STATS 500 HW8

Minxuan Chen

2023-11-16

Table of contents

Github repo: https://github.com/PKUniiiiice/STATS_500

Problem 1

```
library(pls)
library(MASS)
library(glmnet)
library(lars)
gasdata = data.frame(cbind(gasoline$octane,
                           gasoline$NIR[,c(1:40)]))
names(gasdata)=c('octane','NIR1','NIR2','NIR3','NIR4','NIR5','NIR6','NIR7','NIR8','NIR9'
'NIR10','NIR11','NIR12','NIR13','NIR14','NIR15','NIR16','NIR17','NIR18',
'NIR19','NIR20','NIR21','NIR22','NIR23','NIR24','NIR25','NIR26','NIR27',
'NIR28','NIR29','NIR30','NIR31','NIR32','NIR33','NIR34','NIR35','NIR36',
'NIR37','NIR38','NIR39','NIR40')
c = seq(1,56,5)
gasdata_tr = gasdata[-c,]
gasdata_te = gasdata[c,]

rmse <- function(x, y){
  return(sqrt(mean((x-y)^2)))
}
```

Linear regression with all predictors

```
m.linear <- lm(octane ~ . , data=gasdata_tr)
summary(m.linear)
```

```
Call:
lm(formula = octane ~ ., data = gasdata_tr)

Residuals:
     Min       1Q   Median       3Q      Max
-0.43436 -0.13133 -0.04657  0.08413  0.44421

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   107.02      20.01   5.349  0.00107 **
NIR1           35.64     595.87   0.060  0.95398
NIR2          384.21    1005.27   0.382  0.71365
NIR3           10.17    1197.86   0.008  0.99346
NIR4         -297.77     894.68  -0.333  0.74901
NIR5          -93.43     710.43  -0.132  0.89907
NIR6          -14.22     807.62  -0.018  0.98645
NIR7          -35.36     587.05  -0.060  0.95365
NIR8          -47.13     696.38  -0.068  0.94793
```

```
NIR9          219.70     733.58   0.299  0.77326
NIR10         636.73     640.08   0.995  0.35299
NIR11        -354.73     850.22  -0.417  0.68901
NIR12        -905.76     958.04  -0.945  0.37593
NIR13         253.90    1035.57   0.245  0.81335
NIR14        -300.37     769.91  -0.390  0.70803
NIR15        -444.73    1209.94  -0.368  0.72406
NIR16         102.40     681.54   0.150  0.88481
NIR17         330.27     777.94   0.425  0.68391
NIR18        -986.47    1004.00  -0.983  0.35856
NIR19       -1102.90    1753.06  -0.629  0.54924
NIR20        -186.82    1692.29  -0.110  0.91519
NIR21        1767.13    1663.56   1.062  0.32339
NIR22         830.11    1632.29   0.509  0.62670
NIR23         558.40    1523.42   0.367  0.72479
NIR24          33.31    1073.68   0.031  0.97612
NIR25         -71.80    1115.86  -0.064  0.95050
NIR26        -415.67    1705.07  -0.244  0.81439
NIR27        1044.02    1327.17   0.787  0.45729
NIR28       -1774.28    2249.69  -0.789  0.45618
NIR29        1152.78    1935.89   0.595  0.57027
NIR30        -844.72    1964.14  -0.430  0.68007
NIR31        -701.52    2374.26  -0.295  0.77621
NIR32         473.06    2108.36   0.224  0.82888
NIR33         719.48    2467.20   0.292  0.77903
NIR34        1421.30    2230.36   0.637  0.54423
NIR35        -884.61    1775.08  -0.498  0.63351
NIR36       -1848.97    2428.44  -0.761  0.47130
NIR37         899.40    2267.59   0.397  0.70345
NIR38         304.03    1339.73   0.227  0.82696
NIR39         893.17    1379.99   0.647  0.53812
NIR40        -745.12    1682.32  -0.443  0.67119
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5232 on 7 degrees of freedom
Multiple R-squared:  0.983, Adjusted R-squared:  0.8858
F-statistic: 10.11 on 40 and 7 DF,  p-value: 0.001893
```

```r
#training error
paste("Training error",
      tr.linear <- rmse(m.linear$fitted.values,
                        gasdata_tr$octane))
```

```
[1] "Training error 0.199794670776533"
```

```r
#test error
paste("Test error", te.linear <- rmse(predict(m.linear, newdata = gasdata_te),gasdata_te
```

```
[1] "Test error 0.510418176467638"
```

Linear regression with variables selected using AIC

```r
#AIC
step(m.linear, trace=FALSE)
```

```
Call:
lm(formula = octane ~ NIR2 + NIR4 + NIR10 + NIR12 + NIR14 + NIR15 +
    NIR17 + NIR18 + NIR19 + NIR21 + NIR23 + NIR27 + NIR28 + NIR29 +
    NIR30 + NIR31 + NIR33 + NIR34 + NIR36 + NIR38 + NIR39 + NIR40,
    data = gasdata_tr)

Coefficients:
(Intercept)         NIR2         NIR4        NIR10        NIR12        NIR14
       99.7        684.4       -508.3        669.8       -744.8       -275.1
     NIR15        NIR17        NIR18        NIR19        NIR21        NIR23
    -484.8        370.3      -1117.2       -691.3       1460.4        595.2
     NIR27        NIR28        NIR29        NIR30        NIR31        NIR33
     965.0      -1330.5        830.0       -954.0      -1133.2       1231.0
     NIR34        NIR36        NIR38        NIR39        NIR40
    1341.4      -1169.6        666.7        718.8      -1123.0
```

```r
m.AIC <- lm(octane ~ NIR2 + NIR4 + NIR10 + NIR12 +
                     NIR14 + NIR15 + NIR17 + NIR18 + NIR19 +
                     NIR21 + NIR23 + NIR27 + NIR28 + NIR29 +
                     NIR30 + NIR31 + NIR33 + NIR34 + NIR36 +
                     NIR38 + NIR39 + NIR40,
               data = gasdata_tr)
summary(m.AIC)
```

```
Call:
lm(formula = octane ~ NIR2 + NIR4 + NIR10 + NIR12 + NIR14 + NIR15 +
    NIR17 + NIR18 + NIR19 + NIR21 + NIR23 + NIR27 + NIR28 + NIR29 +
    NIR30 + NIR31 + NIR33 + NIR34 + NIR36 + NIR38 + NIR39 + NIR40,
    data = gasdata_tr)

Residuals:
     Min       1Q   Median       3Q      Max
```

```
-0.42681 -0.14459 -0.05653  0.15007  0.51993

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    99.696      4.179  23.858  < 2e-16 ***
NIR2          684.419    179.710   3.808 0.000809 ***
NIR4         -508.335    104.534  -4.863 5.32e-05 ***
NIR10         669.787    139.875   4.788 6.45e-05 ***
NIR12        -744.776    266.674  -2.793 0.009874 **
NIR14        -275.066    252.737  -1.088 0.286823
NIR15        -484.831    335.843  -1.444 0.161260
NIR17         370.275    281.261   1.316 0.199955
NIR18       -1117.209    319.204  -3.500 0.001766 **
NIR19        -691.291    356.085  -1.941 0.063567 .
NIR21        1460.441    456.963   3.196 0.003753 **
NIR23         595.236    406.480   1.464 0.155556
NIR27         965.036    334.688   2.883 0.007976 **
NIR28       -1330.519    533.443  -2.494 0.019596 *
NIR29         830.047    482.421   1.721 0.097682 .
NIR30        -954.016    622.262  -1.533 0.137800
NIR31       -1133.170    568.326  -1.994 0.057183 .
NIR33        1230.981    546.001   2.255 0.033172 *
NIR34        1341.382    503.985   2.662 0.013396 *
NIR36       -1169.612    659.330  -1.774 0.088255 .
NIR38         666.669    380.654   1.751 0.092142 .
NIR39         718.798    433.285   1.659 0.109621
NIR40       -1123.047    594.768  -1.888 0.070656 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3035 on 25 degrees of freedom
Multiple R-squared:  0.9796,    Adjusted R-squared:  0.9616
F-statistic: 54.44 on 22 and 25 DF,  p-value: 5.448e-16
```

```r
#training error
paste("Training error", tr.AIC <- rmse(m.AIC$fitted.values,
                            gasdata_tr$octane))
```

```
[1] "Training error 0.219023876211711"
```

```r
#test error
paste("Test error", te.AIC <- rmse(predict(m.AIC, newdata = gasdata_te),
                                   gasdata_te$octane))
```

```
[1] "Test error 0.665773147711337"
```

Principal component regression - using CV to pick order of model

```r
#PCR
m.PCR <- pcr(octane~., data=gasdata_tr, validation = "CV")
summary(m.PCR)
```

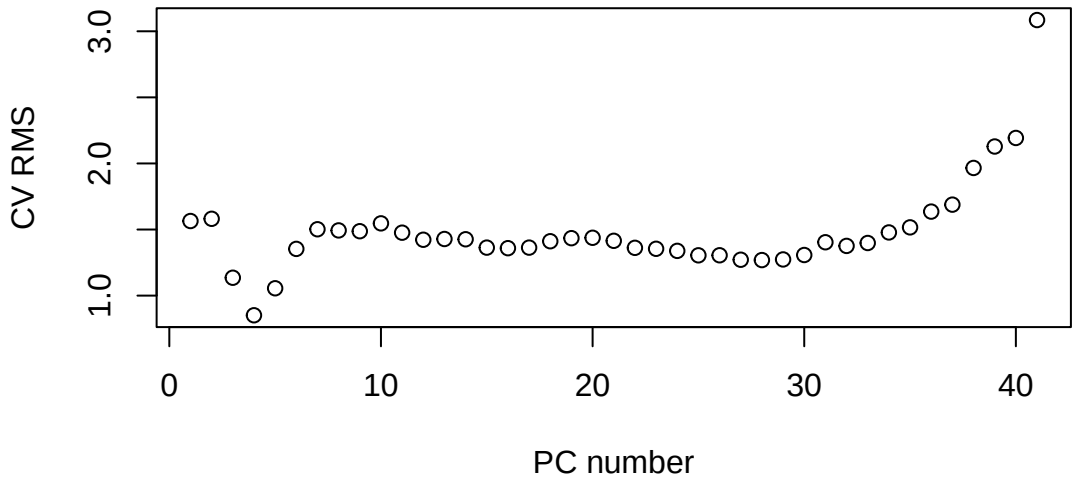
```
Data:   X dimension: 48 40
    Y dimension: 48 1
Fit method: svdpc
Number of components considered: 40

VALIDATION: RMSEP
Cross-validated using 10 random segments.
       (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
CV           1.564    1.581    1.135   0.8511    1.055    1.353    1.502
adjCV        1.564    1.578    1.068   0.8450    1.037    1.319    1.461
       7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
CV       1.493    1.487    1.547     1.476     1.423     1.428     1.426
adjCV    1.446    1.443    1.501     1.416     1.370     1.376     1.373
       14 comps  15 comps  16 comps  17 comps  18 comps  19 comps  20 comps
CV        1.363     1.359     1.363     1.411     1.434     1.438     1.415
adjCV     1.309     1.305     1.309     1.355     1.378     1.381     1.359
       21 comps  22 comps  23 comps  24 comps  25 comps  26 comps  27 comps
CV        1.361     1.354     1.339     1.305     1.305     1.272     1.269
adjCV     1.305     1.298     1.284     1.251     1.252     1.219     1.217
       28 comps  29 comps  30 comps  31 comps  32 comps  33 comps  34 comps
CV        1.274     1.307     1.404     1.376     1.398     1.478     1.516
adjCV     1.222     1.255     1.348     1.319     1.340     1.417     1.452
       35 comps  36 comps  37 comps  38 comps  39 comps  40 comps
CV        1.636     1.688     1.966     2.128     2.193     3.085
adjCV     1.564     1.614     1.875     2.029     2.091     2.936

TRAINING: % variance explained
        1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
X       96.6413    98.08    99.31    99.64    99.72    99.79    99.84    99.88
octane   0.1102    61.00    72.79    81.89    82.05    83.40    85.88    86.21
        9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
X         99.90     99.92     99.93     99.94     99.95     99.96     99.96
octane    86.62     94.45     94.47     94.47     94.99     95.85     96.03
        16 comps  17 comps  18 comps  19 comps  20 comps  21 comps  22 comps
X          99.97     99.97     99.98     99.98     99.98     99.99     99.99
octane     96.04     96.08     96.20     96.23     96.68     97.08     97.09
        23 comps  24 comps  25 comps  26 comps  27 comps  28 comps  29 comps
X          99.99     99.99     99.99     99.99    100.00    100.00    100.00
octane     97.10     97.26     97.26     97.35     97.37     97.39     97.39
        30 comps  31 comps  32 comps  33 comps  34 comps  35 comps  36 comps
```

```
X          100.0    100.00    100.00    100.00    100.00    100.00    100.00
octane      97.4     97.84     97.87     97.88     97.88     98.06     98.06
        37 comps  38 comps  39 comps  40 comps
X         100.00     100.0    100.00     100.0
octane     98.14      98.2     98.27      98.3
```

```
rmsCV <- RMSEP(m.PCR, estimate='CV')
which.min(rmsCV$val)
```

```
[1] 4
```

```
#plot
plot(rmsCV$val, xlab="PC number", ylab="CV RMS")
```

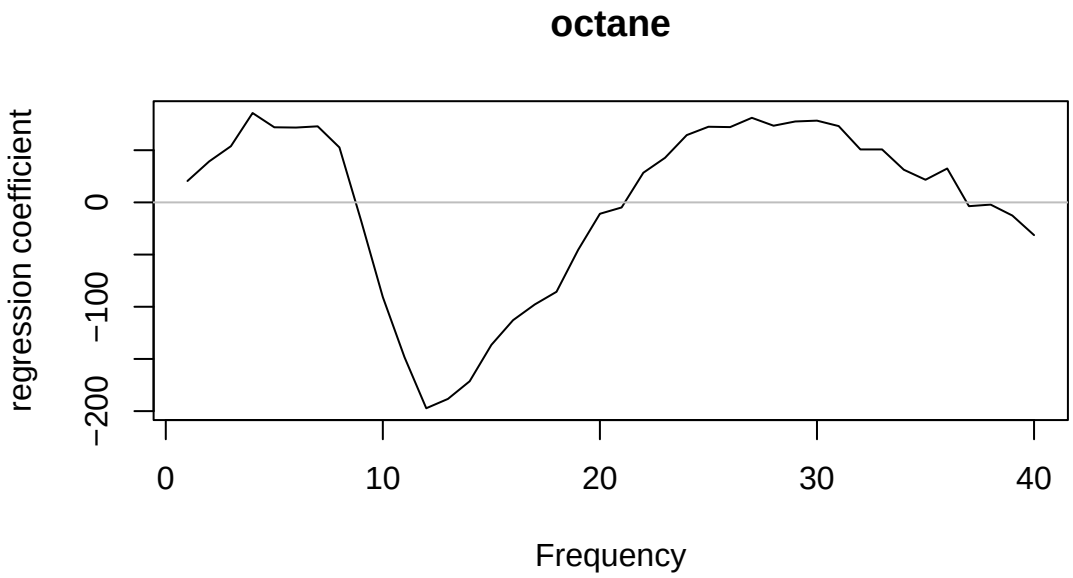



```
m.PCR.best <- pcr(octane~., data=gasdata_tr, ncomp=4, validation = "CV")
summary(m.PCR.best)
```

```
Data:   X dimension: 48 40
    Y dimension: 48 1
Fit method: svdpc
Number of components considered: 4

VALIDATION: RMSEP
Cross-validated using 10 random segments.
       (Intercept)  1 comps  2 comps  3 comps  4 comps
CV           1.564    1.584     1.05   0.9144    1.161
adjCV        1.564    1.581     1.02   0.9055    1.132

TRAINING: % variance explained
        1 comps  2 comps  3 comps  4 comps
```

```
X       96.6413    98.08    99.31    99.64
octane   0.1102    61.00    72.79    81.89
```

```
coefplot(m.PCR.best, ncomp=4, xlab="Frequency")
```




```
pred.pctr <-  predict(m.PCR, newdata=gasdata_tr, ncomp=4)
pred.pcte <-  predict(m.PCR, newdata=gasdata_te, ncomp=4)
#training error
paste("Training error", tr.PCR <- rmse(pred.pctr,
                              gasdata_tr$octane))
```

```
[1] "Training error 0.651816057353665"
```

```
#test error
paste("Test error", te.PCR <- rmse(pred.pcte,gasdata_te$octane))
```

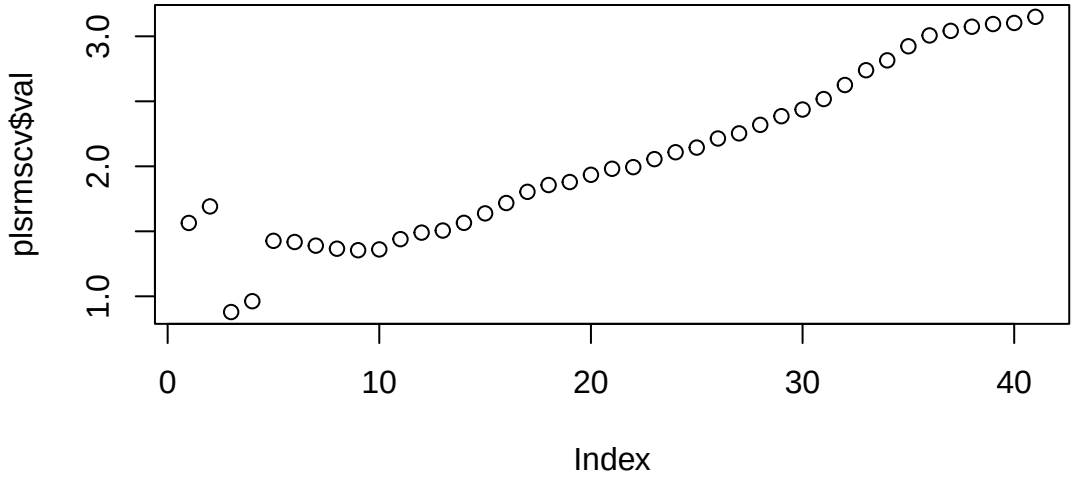
```
[1] "Test error 0.421555414203408"
```

Partial least squares - using CV to pick order of model

```
m.pls <- plsr(octane ~ ., data = gasdata_tr,
                validation = "CV")
summary(m.pls)
```

```
Data:   X dimension: 48 40
    Y dimension: 48 1
Fit method: kernelpls
Number of components considered: 40

VALIDATION: RMSEP
Cross-validated using 10 random segments.
       (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
CV           1.564    1.691   0.8789   0.9622    1.427    1.418    1.390
adjCV        1.564    1.636   0.8717   0.9476    1.377    1.357    1.324
       7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
CV       1.366    1.355    1.361      1.44     1.490     1.506     1.565
adjCV    1.301    1.291    1.295      1.37     1.417     1.432     1.488
       14 comps  15 comps  16 comps  17 comps  18 comps  19 comps  20 comps
CV        1.638     1.717     1.804     1.856     1.879     1.935     1.981
adjCV     1.557     1.633     1.714     1.764     1.786     1.839     1.884
       21 comps  22 comps  23 comps  24 comps  25 comps  26 comps  27 comps
CV        1.994     2.055     2.108     2.144     2.214     2.253     2.319
adjCV     1.896     1.955     2.006     2.040     2.106     2.143     2.205
       28 comps  29 comps  30 comps  31 comps  32 comps  33 comps  34 comps
CV        2.385     2.437     2.517     2.625     2.739     2.815     2.923
adjCV     2.268     2.316     2.392     2.494     2.602     2.673     2.775
       35 comps  36 comps  37 comps  38 comps  39 comps  40 comps
CV        3.007     3.042     3.074     3.094     3.103     3.150
adjCV     2.854     2.887     2.917     2.937     2.945     2.989

TRAINING: % variance explained
        1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
X         84.87    98.05    98.72    99.63    99.69    99.73    99.78    99.84
octane    11.50    76.97    83.03    85.21    92.64    95.68    96.49    96.73
        9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
X         99.86     99.88     99.90     99.93     99.93     99.94     99.95
octane    97.24     97.45     97.64     97.74     97.90     98.01     98.06
        16 comps  17 comps  18 comps  19 comps  20 comps  21 comps  22 comps
X          99.95     99.96     99.96     99.96     99.97     99.98     99.98
octane     98.10     98.14     98.17     98.21     98.23     98.25     98.26
        23 comps  24 comps  25 comps  26 comps  27 comps  28 comps  29 comps
X          99.98     99.98     99.99     99.99     99.99     99.99     99.99
octane     98.26     98.27     98.29     98.29     98.29     98.30     98.30
        30 comps  31 comps  32 comps  33 comps  34 comps  35 comps  36 comps
X          99.99     99.99     100.0     100.0     100.0     100.0     100.0
octane     98.30     98.30      98.3      98.3      98.3      98.3      98.3
        37 comps  38 comps  39 comps  40 comps
X          100.0     100.0     100.0     100.0
octane      98.3      98.3      98.3      98.3
```

```
plsrmscv <- RMSEP(m.pls,estimate='CV')
plot(plsrmscv$val)
```




```
which.min(plsrmscv$val)
```

```
[1] 3
```

```
pred.plstr = predict(m.pls, newdata=gasdata_tr, ncomp=3)
pred.plste = predict(m.pls, newdata=gasdata_te, ncomp=3)

#training error
paste("Training error", tr.pls <- rmse(pred.plstr,
                              gasdata_tr$octane))
```

```
[1] "Training error 0.6309353614644"
```

```
#test error
paste("Test error", te.pls <- rmse(pred.plste,gasdata_te$octane))
```

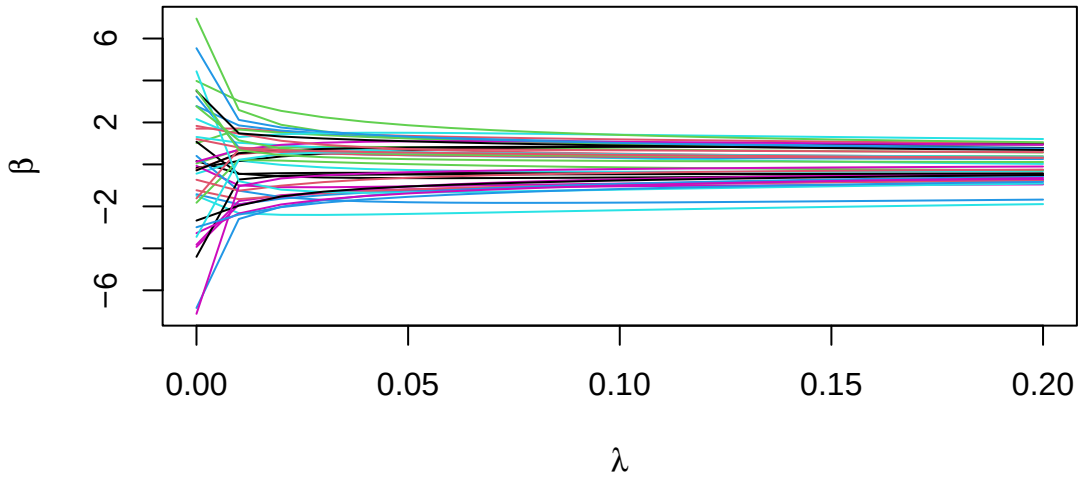
```
[1] "Test error 0.32577507300782"
```

Ridge regression - using GCV to pick regularization parameter

```
m.ridge <-  lm.ridge(octane ~ .,
                     lambda=seq(0, .2, .01),
                     data = gasdata_tr)
head(m.ridge$coef)
```

```
            0.00       0.01        0.02       0.03       0.04       0.05
NIR1  0.16119223 -0.4436893 -0.57325342 -0.6232798 -0.6451038 -0.6533857
NIR2  1.70484815  1.7086173  1.56902266  1.4791696  1.4128115  1.3603658
NIR3  0.04577975  0.7002692  0.81807474  0.8413829  0.8367568  0.8215341
NIR4 -1.42156175 -1.9007774 -1.63293466 -1.4617888 -1.3408500 -1.2492333
NIR5 -0.43953606  0.1809873 -0.02078692 -0.1427672 -0.2191168 -0.2686981
NIR6 -0.06985119 -0.9974199 -1.08400188 -1.0876496 -1.0697388 -1.0449144
           0.06       0.07       0.08       0.09       0.10       0.11
NIR1 -0.6540934 -0.6501841 -0.6433181 -0.6345095 -0.6244151 -0.6134783
NIR2  1.3170269  1.2800392  1.2476937  1.2188703  1.1928032  1.1689502
NIR3  0.8020527  0.7810546  0.7598394  0.7390505  0.7190060  0.6998534
NIR4 -1.1764229 -1.1164996 -1.0658482 -1.0221226 -0.9837278 -0.9495380
NIR5 -0.3014929 -0.3231959 -0.3372779 -0.3459666 -0.3507517 -0.3526627
NIR6 -1.0180082 -0.9908823 -0.9643080 -0.9386054 -0.9138929 -0.8901947
           0.12       0.13       0.14       0.15       0.16       0.17
NIR1 -0.6020081 -0.5902244 -0.5782861 -0.5663094 -0.5543795 -0.5425594
NIR2  1.1469160  1.1264051  1.1071916  1.0890992  1.0719879  1.0557448
NIR3  0.6816481  0.6643943  0.6480682  0.6326309  0.6180362  0.6042352
NIR4 -0.9187347 -0.8907071 -0.8649900 -0.8412221 -0.8191184 -0.7984508
NIR5 -0.3524303 -0.3505844 -0.3475168 -0.3435214 -0.3388218 -0.3335899
NIR6 -0.8674908 -0.8457405 -0.8248946 -0.8049016 -0.7857107 -0.7672733
           0.18       0.19       0.20
NIR1 -0.5308952 -0.5194203 -0.5081589
NIR2  1.0402773  1.0255086  1.0113742
NIR3  0.5911786  0.5788184  0.5671088
NIR4 -0.7790344 -0.7607174 -0.7433744
NIR5 -0.3279592 -0.3220339 -0.3158964
NIR6 -0.7495437 -0.7324793 -0.7160407
```

```r
matplot(m.ridge$lambda,
        t(m.ridge$coef), type="l",
        lty=1,xlab=expression(lambda),
        ylab=expression(hat(beta)))
```



```r
## Select an appropriate lambda
select(m.ridge)
```

```
modified HKB estimator is 0.02736696
modified L-W estimator is 4.510166
smallest value of GCV  at 0.07
```

```r
m.ridge.best <- lm.ridge(octane ~ .,
                         lambda=0.07,
                         data = gasdata_tr); m.ridge.best$coef
```

```
       NIR1        NIR2        NIR3        NIR4        NIR5        NIR6
-0.65018407  1.28003922  0.78105457 -1.11649959 -0.32319592 -0.99088228
       NIR7        NIR8        NIR9       NIR10       NIR11       NIR12
 0.77130035  0.70097931  1.12018974  1.19684236 -2.28998131 -1.11024805
      NIR13       NIR14       NIR15       NIR16       NIR17       NIR18
-0.41630192 -1.07836555 -0.06183079 -1.83155778  0.45444323 -0.87501642
      NIR19       NIR20       NIR21       NIR22       NIR23       NIR24
-0.47712229 -0.51357628  1.11889820  0.38226678  1.47916562  1.10589907
      NIR25       NIR26       NIR27       NIR28       NIR29       NIR30
 0.83503834  0.39766529  1.63225580 -1.12931271 -1.29815497 -1.19598157
      NIR31       NIR32       NIR33       NIR34       NIR35       NIR36
-0.90783681  0.61638616  0.44403495  1.17419805  0.54167824 -0.29424581
      NIR37       NIR38       NIR39       NIR40
 1.00761541  0.51718366  0.20540635 -1.36124868
```

```r
yfittr <-  m.ridge.best$ym + scale(gasdata_tr[,-1],
                                   center=m.ridge.best$xm,
                                   scale=m.ridge.best$scales) %*%
                                   m.ridge.best$coef
#  training error
paste("Training error", tr.ridge <- rmse(yfittr,
                                 gasdata_tr$octane))
```

```
[1] "Training error 0.279915048711154"
```

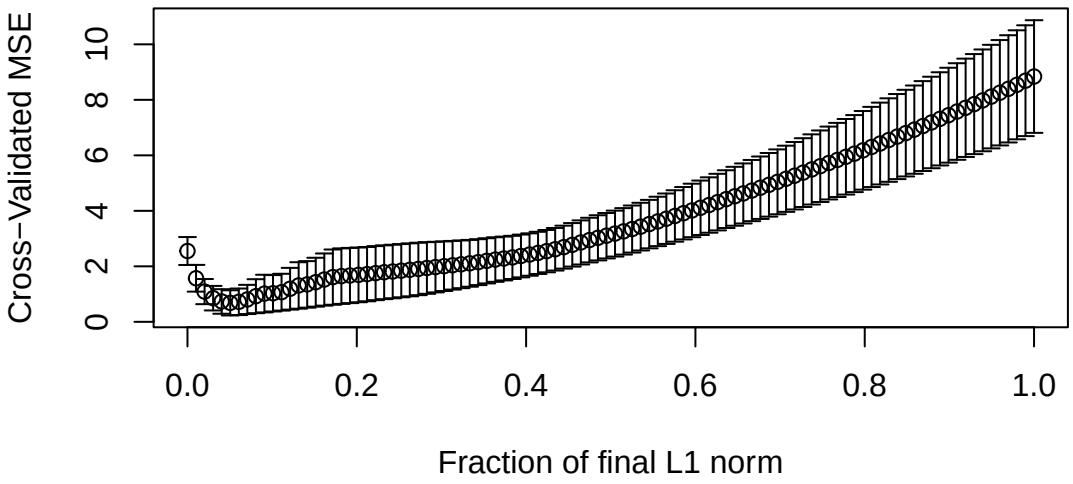
```r
yfitte <-  m.ridge.best$ym + scale(gasdata_te[,-1],
                                     center=m.ridge.best$xm,
                                     scale=m.ridge.best$scales) %*%
                                     m.ridge.best$coef
#test error
paste("Test error", te.ridge <- rmse(yfitte,
```

```
                          gasdata_te$octane))
```

```
[1] "Test error 0.491558224890926"
```

Lasso regression - using CV to pick regularization parameter t.

```
set.seed(123)
m.lasso <-  lars(as.matrix(gasdata_tr[,-1]),
                 gasdata_tr$octane)
cvout <-  cv.lars(as.matrix(gasdata_tr[,-1]),
                  gasdata_tr$octane)
```




```
cvout$index[which.min(cvout$cv)]
```

```
[1] 0.05050505
```

```
m.lasso <-  lars(as.matrix(gasdata_tr[,-1]),gasdata_tr$octane)
predlars_tr <-  predict(m.lasso,as.matrix(gasdata_tr[,-1]),
                        s=cvout$index[which.min(cvout$cv)],
                        mode="fraction")$fit
#  training error
paste("Training error", tr.lasso <- rmse(predlars_tr, gasdata_tr$octane))
```

```
[1] "Training error 0.826073321265821"
```

```
predlars_te <-  predict(m.lasso,as.matrix(gasdata_te[,-1]),
                        s=cvout$index[which.min(cvout$cv)],
```

```
                          mode="fraction")$fit
#test error
paste("Test error", te.lasso <- rmse(predlars_te, gasdata_te$octane))
```

```
[1] "Test error 0.646539408291542"
```

Summary

```
rmse.data <- matrix(c(tr.linear, tr.AIC, tr.PCR, tr.pls, tr.ridge,tr.lasso,
                      te.linear, te.AIC, te.PCR, te.pls, te.ridge, te.lasso),
                    ncol=6, byrow=1)
colnames(rmse.data) <- c("LR", "LR.AIC", "PCR",
                         "PLS", "Ridge","Lasso")
row.names(rmse.data) <- c("Training RMSE", "Test RMSE")

print(rmse.data)
```

```
                    LR    LR.AIC       PCR       PLS     Ridge     Lasso
Training RMSE 0.1997947 0.2190239 0.6518161 0.6309354 0.2799150 0.8260733
Test RMSE     0.5104182 0.6657731 0.4215554 0.3257751 0.4915582 0.6465394
```

Prediction

```
meanfreq <-  data.frame(t(colMeans(gasdata[,2:41])))

pred.l <-  predict(m.linear, newdata=meanfreq)
pred.aic <-  predict(m.AIC, newdata=meanfreq)
pred.pcr <-  predict(m.PCR.best, newdata=meanfreq, ncomp=4)
pred.pls <-  predict(m.pls, newdata=meanfreq, ncomp=3)
pred.ridge <-  m.ridge.best$ym +
  scale(meanfreq,center=m.ridge.best$xm,
        scale=m.ridge.best$scales)%*%m.ridge.best$coef
pred.lasso <- predict(m.lasso,as.matrix(meanfreq),s=0.05050505,mode="fraction")$fit
c(pred.l,pred.aic,pred.pcr,pred.pls,pred.ridge,pred.lasso)
```

```
       1        1
87.11089 87.13930 87.13647 87.15794 87.13247 87.16033
```