덱스 벡터들을 만들었다.

```
regfit.best=regsubsets(Salary~.,data=Hitters[train,],nvmax=19)
```

훈련자료를 이용해 regsubsets을 돌렸다.

```
test.mat=model.matrix(Salary~.,data=Hitters[test,])
```

model.matrix는 모형식(model formula)를 이용해 design matrix를 반환하는 함수이다. 디자인행렬을 만들었다.

```
val.errors=rep(NA,19)
for(i in 1:19){
   coefi=coef(regfit.best,id=i)
   pred=test.mat[,names(coefi)]%*%coefi
   val.errors[i]=mean((Hitters$Salary[test]-pred)^2)
}
val.errors
```

```
##  [1] 220968 169157 178518 163426 168418 171271 162377 157909 154056 148162
## [11] 151156 151742 152214 157359 158541 158743 159973 159860 160106
```

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

```
## [1] 10
```

```
coef(regfit.best,10)
```

```
## (Intercept)       AtBat        Hits       Walks      CAtBat       CHits
##    -80.2751     -1.4684      7.1625      3.6430     -0.1856      1.1053
##      CHmRun      CWalks     LeagueN   DivisionW     PutOuts
##      1.3845     -0.7483     84.5576    -53.0290      0.2382
```

변수의 개수별로 훈련자료로 구한 최적의 모형에 대한 시험오차를 구했다. 시험오차가 가장 작은 모형은 변수의 개수가 10개인 모형이다.

```
predict.regsubsets=function(object,newdata,id,...){
  form=as.formula(object$call[[2]])
  mat=model.matrix(form,newdata)
  coefi=coef(object,id=id)
  xvars=names(coefi)
  mat[,xvars]%*%coefi
  }
```

regsubsets 함수는 예측값을 구하는 기능이 없어서, regsubsets 객체, 새로운 자료, 변수의 개수를 받아들여 예측값을 구하는 함수를 만들었다.

```
regfit.best=regsubsets(Salary~.,data=Hitters,nvmax=19)
coef(regfit.best,10)
```

```
## (Intercept)       AtBat        Hits       Walks      CAtBat       CRuns
##    162.5354     -2.1687      6.9180      5.7732     -0.1301      1.4082
##        CRBI      CWalks   DivisionW     PutOuts     Assists
##      0.7743     -0.8308   -112.3801      0.2974      0.2832
```

10개의 변수를 갖는 모형이 최적이라는 것을 검증자료 방법으로 알았다. 마지막으로 자료 전체를 다 써서 최적모형을 구했다. 마지막에 자료 전체를 다 써야 정확한 회귀계수 추정량을 구할 수 있다.

5 K 겹 교차검증을 이용한 변수 선택

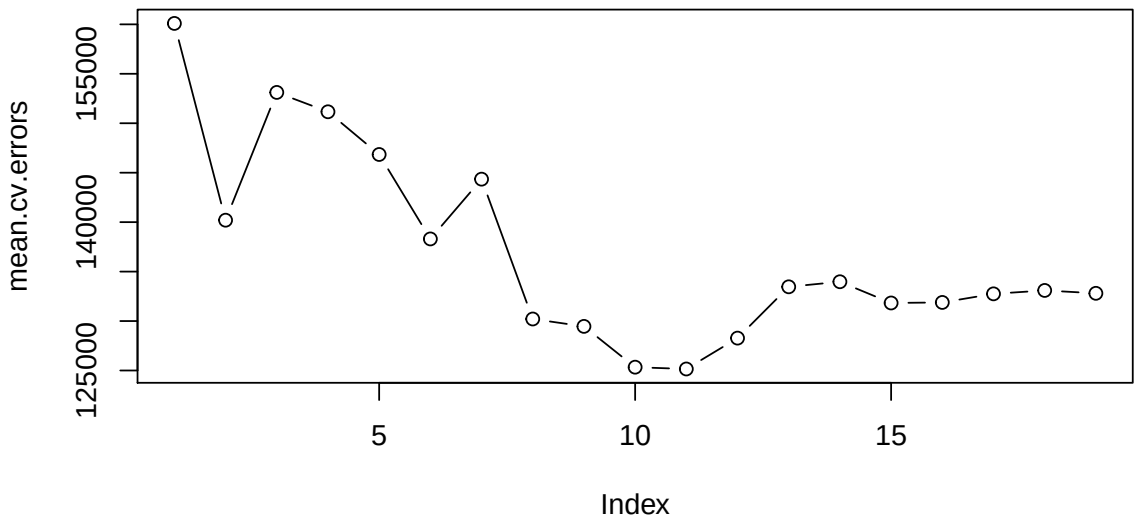
```
k=10
set.seed(1)
folds=sample(1:k,nrow(Hitters),replace=TRUE)
cv.errors=matrix(NA,k,19, dimnames=list(NULL, paste(1:19)))
```

k=10 겹 교차검증 방법을 이용한다. folds는 전체 자료의 인덱스를 1에서 k로 랜덤하게 할당한다. $k \times 19$ 행렬을 만들어 cv.errors라고 이름을 붙였다. 행은 겹을 나타내고 열은 각 겹에서 변수의 개수를 나타낸다. cv.errors[j,i]는 j번째 부분을 뺀 자료를 훈련자료로 모형을 적합했을 때, 변수의 개수가 i개인 모형 중 최적의 모형의 시험오차를 넣은 것이다.

```
for(j in 1:k){
  best.fit=regsubsets(Salary~.,data=Hitters[folds!=j,],nvmax=19)
  for(i in 1:19){
    pred=predict(best.fit,Hitters[folds==j,],id=i)
    cv.errors[j,i]=mean( (Hitters$Salary[folds==j]-pred)^2)
    }
  }
mean.cv.errors=apply(cv.errors,2,mean)
mean.cv.errors
```

```
##      1      2      3      4      5      6      7      8      9     10
## 160093 140197 153117 151159 146841 138303 144346 130208 129460 125335
##     11     12     13     14     15     16     17     18     19
## 125154 128274 133461 133975 131826 131883 132751 133096 132805
```

```
plot(mean.cv.errors,type='b')
```




```
which.min(mean.cv.errors)
```

```
## 11
## 11
```

cv.errors를 계산하고 변수의 개수에 따른 최적의 모형의 시험오차를 구한 값을 계산했다. 시험오차의 그림을 그렸다. 변수의 개수가 11개인 모형이 최적의 모형이다.

```
reg.best=regsubsets(Salary~.,data=Hitters, nvmax=19)
coef(reg.best,11)

## (Intercept)       AtBat        Hits       Walks      CAtBat       CRuns
##    135.7512     -2.1277      6.9237      5.6203     -0.1390      1.4553
##        CRBI      CWalks     LeagueN   DivisionW     PutOuts     Assists
##      0.7853     -0.8229     43.1116   -111.1460      0.2894      0.2688
```

전체 자료로 변수의 개수가 11인 모형을 구했다.

6 능선회귀와 라쏘

```
x=model.matrix(Salary~.,Hitters)[,-1]
y=Hitters$Salary
```

이 장에는 glmnet 패키지의 glmnet() 함수를 쓰는데, 이 함수는 모형식을 받아들이지 않고 설명변수와 반응변수를 행렬과 벡터로 대입해야 한다. model.matrix는 가변수도 자동적으로 생성해준다. 절편항은 디자인행렬에서 삭제했다.

6.1 능선회귀

```
library(glmnet)

## Loading required package: Matrix
## Loaded glmnet 1.9-8

grid=10^seq(10,-2,length=100)
ridge.mod=glmnet(x,y,alpha=0,lambda=grid)
```

glmnet은 GLM 모형에서 벌점가능도 추정치를 구하는 함수이다. y가 연속형 변수일 경우 옵션 family = gaussian이 디폴트이다. 벌점은

$$\frac{1-\alpha}{2}||\beta||_2^2 + \alpha||\beta||_1$$

와 같이 정의된다. alpha는 elastic net mixing 파라미터로 alpha = 0이면 능선회귀를 나타낸다. 벌점가능도는 가우시안 모형의 경우

$$\frac{1}{2}RSS/n + \lambda penalty$$

이고 다른 모형의 경우

$$-log - likelihood + \lambda penalty$$

이다. 옵션의 lambda는 위의 λ를 정할 때 쓰도록 주는 그리드 값이다. 위에서 lambda는 10^10에서 10^{-2}까지 변하고, 100개의 값을 갖는다. glmnet은 변수를 자동적으로 표준화한다.

```
dim(coef(ridge.mod))

## [1]  20 100
```

100개의 lambda값에 따른 추정된 회귀계수의 값이다. 20개의 row가 있는 이유는 19개의 변수와 절편항 때문이다.

```
ridge.mod$lambda[50]

## [1] 11498

coef(ridge.mod)[,50]

## (Intercept)       AtBat        Hits       HmRun        Runs         RBI
##  407.356050    0.036957    0.138180    0.524630    0.230702    0.239841
##       Walks       Years      CAtBat       CHits      CHmRun       CRuns
##    0.289619    1.107703    0.003132    0.011654    0.087546    0.023380
##        CRBI      CWalks     LeagueN   DivisionW     PutOuts     Assists
##    0.024138    0.025015    0.085028   -6.215441    0.016483    0.002613
##      Errors  NewLeagueN
##   -0.020503    0.301434

sqrt(sum(coef(ridge.mod)[-1,50]^2))

## [1] 6.361
```

50번째 lambda값과 그 lambda값에서 추정된 회귀계수, 회귀계수의 L_2 놈이다.

```
ridge.mod$lambda[60]

## [1] 705.5

coef(ridge.mod)[,60]

## (Intercept)       AtBat        Hits       HmRun        Runs         RBI
```

```
##    54.32520     0.11211     0.65622     1.17981     0.93770     0.84719
##       Walks       Years      CAtBat       CHits      CHmRun       CRuns
##     1.31988     2.59640     0.01083     0.04675     0.33777     0.09356
##        CRBI      CWalks     LeagueN   DivisionW     PutOuts     Assists
##     0.09780     0.07190    13.68370   -54.65878     0.11852     0.01606
##      Errors  NewLeagueN
##    -0.70359     8.61181
```

```
sqrt(sum(coef(ridge.mod)[-1,60]^2))
```

```
## [1] 57.11
```

60번째 lambda값과 그 lambda값에서 추정된 회귀계수, 회귀계수의 L_2 놈이다.

```
predict(ridge.mod,s=50,type="coefficients")[1:20,]
```

```
## (Intercept)       AtBat        Hits       HmRun        Runs         RBI
##   4.877e+01  -3.581e-01   1.969e+00  -1.278e+00   1.146e+00   8.038e-01
##       Walks       Years      CAtBat       CHits      CHmRun       CRuns
##   2.716e+00  -6.218e+00   5.448e-03   1.065e-01   6.245e-01   2.215e-01
##        CRBI      CWalks     LeagueN   DivisionW     PutOuts     Assists
##   2.187e-01  -1.500e-01   4.593e+01  -1.182e+02   2.502e-01   1.216e-01
##      Errors  NewLeagueN
##  -3.279e+00  -9.497e+00
```

predict함수는 여러 가지 목적으로 사용될 수 있다. s는 예측값을 구하는 lambda의 값을 지정하는 옵션이다. type="response"인 경우는 예측값(gaussian)이나 예측확률(binomial) 등을 준다. type="coefficient"인 경우는 추정된 회귀계수 값을 준다. 여기서는 lambda = 50에서의 회귀계수의 추정치를 리턴한다.

```
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] 101037
```

자료를 훈련자료와 시험자료로 나눈 후 lambda=4에서 시험오차를 구했다.

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

## [1] 193253
```

훈련자료를 예측치로 썼을 때 시험오차이다.

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

## [1] 193253
```

lambda = 10^10 일 때, 시험오차이다.

```
ridge.pred=predict(ridge.mod,s=0,newx=x[test,],exact=T)
mean((ridge.pred-y.test)^2)

## [1] 114783
```

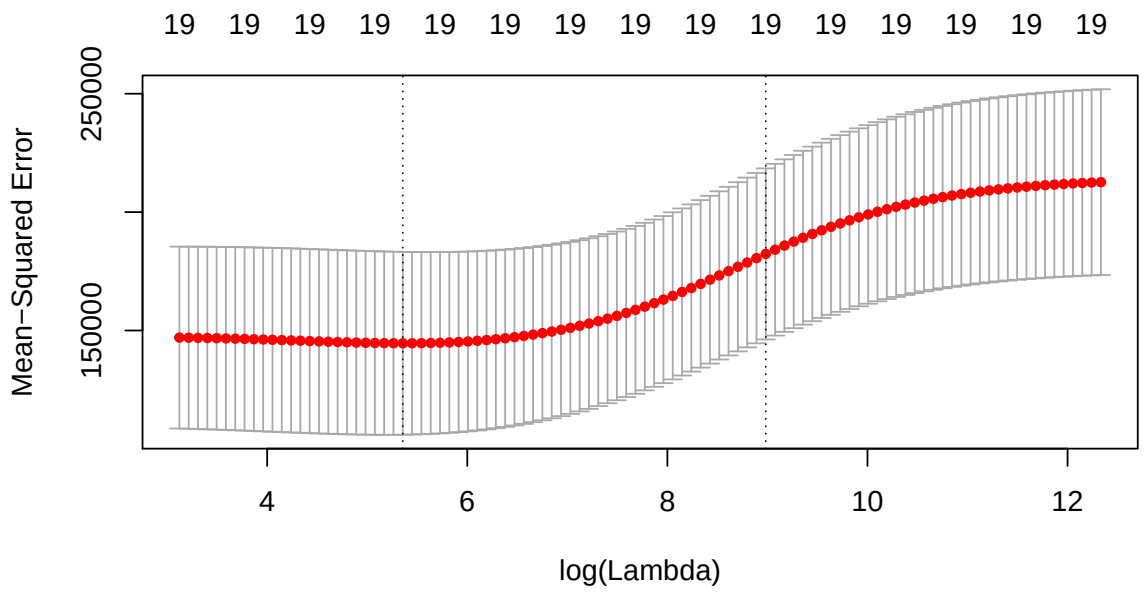
lambda = 0일 때 시험오차이다.

```
lm(y~x, subset=train)

##
## Call:
## lm(formula = y ~ x, subset = train)
##
## Coefficients:
## (Intercept)       xAtBat        xHits       xHmRun        xRuns
##    299.4285      -2.5403       8.3668      11.6451      -9.0992
##        xRBI       xWalks       xYears      xCAtBat       xCHits
##      2.4410       9.2344     -22.9367      -0.1815      -0.1160
##     xCHmRun       xCRuns        xCRBI      xCWalks     xLeagueN
##     -1.3389       3.3284       0.0754      -1.0784      59.7607
##  xDivisionW     xPutOuts     xAssists      xErrors  xNewLeagueN
##    -98.8623       0.3409       0.3416      -0.6421      -0.6744

predict(ridge.mod,s=0,exact=T,type="coefficients")[1:20,]
```

```
## (Intercept)       AtBat        Hits       HmRun        Runs         RBI
##   299.42884    -2.54015     8.36612    11.64401    -9.09878     2.44152
##       Walks       Years      CAtBat       CHits      CHmRun       CRuns
##     9.23404   -22.93584    -0.18161    -0.11561    -1.33837     3.32818
##        CRBI      CWalks     LeagueN   DivisionW     PutOuts     Assists
##     0.07512    -1.07829    59.76529   -98.85997     0.34086     0.34166
##      Errors  NewLeagueN
##    -0.64206    -0.67606
```

lm을 썼을 때와 lambda = 0일 때의 회귀계수 추정치의 비교이다. 두 값이 동일하다.

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



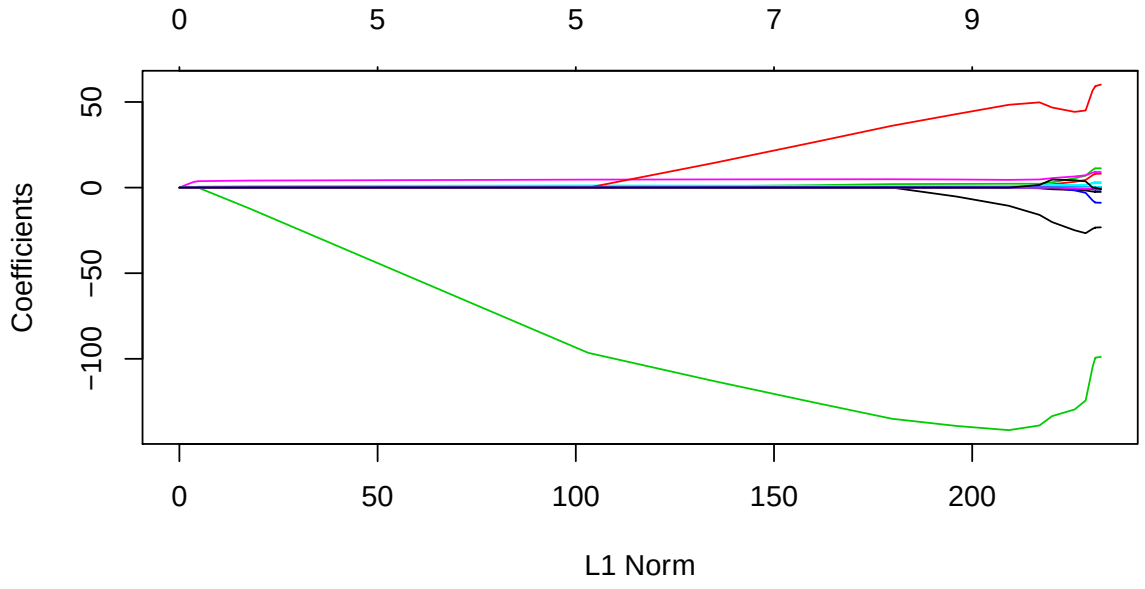

```
bestlam=cv.out$lambda.min
bestlam

## [1] 211.7
```

cv.glmnet은 k 겹 교차검증을 수행하는 함수이다. 옵션 nfold는 겹의 수 k이고, 디폴트 값은 10이다. 최적의 lambda 값은 약 212이다.

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

## [1] 96016

out=glmnet(x,y,alpha=0)
predict(out,type="coefficients",s=bestlam)[1:20,]

## (Intercept)       AtBat        Hits       HmRun        Runs         RBI
##     9.88487     0.03144     1.00883     0.13928     1.11321     0.87319
##       Walks       Years      CAtBat       CHits      CHmRun       CRuns
##     1.80410     0.13074     0.01114     0.06490     0.45159     0.12900
##        CRBI      CWalks     LeagueN   DivisionW     PutOuts     Assists
##     0.13738     0.02909    27.18228   -91.63411     0.19149     0.04255
##      Errors  NewLeagueN
##    -1.81244     7.21208
```

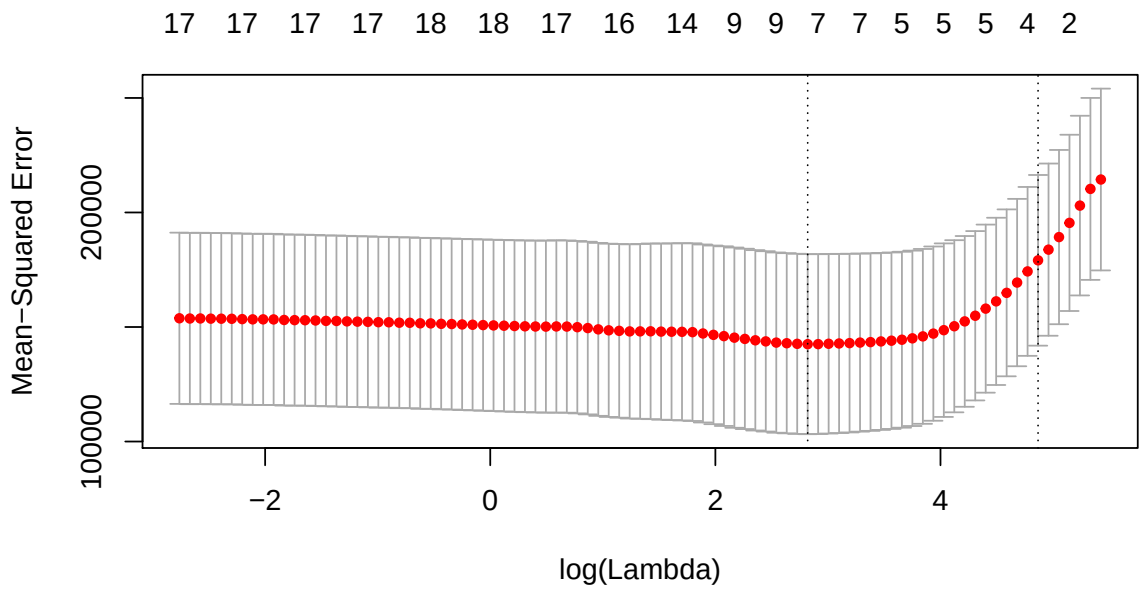
6.2 라쏘

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



alpha=1이면 라쏘를 적합하는 것이다.

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




```
bestlam=cv.out$lambda.min
```

훈련자료를 이용해서 최적의 lambda값을 구했다.

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

## [1] 100743

out=glmnet(x,y,alpha=1,lambda=grid)
lasso.coef=predict(out,type="coefficients",s=bestlam)[1:20,]
lasso.coef

## (Intercept)       AtBat        Hits       HmRun        Runs         RBI
##     18.5395      0.0000      1.8735      0.0000      0.0000      0.0000
##       Walks       Years      CAtBat       CHits      CHmRun       CRuns
##      2.2178      0.0000      0.0000      0.0000      0.0000      0.2071
```

```
##        CRBI      CWalks     LeagueN   DivisionW     PutOuts     Assists
##      0.4130      0.0000      3.2667   -103.4845      0.2204      0.0000
##      Errors  NewLeagueN
##      0.0000      0.0000

lasso.coef[lasso.coef!=0]

## (Intercept)        Hits       Walks       CRuns        CRBI     LeagueN
##     18.5395      1.8735      2.2178      0.2071      0.4130      3.2667
##   DivisionW     PutOuts
##   -103.4845      0.2204
```