Untitled

Michael Guel

10/7/2022

```
library(readxl)
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 4.1.3
```

```
library(lars)
```

```
## Warning: package 'lars' was built under R version 4.1.3
```

```
## Loaded lars 1.3
```

```
data = read_xlsx('prostatedate.xlsx')

train = data[data$train == 'T',]

test = data[data$train == 'F',]

train = train[2:10]

test = test[2:10]

############################## Fit an OLS regression model (with intercept). Report its R2, p-values of i

x = lm(lpsa~.,data=train)

summary(x)
```

```
##
## Call:
## lm(formula = lpsa ~ ., data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.64870 -0.34147 -0.05424  0.44941  1.48675
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.429170   1.553588   0.276  0.78334
## lcavol       0.576543   0.107438   5.366 1.47e-06 ***
```

```
## lweight      0.614020   0.223216   2.751  0.00792 **
## age         -0.019001   0.013612  -1.396  0.16806
## lbph         0.144848   0.070457   2.056  0.04431 *
## svi          0.737209   0.298555   2.469  0.01651 *
## lcp         -0.206324   0.110516  -1.867  0.06697 .
## gleason     -0.029503   0.201136  -0.147  0.88389
## pgg45        0.009465   0.005447   1.738  0.08755 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7123 on 58 degrees of freedom
## Multiple R-squared:  0.6944, Adjusted R-squared:  0.6522
## F-statistic: 16.47 on 8 and 58 DF,  p-value: 2.042e-12
```

```
# R2 = 0.6522

print(deviance(x))
```

```
## [1] 29.42638
```

```
# RSS = 29.42638

############################## TEST DATA

x = lm(lpsa~.,data=test)

summary(x)
```

```
##
## Call:
## lm(formula = lpsa ~ ., data = test)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.11094 -0.37674  0.03635  0.48003  0.97051
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.4284155  2.7300468   0.157   0.8768
## lcavol       0.4557043  0.1772497   2.571   0.0178 *
## lweight      0.5553612  0.5335348   1.041   0.3098
## age         -0.0089424  0.0239842  -0.373   0.7130
## lbph        -0.0810177  0.1244522  -0.651   0.5221
## svi          0.6597360  0.4405151   1.498   0.1491
## lcp          0.1697084  0.1738277   0.976   0.3400
## gleason     -0.0184376  0.2921209  -0.063   0.9503
## pgg45        0.0007157  0.0090719   0.079   0.9379
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6795 on 21 degrees of freedom
## Multiple R-squared:  0.6921, Adjusted R-squared:  0.5748
## F-statistic:   5.9 on 8 and 21 DF,  p-value: 0.0005124
```

```
# R2 = 0.5748

print(deviance(x))
```

```
## [1] 9.695596
```

```
# RSS = 9.695596

############################## Apply forward selection to select variables use R function regsubsets() in

trainsearch = regsubsets(lpsa~.,data=train,nvmax=8,method="forward")

testsearch = regsubsets(lpsa~.,data=test,nvmax=8,method="forward")



traincoef = summary(trainsearch)

traincoef$rss
```

```
## [1] 44.52858 37.09185 34.90775 32.81499 32.06945 30.53978 29.43730 29.42638
```

```
traincoef$bic
```

```
## [1] -43.25728 -51.29578 -51.15720 -51.09467 -48.42976 -47.49961 -45.75833
## [8] -41.57849
```

```
coeffi = coef(trainsearch,1:8)

coeffi
```

```
## [[1]]
## (Intercept)      lcavol
##   1.5163048   0.7126351
##
## [[2]]
## (Intercept)      lcavol     lweight
##  -1.0494396   0.6276074   0.7383751
##
## [[3]]
## (Intercept)      lcavol     lweight         svi
##  -1.0227780   0.5199861   0.7367954   0.5379032
##
## [[4]]
## (Intercept)      lcavol     lweight        lbph         svi
##  -0.3259212   0.5055209   0.5388292   0.1400111   0.6718487
##
## [[5]]
##  (Intercept)       lcavol      lweight         lbph          svi        pgg45
## -0.465877591  0.472278483  0.563935476  0.137116261  0.578163005  0.004330753
##
```

```
## [[6]]
##  (Intercept)       lcavol      lweight         lbph          svi          lcp
## -0.728972257  0.549778034  0.563105747  0.125978836  0.756354835 -0.190824719
##        pgg45
##  0.007541236
##
## [[7]]
##  (Intercept)       lcavol      lweight          age         lbph          svi
##  0.259061747  0.573930391  0.619208833 -0.019479879  0.144426474  0.741781258
##          lcp        pgg45
## -0.205416986  0.008944996
##
## [[8]]
##  (Intercept)       lcavol      lweight          age         lbph          svi
##  0.429170133  0.576543185  0.614020004 -0.019001022  0.144848082  0.737208645
##          lcp      gleason        pgg45
## -0.206324227 -0.029502884  0.009465162
```

```
testcoef = summary(testsearch)

testcoef$rss
```

```
## [1] 14.366665 10.865766 10.307142 10.148351  9.760796  9.698529  9.697435
## [8]  9.695596
```

```
testcoef$bic
```

```
## [1] -16.739721 -21.717329 -19.899532 -16.964110 -14.731029 -11.521825  -8.124011
## [8]  -4.728504
```

```
coeffi = coef(testsearch,1:8)

coeffi
```

```
## [[1]]
## (Intercept)      lcavol
##   1.4754147   0.7412385
##
## [[2]]
## (Intercept)      lcavol         svi
##   1.4977515   0.5955364   0.9312273
##
## [[3]]
## (Intercept)      lcavol         svi         lcp
##   1.7066474   0.5007631   0.6465740   0.1604053
##
## [[4]]
## (Intercept)      lcavol     lweight         svi         lcp
##   0.7918648   0.4922957   0.2534292   0.6688351   0.1489109
##
## [[5]]
## (Intercept)      lcavol     lweight        lbph         svi         lcp
```

```
##  0.03133947  0.43934512  0.49112380 -0.09869627  0.63146120  0.18833670
##
## [[6]]
##  (Intercept)       lcavol      lweight          age         lbph          svi
##  0.309743339  0.453080344  0.555307619 -0.008767265 -0.078820589  0.663025505
##          lcp
##  0.171498117
##
## [[7]]
##   (Intercept)        lcavol       lweight           age          lbph
##  0.3074659816  0.4533397507  0.5556365833 -0.0088593631 -0.0801778848
##           svi           lcp         pgg45
##  0.6626831537  0.1683604279  0.0003165936
##
## [[8]]
##   (Intercept)        lcavol       lweight           age          lbph
##  0.4284155359  0.4557042872  0.5553611979 -0.0089424306 -0.0810176975
##           svi           lcp       gleason         pgg45
##  0.6597360274  0.1697083918 -0.0184375621  0.0007157027
```

```
############################## MINIMIZE BIC BEST MODEL

which.min(traincoef$bic)
```

```
## [1] 2
```

```
which.min(testcoef$bic)
```

```
## [1] 2
```

```
# 2 MINIMIZES BIC

coef(trainsearch,2)
```

```
## (Intercept)      lcavol     lweight
##  -1.0494396   0.6276074   0.7383751
```

```
coef(testsearch,2)
```

```
## (Intercept)      lcavol         svi
##   1.4977515   0.5955364   0.9312273
```

```
finalmodelbic = lm(lpsa~lcavol+lweight,data=test)

summary(finalmodelbic)
```

```
##
## Call:
## lm(formula = lpsa ~ lcavol + lweight, data = test)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -1.29207 -0.49256 -0.05637  0.35382  1.61070
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.4078     1.6355   0.249    0.805
## lcavol        0.7235     0.1324   5.464 8.77e-06 ***
## lweight       0.3007     0.4561   0.659    0.515
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7237 on 27 degrees of freedom
## Multiple R-squared:  0.551,  Adjusted R-squared:  0.5177
## F-statistic: 16.57 on 2 and 27 DF,  p-value: 2.021e-05
```

```
deviance(finalmodelbic)
```

```
## [1] 14.13908
```

```
############################### In 2, replace BIC by AIC where AIC = nlog(RSStrn/n) + 2|M^j|. Choose the

r = data.frame(testcoef$rss)

i = numeric()

for (x in 1:length(r$testcoef.rss)){
  t = log(((r$testcoef.rss[x])/30))+2*abs(x+1)
  i = c(i,t)
}

min(i)
```

```
## [1] 3.263713
```

```
##### AIC IS LOWEST WITH FIRST ITERATION

coeffi[1]
```

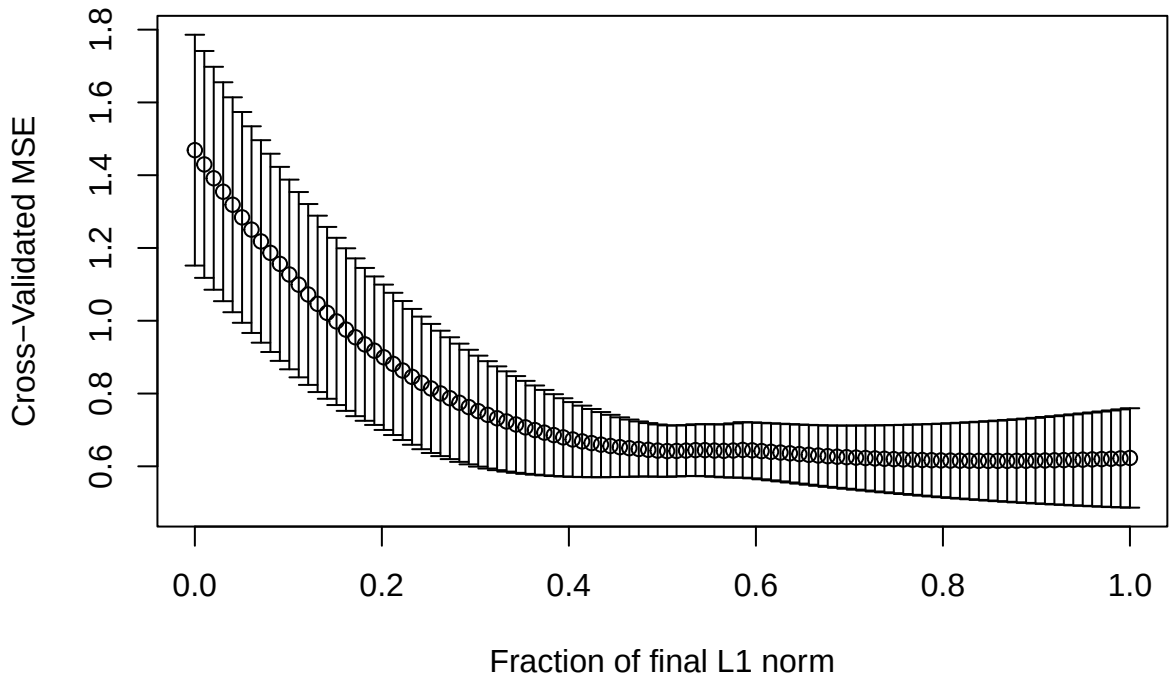
```
## [[1]]
## (Intercept)      lcavol
##   1.4754147   0.7412385
```

```
# lcavol

############################### Use R functions lars() and cv.lars()) to fit Lasso without an intercept

#split matrix of predictors and response

trainpred = data.matrix(train[1:8])

trainrep = data.matrix(train[9])
```

```
q = lars(trainpred,trainrep,type = 'lasso')

summary(q)
```

```
## LARS/LASSO
## Call: lars(x = trainpred, y = trainrep, type = "lasso")
##   Df    Rss       Cp
## 0  1 96.281 124.7727
## 1  2 58.347  52.0025
## 2  3 50.391  38.3213
## 3  4 40.271  20.3741
## 4  5 40.012  21.8653
## 5  6 32.738   9.5274
## 6  7 32.069  10.2082
## 7  8 29.468   7.0828
## 8  9 29.426   9.0000
```

```
cv.lars(trainpred,trainrep,K=5,type = 'lasso')
```




```
############################## FIND BEST MODEL WITH VALIDATION MATRIX

search= regsubsets(lpsa~.,data=train,nvmax=8,method="forward")
valid.mat=model.matrix(lpsa~.,test)
```

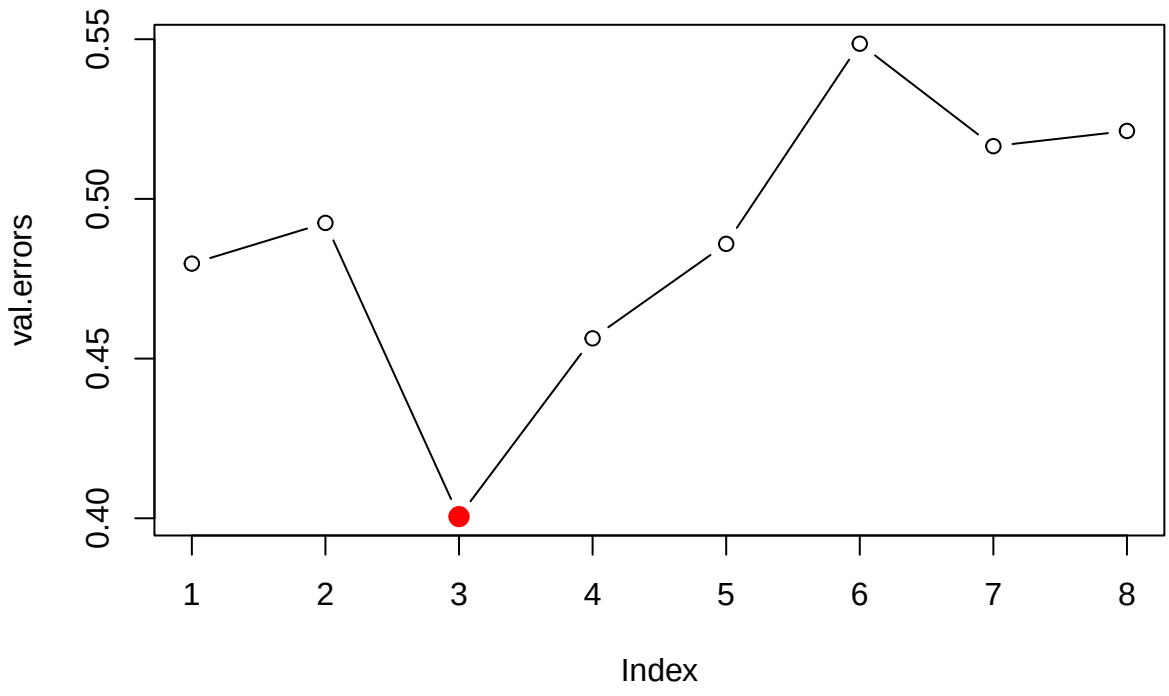
```
val.errors=numeric(8)
for(i in 1:8){
  coefi=coef(search,id=i)
  pred=valid.mat[,names(coefi)]%*%coefi
  val.errors[i]=mean((test$lpsa-pred)^2)
}
val.errors
```

```
## [1] 0.4797387 0.4924823 0.4005308 0.4563321 0.4859242 0.5485933 0.5165135
## [8] 0.5212740
```

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

```
## [1] 3
```

```
plot(val.errors,type="b")
points(best,val.errors[best],col="red",cex=2,pch=20)
```




```
coef(search,best)
```

```
## (Intercept)      lcavol     lweight         svi
##  -1.0227780   0.5199861   0.7367954   0.5379032
```

```
res=summary(search)
res$adjr2[best]
```

```
## [1] 0.6201758
```

```
summary(search)
```

```
## Subset selection object
## Call: regsubsets.formula(lpsa ~ ., data = train, nvmax = 8, method = "forward")
## 8 Variables  (and intercept)
##         Forced in Forced out
## lcavol      FALSE      FALSE
## lweight     FALSE      FALSE
## age         FALSE      FALSE
## lbph        FALSE      FALSE
## svi         FALSE      FALSE
## lcp         FALSE      FALSE
## gleason     FALSE      FALSE
## pgg45       FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: forward
##          lcavol lweight age lbph svi lcp gleason pgg45
## 1  ( 1 ) "*"    " "     " " " "  " " " " " " "     " "
## 2  ( 1 ) "*"    "*"     " " " "  " " " " " " "     " "
## 3  ( 1 ) "*"    "*"     " " " "  "*" " " " " "     " "
## 4  ( 1 ) "*"    "*"     " " "*"  "*" " " " " "     " "
## 5  ( 1 ) "*"    "*"     " " "*"  "*" " " " " "     "*"
## 6  ( 1 ) "*"    "*"     " " "*"  "*" "*" " " "     "*"
## 7  ( 1 ) "*"    "*"     "*" "*"  "*" "*" " " "     "*"
## 8  ( 1 ) "*"    "*"     "*" "*"  "*" "*" "*"     "*"
```