Shrinkage Methods in R

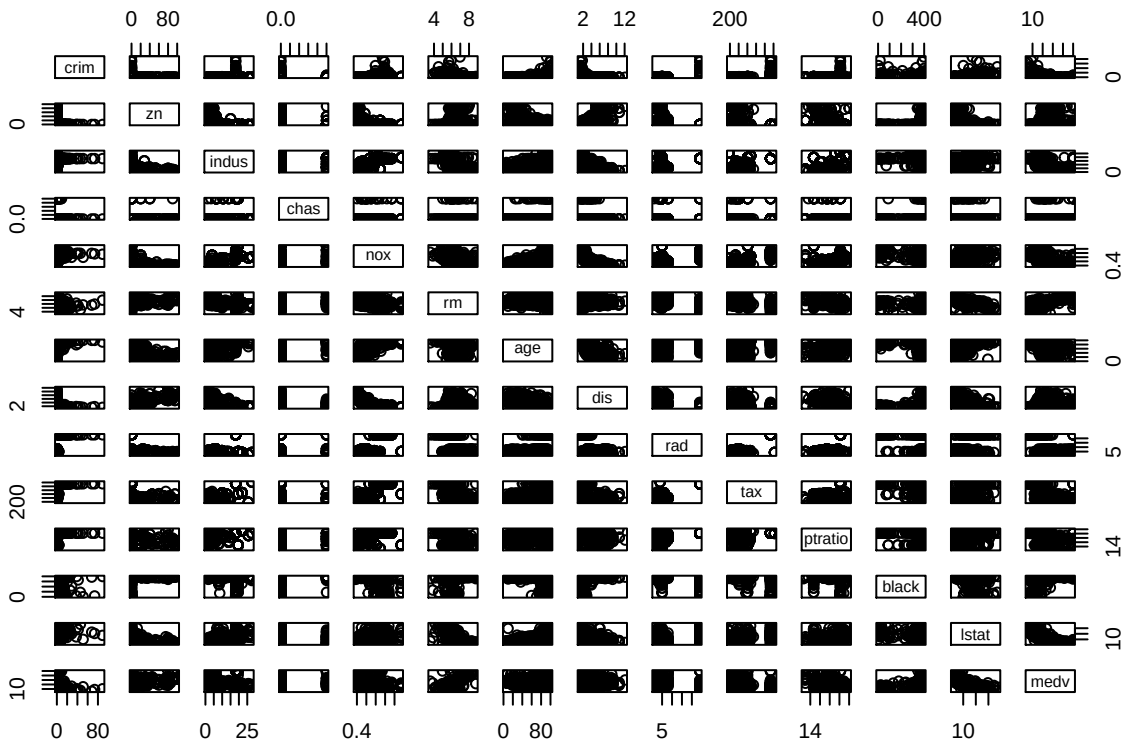
Jeremy Williams

March 22, 2018

```
##### Multiple Regression Chap 3 ISL

## Install packages
#install.packages("leaps");install.packages("car");
#install.packages("glmnet"); install.packages("plotmo")
## load libraries
suppressMessages(suppressWarnings(library(MASS)))#para el vif
suppressMessages(suppressWarnings(library(car)))#para el vif
suppressMessages(suppressWarnings(library(glmnet)))#lasso y ridge
suppressMessages(suppressWarnings(library(leaps)))#subset selection, Cp, AIC, BIC
###### Data
data("Boston")
names(Boston)
```

```
##  [1] "crim"    "zn"      "indus"   "chas"    "nox"     "rm"      "age"
##  [8] "dis"     "rad"     "tax"     "ptratio" "black"   "lstat"   "medv"
```

```
# medv=median house value
attach(Boston)
pairs(Boston)
```



```r
modelo1=lm(medv~.,data=Boston)
summary(modelo1)
```

```
##
## Call:
## lm(formula = medv ~ ., data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.595  -2.730  -0.518   1.777  26.199
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
## crim        -1.080e-01  3.286e-02  -3.287 0.001087 **
## zn           4.642e-02  1.373e-02   3.382 0.000778 ***
## indus        2.056e-02  6.150e-02   0.334 0.738288
## chas         2.687e+00  8.616e-01   3.118 0.001925 **
## nox         -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
## rm           3.810e+00  4.179e-01   9.116  < 2e-16 ***
## age          6.922e-04  1.321e-02   0.052 0.958229
## dis         -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
## rad          3.060e-01  6.635e-02   4.613 5.07e-06 ***
## tax         -1.233e-02  3.760e-03  -3.280 0.001112 **
## ptratio     -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
## black        9.312e-03  2.686e-03   3.467 0.000573 ***
## lstat       -5.248e-01  5.072e-02 -10.347  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.745 on 492 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7338
## F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

```r
vif(modelo1)
```

```
##     crim       zn    indus     chas      nox       rm      age      dis
## 1.792192 2.298758 3.991596 1.073995 4.393720 1.933744 3.100826 3.955945
##      rad      tax  ptratio    black    lstat
## 7.484496 9.008554 1.799084 1.348521 2.941491
```

```r
modelo2=update(modelo1,~.-tax)
summary(modelo2)
```

```
##
## Call:
## lm(formula = medv ~ crim + zn + indus + chas + nox + rm + age +
##     dis + rad + ptratio + black + lstat, data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.1449  -2.9143  -0.5661   1.7438  26.3113
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)  3.463e+01  5.123e+00   6.760 3.92e-11 ***
## crim        -1.067e-01  3.319e-02  -3.216 0.001384 **
## zn           3.637e-02  1.351e-02   2.692 0.007354 **
## indus       -6.778e-02  5.583e-02  -1.214 0.225317
## chas         3.029e+00  8.637e-01   3.507 0.000494 ***
## nox         -1.870e+01  3.847e+00  -4.862 1.57e-06 ***
## rm           3.912e+00  4.209e-01   9.294  < 2e-16 ***
## age         -6.054e-04  1.333e-02  -0.045 0.963804
## dis         -1.488e+00  2.014e-01  -7.390 6.31e-13 ***
## rad          1.346e-01  4.125e-02   3.262 0.001182 **
## ptratio     -9.851e-01  1.317e-01  -7.478 3.48e-13 ***
## black        9.546e-03  2.711e-03   3.521 0.000470 ***
## lstat       -5.222e-01  5.121e-02 -10.198  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.792 on 493 degrees of freedom
## Multiple R-squared:  0.735,  Adjusted R-squared:  0.7285
## F-statistic: 113.9 on 12 and 493 DF,  p-value: < 2.2e-16
```

```r
modelo3=update(modelo1,~.-age)
summary(modelo3)
```

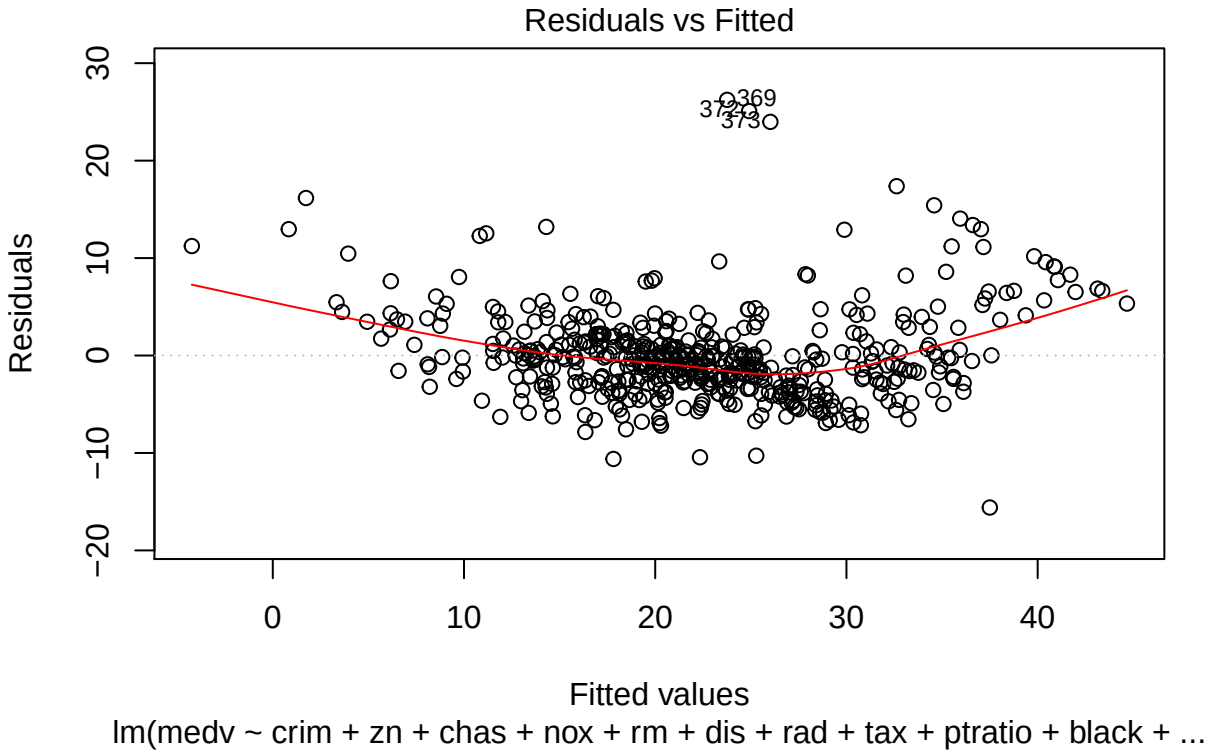
```
##
## Call:
## lm(formula = medv ~ crim + zn + indus + chas + nox + rm + dis +
##     rad + tax + ptratio + black + lstat, data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.6054  -2.7313  -0.5188   1.7601  26.2243
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  36.436927   5.080119   7.172 2.72e-12 ***
## crim         -0.108006   0.032832  -3.290 0.001075 **
## zn            0.046334   0.013613   3.404 0.000719 ***
## indus         0.020562   0.061433   0.335 0.737989
## chas          2.689026   0.859598   3.128 0.001863 **
## nox         -17.713540   3.679308  -4.814 1.97e-06 ***
## rm            3.814394   0.408480   9.338  < 2e-16 ***
## dis          -1.478612   0.190611  -7.757 5.03e-14 ***
## rad           0.305786   0.066089   4.627 4.75e-06 ***
## tax          -0.012329   0.003755  -3.283 0.001099 **
## ptratio      -0.952211   0.130294  -7.308 1.10e-12 ***
## black         0.009321   0.002678   3.481 0.000544 ***
## lstat        -0.523852   0.047625 -10.999  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.74 on 493 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7343
## F-statistic: 117.3 on 12 and 493 DF,  p-value: < 2.2e-16
```

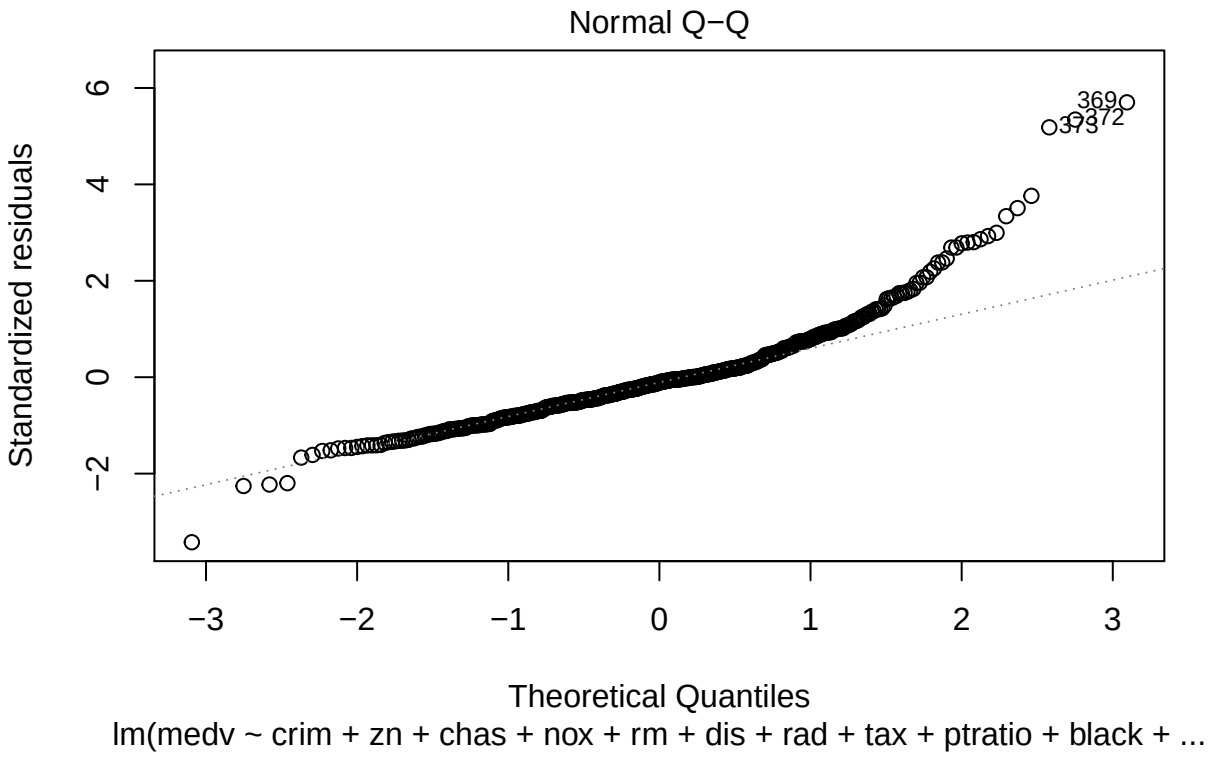
```r
saic=stepAIC(modelo1)
```

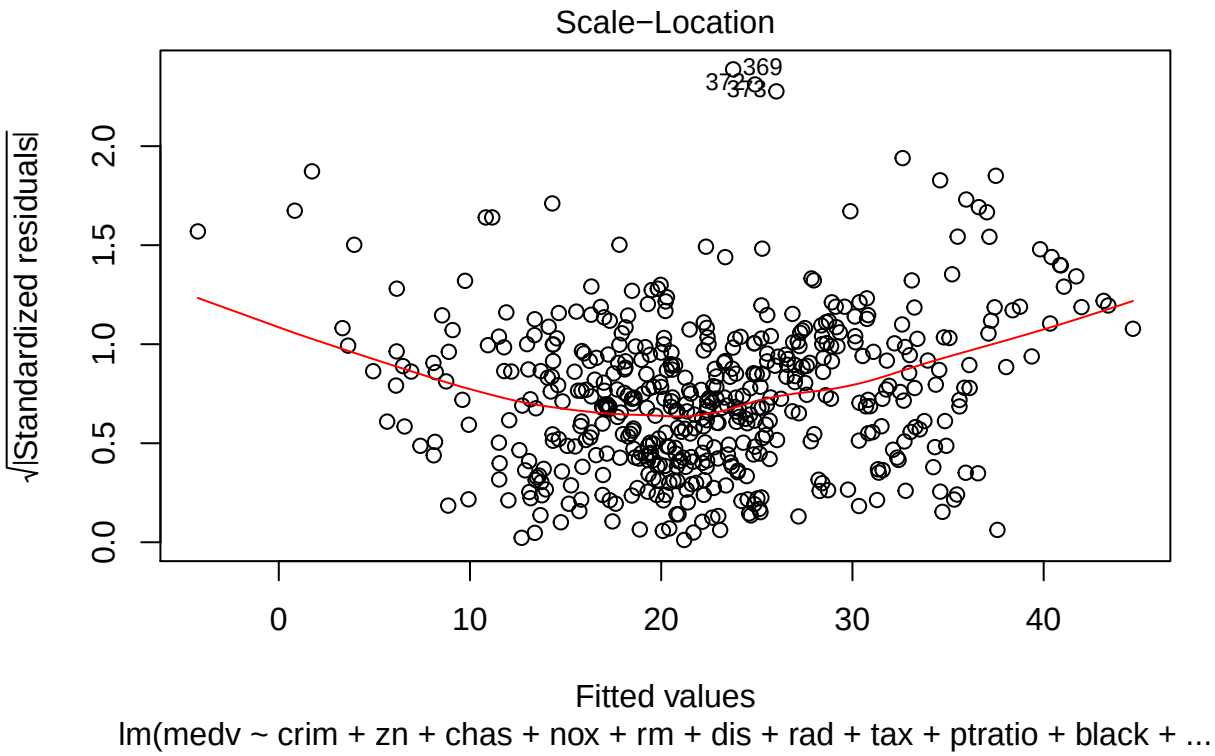
```
## Start:  AIC=1589.64
## medv ~ crim + zn + indus + chas + nox + rm + age + dis + rad +
##     tax + ptratio + black + lstat
##
##           Df Sum of Sq   RSS    AIC
## - age      1      0.06 11079 1587.7
## - indus    1      2.52 11081 1587.8
## <none>                 11079 1589.6
## - chas     1    218.97 11298 1597.5
## - tax      1    242.26 11321 1598.6
## - crim     1    243.22 11322 1598.6
## - zn       1    257.49 11336 1599.3
## - black    1    270.63 11349 1599.8
## - rad      1    479.15 11558 1609.1
## - nox      1    487.16 11566 1609.4
## - ptratio  1   1194.23 12273 1639.4
## - dis      1   1232.41 12311 1641.0
## - rm       1   1871.32 12950 1666.6
## - lstat    1   2410.84 13490 1687.3
##
## Step:  AIC=1587.65
## medv ~ crim + zn + indus + chas + nox + rm + dis + rad + tax +
##     ptratio + black + lstat
##
##           Df Sum of Sq   RSS    AIC
## - indus    1      2.52 11081 1585.8
## <none>                 11079 1587.7
## - chas     1    219.91 11299 1595.6
## - tax      1    242.24 11321 1596.6
## - crim     1    243.20 11322 1596.6
## - zn       1    260.32 11339 1597.4
## - black    1    272.26 11351 1597.9
## - rad      1    481.09 11560 1607.2
## - nox      1    520.87 11600 1608.9
## - ptratio  1   1200.23 12279 1637.7
## - dis      1   1352.26 12431 1643.9
## - rm       1   1959.55 13038 1668.0
## - lstat    1   2718.88 13798 1696.7
##
## Step:  AIC=1585.76
## medv ~ crim + zn + chas + nox + rm + dis + rad + tax + ptratio +
##     black + lstat
##
##           Df Sum of Sq   RSS    AIC
## <none>                 11081 1585.8
## - chas     1    227.21 11309 1594.0
## - crim     1    245.37 11327 1594.8
## - zn       1    257.82 11339 1595.4
## - black    1    270.82 11352 1596.0
## - tax      1    273.62 11355 1596.1
## - rad      1    500.92 11582 1606.1
## - nox      1    541.91 11623 1607.9
```

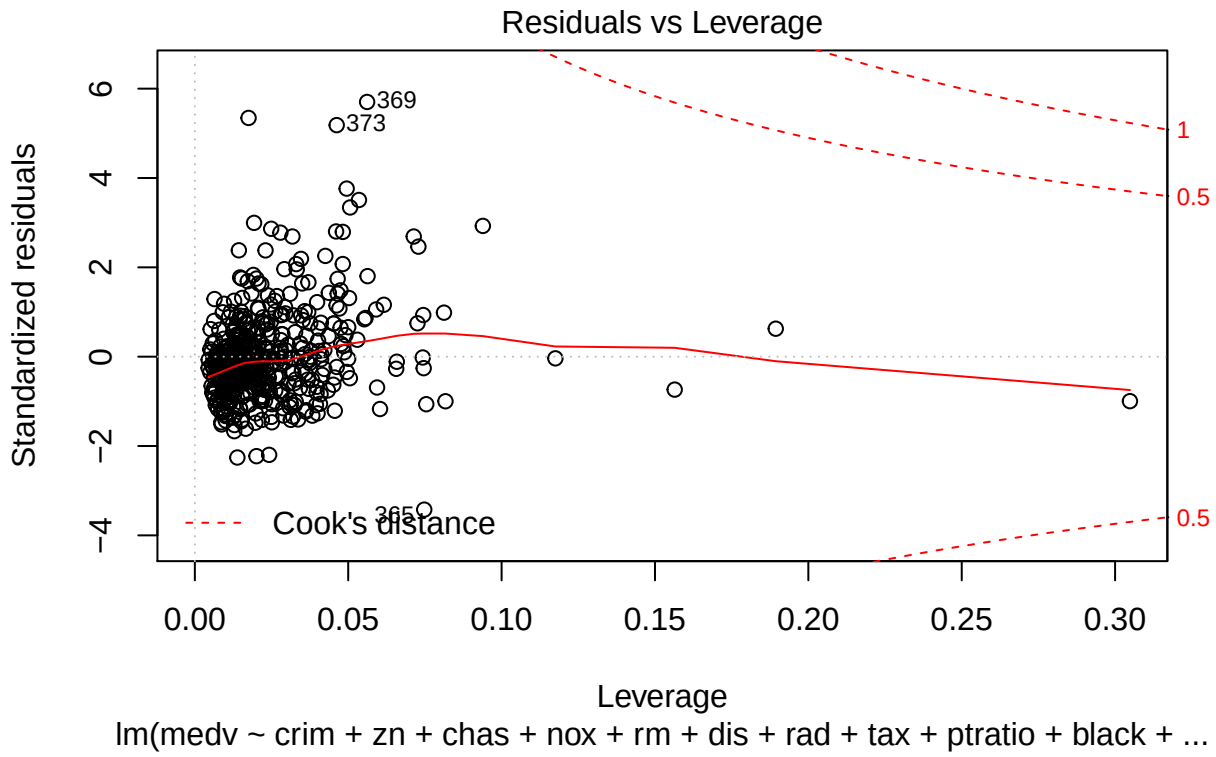
```
## - ptratio  1   1206.45 12288 1636.0
## - dis      1   1448.94 12530 1645.9
## - rm       1   1963.66 13045 1666.3
## - lstat    1   2723.48 13805 1695.0
```

```
plot(saic)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(medv ~ crim + zn + chas + nox + rm + dis + rad + tax + ptratio + black + ...
369
372
373
−3
−2
−1
0
1
2
3
−2
0
2
4
6


Scale−Location
369
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
0
10
20
30
40
Fitted values
lm(medv ~ crim + zn + chas + nox + rm + dis + rad + tax + ptratio + black + ...




```
######### seleccion de modelos
fit.full=regsubsets(medv~.,data=Boston)
summary(fit.full) #max 8 predictors , change it with nvmax=...
```

```
## Subset selection object
## Call: regsubsets.formula(medv ~ ., data = Boston)
## 13 Variables  (and intercept)
##         Forced in Forced out
## crim        FALSE      FALSE
## zn          FALSE      FALSE
## indus       FALSE      FALSE
## chas        FALSE      FALSE
## nox         FALSE      FALSE
## rm          FALSE      FALSE
## age         FALSE      FALSE
## dis         FALSE      FALSE
## rad         FALSE      FALSE
## tax         FALSE      FALSE
## ptratio     FALSE      FALSE
## black       FALSE      FALSE
## lstat       FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: exhaustive
##          crim zn  indus chas nox rm  age dis rad tax ptratio black lstat
## 1  ( 1 ) " "  " " " "   " "  " " " " " " " " " " " "     " "   "*"
## 2  ( 1 ) " "  " " " "   " "  " " "*" " " " " " " " " "     " "   "*"
```

```
## 3  ( 1 ) " "  " " " " "   " "  " " "*" " " " " " " " " " "*"     " "   "*"
## 4  ( 1 ) " "  " " " " "   " "  " " "*" " " "*" " " " " " "*"     " "   "*"
## 5  ( 1 ) " "  " " " " "   " "  "*" "*" " " "*" " " " " " "*"     " "   "*"
## 6  ( 1 ) " "  " " " " "   "*"  "*" "*" " " "*" " " " " " "*"     " "   "*"
## 7  ( 1 ) " "  " " " " "   "*"  "*" "*" " " "*" " " " " " "*"     "*"   "*"
## 8  ( 1 ) " "  "*" " " "   "*"  "*" "*" " " "*" " " " " " "*"     "*"   "*"
```

```
summary.fit.full=summary(fit.full)
names(summary.fit.full)
```

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

```
summary.fit.full$cp # proportional to AIC
```

```
## [1] 362.75295 185.64743 111.64889  91.48526  59.75364  47.17537  37.05889
## [8]  30.62398
```

```
summary.fit.full$bic
```

```
## [1] -385.0521 -496.2582 -549.4767 -561.9884 -585.6823 -592.9553 -598.2295
## [8] -600.1663
```

```
summary.fit.full$adjr2
```

```
## [1] 0.5432418 0.6371245 0.6767036 0.6878351 0.7051702 0.7123567 0.7182560
## [8] 0.7222072
```

```
## All criteria choose model 8

fit.full.larger=regsubsets(medv~.,data=Boston,nvmax=13)
summary.fit.full.larger=summary(fit.full.larger)
summary.fit.full.larger$cp  #mod 11
```

```
##  [1] 362.75295 185.64743 111.64889  91.48526  59.75364  47.17537  37.05889
##  [8]  30.62398  25.86592  18.20493  10.11455  12.00275  14.00000
```

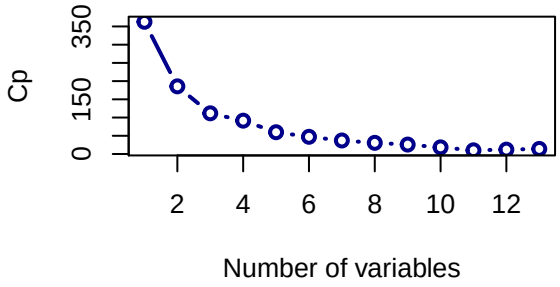
```
summary.fit.full.larger$bic #mod 11
```

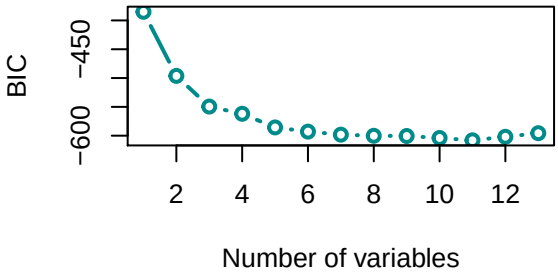
```
##  [1] -385.0521 -496.2582 -549.4767 -561.9884 -585.6823 -592.9553 -598.2295
##  [8] -600.1663 -600.5767 -603.9917 -608.0353 -601.9237 -595.7000
```

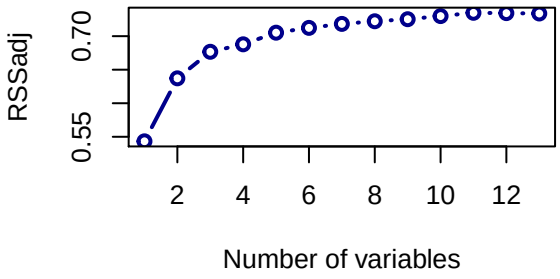
```
summary.fit.full.larger$adjr2 #mod 11
```

```
##  [1] 0.5432418 0.6371245 0.6767036 0.6878351 0.7051702 0.7123567 0.7182560
##  [8] 0.7222072 0.7252743 0.7299149 0.7348058 0.7343282 0.7337897
```

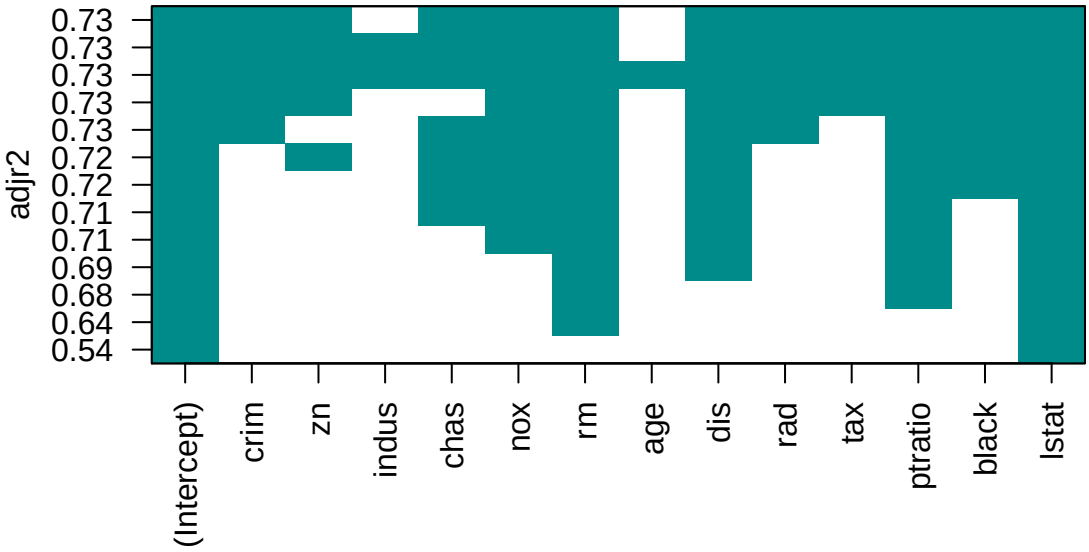
```
par(mfrow=c(2,2))
plot(summary.fit.full.larger$cp,xlab="Number of variables",ylab="Cp", type="b",col="darkblue",lwd=2)
plot(summary.fit.full.larger$bic,xlab="Number of variables",ylab="BIC", type="b",col="darkcyan",lwd=2)
plot(summary.fit.full.larger$adjr2,xlab="Number of variables",ylab="RSSadj", type="b",col="darkblue",lw

### a particular plot from regsubsets:
par(mfrow=c(1,1))
```



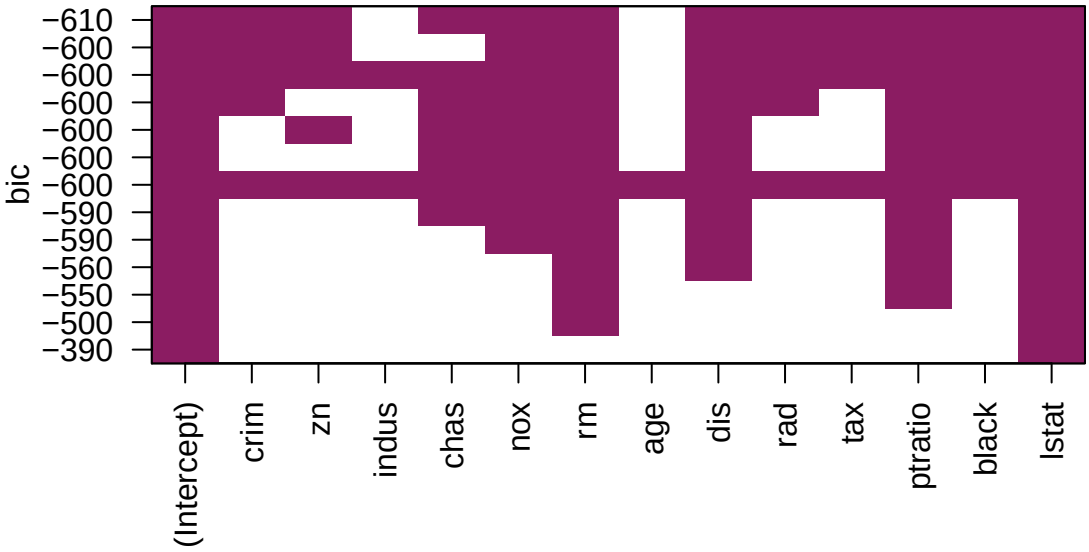




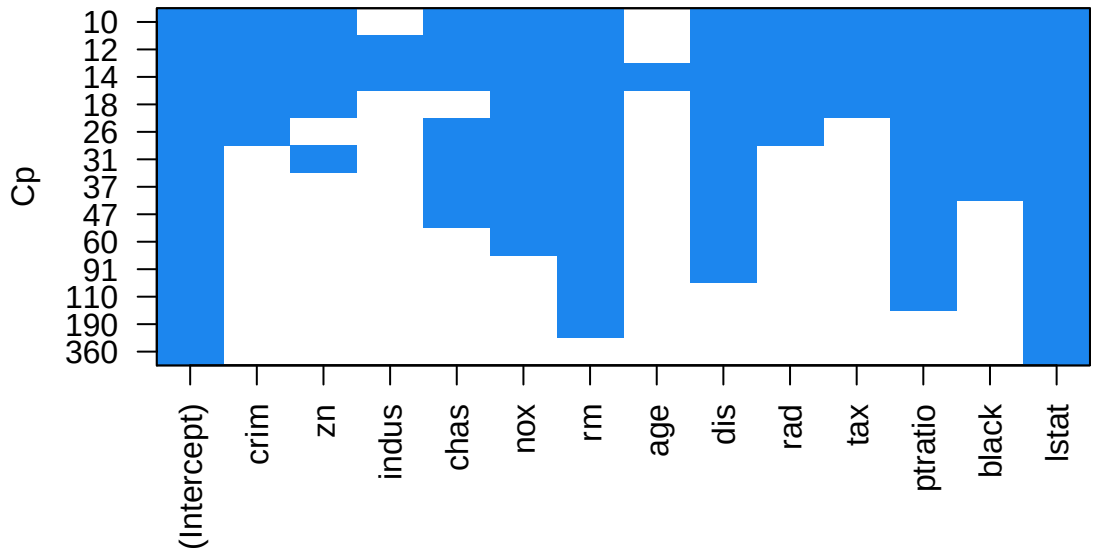

```
#squares indicate which variable is present
plot(fit.full.larger, scale="adjr2", col="darkcyan")
```




```
plot(fit.full.larger, scale="bic", col="maroon4")
```




```
plot(fit.full.larger, scale="Cp", col="dodgerblue2")
```




```
###### forward selection
forward=regsubsets(medv~.,data=Boston,nvmax=13,method="forward")
summary(forward)
```

```
## Subset selection object
## Call: regsubsets.formula(medv ~ ., data = Boston, nvmax = 13, method = "forward")
## 13 Variables  (and intercept)
##         Forced in Forced out
## crim        FALSE      FALSE
## zn          FALSE      FALSE
## indus       FALSE      FALSE
## chas        FALSE      FALSE
## nox         FALSE      FALSE
## rm          FALSE      FALSE
## age         FALSE      FALSE
## dis         FALSE      FALSE
## rad         FALSE      FALSE
## tax         FALSE      FALSE
## ptratio     FALSE      FALSE
## black       FALSE      FALSE
## lstat       FALSE      FALSE
## 1 subsets of each size up to 13
## Selection Algorithm: forward
##          crim zn  indus chas nox rm  age dis rad tax ptratio black lstat
## 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 ) "*"  "*" "*" "   "*"  "*" "*" "*" "*" "*" "*" "*"     "*"   "*"
```

```
######### how do we choose????? Cross-validation (later)

########## LASSO  lambda is chosen wirh CV as well...  train/validate
## in order to split the data, put iyt in vector Y matrix X
library(plotmo) # for plot_glmnet
```

```
## Warning: package 'plotmo' was built under R version 3.4.4
```

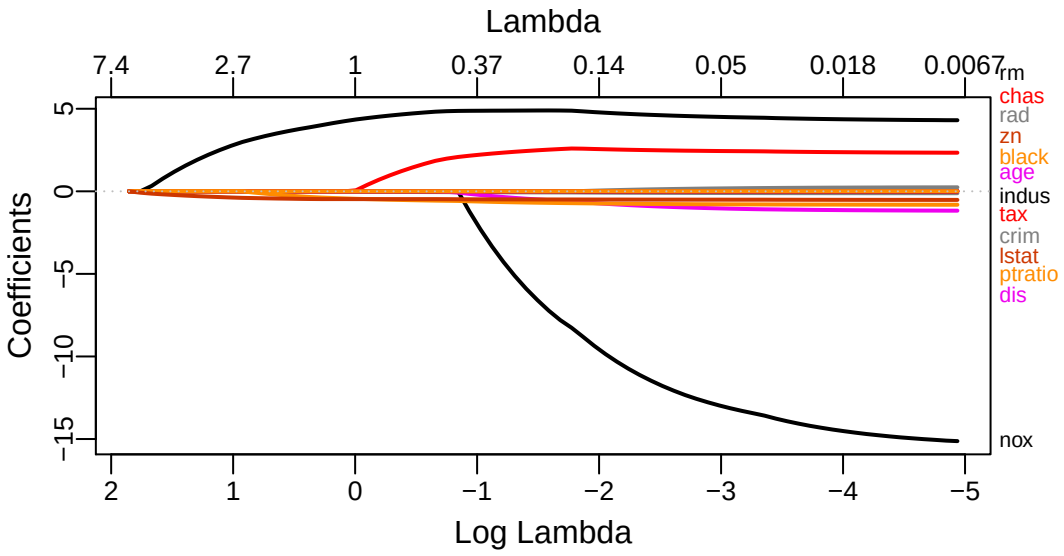
```
## Loading required package: plotrix
```

```
## Warning: package 'plotrix' was built under R version 3.4.3
```

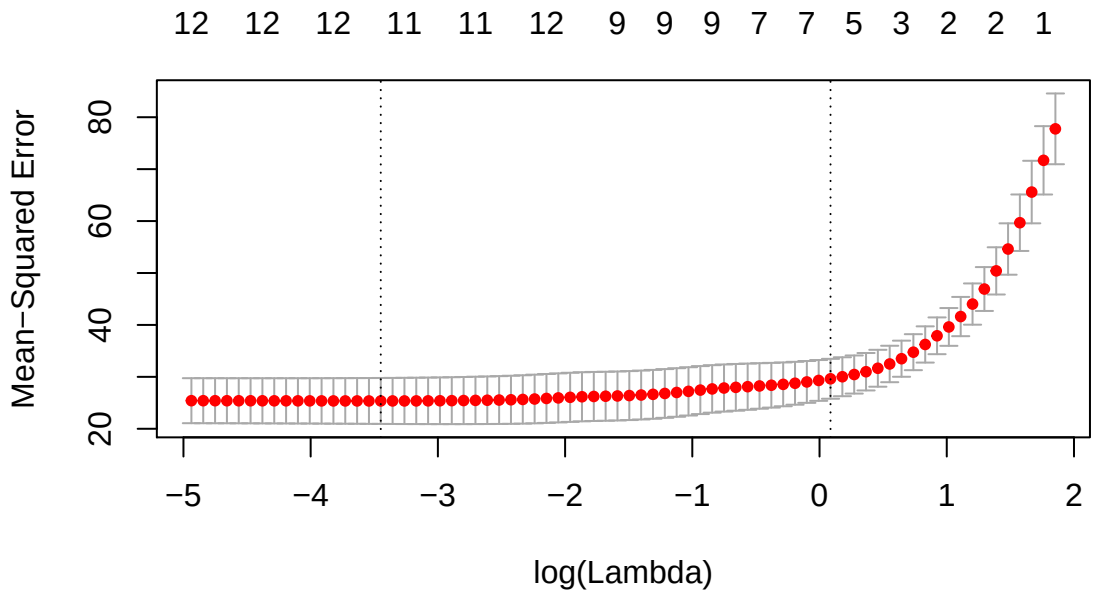
```
## Loading required package: TeachingDemos
```

```
## Warning: package 'TeachingDemos' was built under R version 3.4.4
```

```
set.seed(115)
x=model.matrix(medv~.,data=Boston)[,-1]
y=Boston$medv
train=sample(1:nrow(x),nrow(x)/2)
test=(-train)
y.test=y[test]
lasso.mod=glmnet(x[train,], y[train],alpha=1,standardize=TRUE)
plot_glmnet(lasso.mod, label=TRUE,lwd=2)
```




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




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

```
## [1] 0.03177962
```

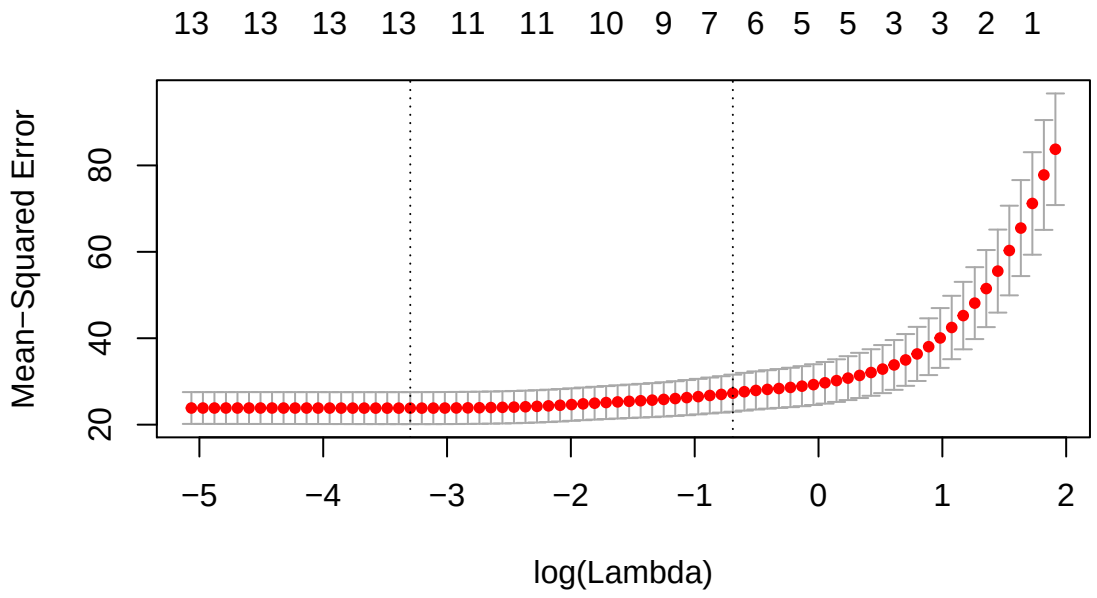
```
lasso.pred=predict (lasso.mod ,s=bestlam ,newx=x[test ,])
mean(( lasso.pred -y.test)^2)
```

```
## [1] 23.53327
```

```
out=glmnet (x,y,alpha =1) #### with all data
lasso.coef=predict (out,type="coefficients",s=bestlam )[1:13,]
lasso.coef
```

```
##   (Intercept)          crim            zn         indus          chas
##  34.177888312  -0.097056024   0.040871485   0.000000000   2.680208281
##           nox            rm           age           dis           rad
## -16.165836825   3.875224152   0.000000000  -1.383325999   0.246674670
##           tax       ptratio         black
##  -0.009576472  -0.927822563   0.008990665
```

```
#### other training proportion
train2=sample(1:nrow(x),2*nrow(x)/3)
test2=(-train2)
y2.test=y[test2]
cv.out =cv.glmnet (x[train2,],y[train2],alpha =1)
plot(cv.out)
```




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

```
## [1] 0.03699728
```

```
out=glmnet (x,y,alpha =1) #### with all data
lasso.coef=predict (out,type="coefficients",s=bestlam )[1:13,]
lasso.coef
```

```
##   (Intercept)          crim            zn         indus          chas
##  33.831720064  -0.095255854   0.040075335   0.000000000   2.673437201
##           nox            rm           age           dis           rad
## -15.969813731   3.886800185   0.000000000  -1.365652401   0.238293199
##           tax       ptratio         black
##  -0.009227547  -0.924818270   0.008941939
```

```
################ mtcars  (far away from "big" data)
data(mtcars)
correlations=cor(mtcars)
round(correlations,2)
```

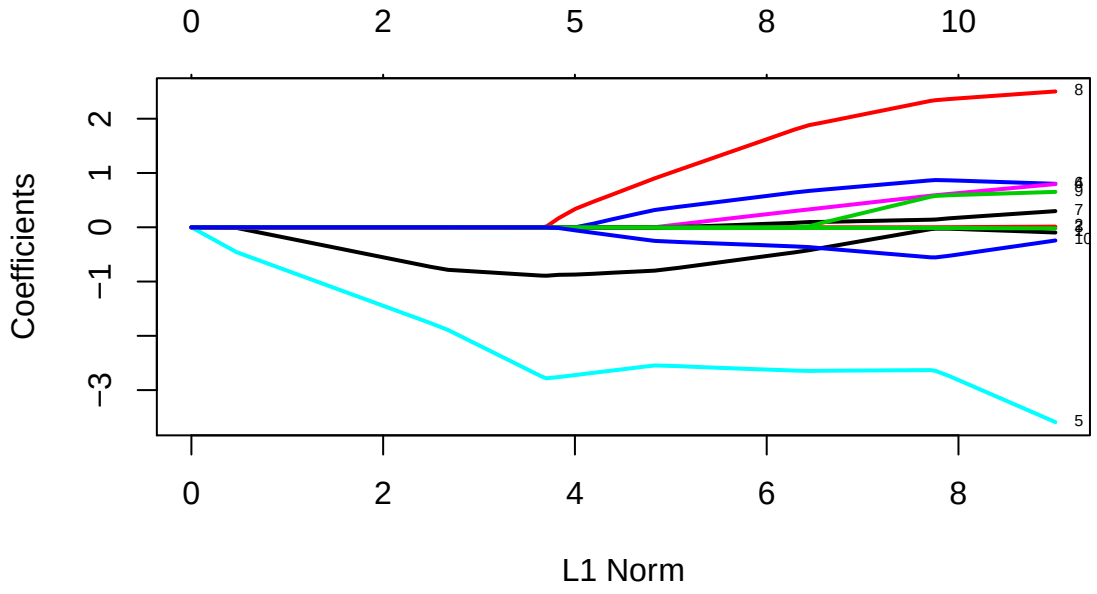
```
##        mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
## mpg   1.00 -0.85 -0.85 -0.78  0.68 -0.87  0.42  0.66  0.60  0.48 -0.55
## cyl  -0.85  1.00  0.90  0.83 -0.70  0.78 -0.59 -0.81 -0.52 -0.49  0.53
## disp -0.85  0.90  1.00  0.79 -0.71  0.89 -0.43 -0.71 -0.59 -0.56  0.39
## hp   -0.78  0.83  0.79  1.00 -0.45  0.66 -0.71 -0.72 -0.24 -0.13  0.75
## drat  0.68 -0.70 -0.71 -0.45  1.00 -0.71  0.09  0.44  0.71  0.70 -0.09
## wt   -0.87  0.78  0.89  0.66 -0.71  1.00 -0.17 -0.55 -0.69 -0.58  0.43
## qsec  0.42 -0.59 -0.43 -0.71  0.09 -0.17  1.00  0.74 -0.23 -0.21 -0.66
```

```
## vs    0.66 -0.81 -0.71 -0.72  0.44 -0.55  0.74  1.00  0.17  0.21 -0.57
## am    0.60 -0.52 -0.59 -0.24  0.71 -0.69 -0.23  0.17  1.00  0.79  0.06
## gear  0.48 -0.49 -0.56 -0.13  0.70 -0.58 -0.21  0.21  0.79  1.00  0.27
## carb -0.55  0.53  0.39  0.75 -0.09  0.43 -0.66 -0.57  0.06  0.27  1.00
```

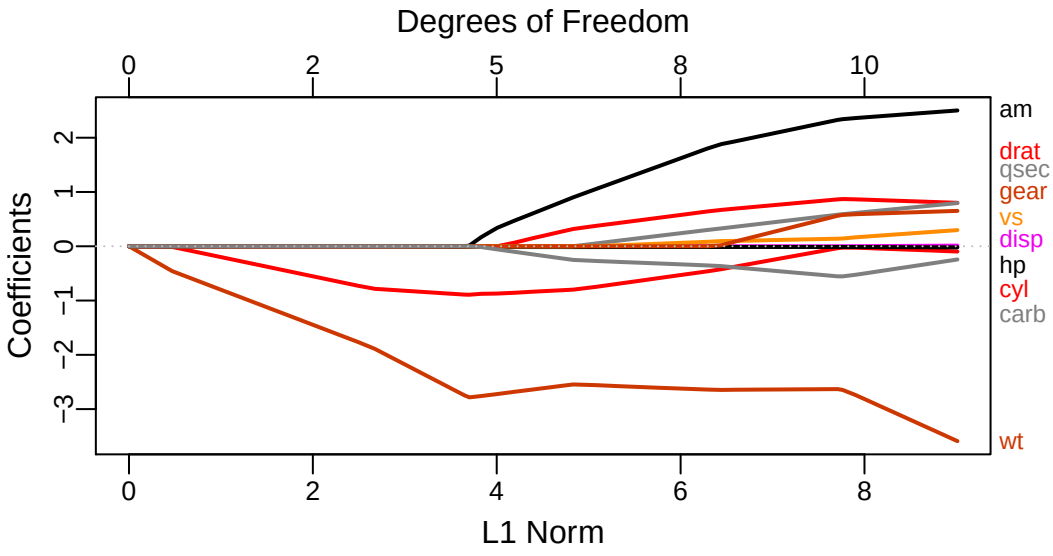
```
modelo=lm(mpg~., data=mtcars)
anova(modelo)
```

```
## Analysis of Variance Table
##
## Response: mpg
##           Df Sum Sq Mean Sq  F value    Pr(>F)
## cyl        1 817.71  817.71 116.4245 5.034e-10 ***
## disp       1  37.59   37.59   5.3526  0.030911 *
## hp         1   9.37    9.37   1.3342  0.261031
## drat       1  16.47   16.47   2.3446  0.140644
## wt         1  77.48   77.48  11.0309  0.003244 **
## qsec       1   3.95    3.95   0.5623  0.461656
## vs         1   0.13    0.13   0.0185  0.893173
## am         1  14.47   14.47   2.0608  0.165858
## gear       1   0.97    0.97   0.1384  0.713653
## carb       1   0.41    0.41   0.0579  0.812179
## Residuals 21 147.49    7.02
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

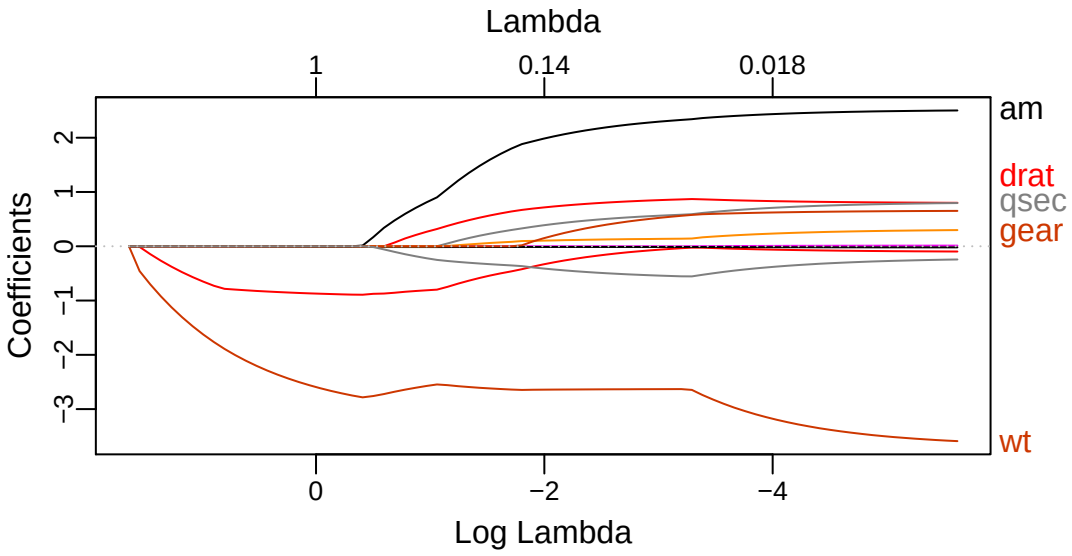
```
x=as.matrix(mtcars[-1])
y=mtcars[,1]
mod=glmnet(x,y,standardize=T,alpha=1)
plot.glmnet(mod,label=T,lwd=2)
```



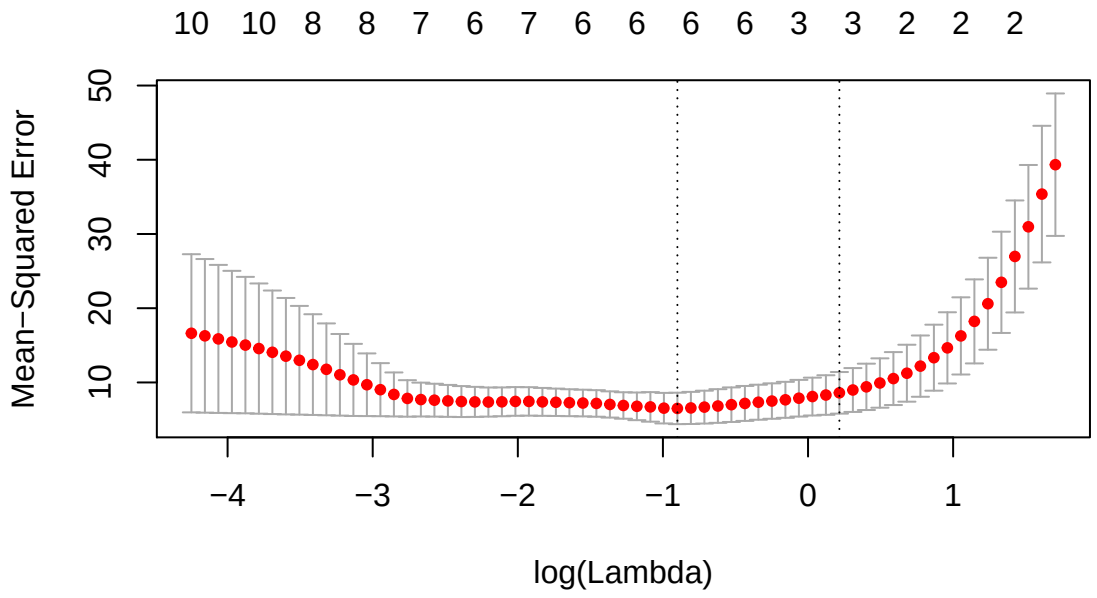

```
mod=glmnet(as.matrix(x),y,standardize=T,alpha=1)
plot_glmnet(mod, label=T, lwd=2, xvar="norm")                    # default colors
```




```
plot_glmnet(mod, label=5)                    # label the 5 biggest final coefs
```




```
######choosing lambda

train=sample(1:nrow(x),nrow(x)*0.66)
test=(-train)
y.test=y[test]
cv.out =cv.glmnet (as.matrix(x[train,]),y[train],alpha =1,nfold=5)
plot(cv.out) # dotted line on the left min cv-error, right error within 1 stdev from min
```




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

```
## [1] 0.4064773
```

```
out=glmnet(as.matrix(x),y,alpha =1) #### with all data, no standarization
lasso.coef=predict (out,type="coefficients",s=bestlam )[1:11,]
lasso.coef
```

```
## (Intercept)         cyl        disp          hp        drat          wt
## 35.08277506 -0.82025505  0.00000000 -0.01446991  0.22508917 -2.59677886
##        qsec          vs          am        gear        carb
##  0.00000000  0.00000000  0.73776797  0.00000000 -0.19521627
```

```
######################## large p
set.seed(19874)
n <- 1000    # Number of observations
p <- 5000     # Number of predictors included in model
real_p <- 1500  # Number of true predictors
x <- matrix(rnorm(n*p), nrow=n, ncol=p)
y <- apply(x[,1:real_p], 1, sum) + rnorm(n)

# Split data into train and test sets
train_rows <- sample(1:n, .66*n)
x.train <- x[train_rows, ]
x.test <- x[-train_rows, ]

y.train <- y[train_rows]
y.test <- y[-train_rows]
```

```
fit.lasso <- glmnet(x.train, y.train, family="gaussian", alpha=1)
fit.lasso.cv <- cv.glmnet(x.train, y.train, type.measure="mse", alpha=1,family="gaussian")
bestlam =fit.lasso.cv$lambda.min;bestlam
```

```
## [1] 2.688836
```

```
out=glmnet (x,y,alpha =1) #### with all data
lasso.coef=predict (out,type="coefficients",s=bestlam )[1:5000,]
length(lasso.coef[lasso.coef!=0])
```

```
## [1] 138
```