Cross validation and model selection

Data Mining
Master Degree in Computer Science
University of Padova

a.y. 2017/2018

Annamaria Guolo

1 Boston dataset

Consider `Boston` dataset in library `MASS`, the dataset already used in lab of linear regression model.

```
library(MASS)
data(Boston)
Boston[1,]

##      crim zn indus chas   nox    rm  age  dis rad tax ptratio black lstat medv
## 1 0.00632 18  2.31    0 0.538 6.575 65.2 4.09   1 296    15.3 396.9  4.98   24
```

One of the fitted model specified a quadratic relationship between the median price of the houses `medv` and the percentage of lower status of the population `lstat`

```
m2 <- lm(medv ~ lstat + I(lstat^2), data=Boston)
summary(m2)

##
## Call:
## lm(formula = medv ~ lstat + I(lstat^2), data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.2834  -3.8313  -0.5295   2.3095  25.4148
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 42.862007   0.872084   49.15   <2e-16 ***
## lstat       -2.332821   0.123803  -18.84   <2e-16 ***
## I(lstat^2)   0.043547   0.003745   11.63   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.524 on 503 degrees of freedom
## Multiple R-squared:  0.6407,Adjusted R-squared:  0.6393
## F-statistic: 448.5 on 2 and 503 DF,  p-value: < 2.2e-16
```

A we saw during classes, we can evaluate whether increasing the degree of the polynomial is a reasonable choice. To this purpose we can use cross-validation, AIC, $\overline{R}^2$.
Model with polynomial of third degree

```
m3 <- lm(medv ~ poly(lstat, 3), data=Boston)
```

Function `poly()` creates a polynomial with the desired degree

```
summary(m3)

##
## Call:
## lm(formula = medv ~ poly(lstat, 3), data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.5441  -3.7122  -0.5145   2.4846  26.4153
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       22.5328     0.2399  93.937  < 2e-16 ***
## poly(lstat, 3)1 -152.4595     5.3958 -28.255  < 2e-16 ***
## poly(lstat, 3)2   64.2272     5.3958  11.903  < 2e-16 ***
## poly(lstat, 3)3  -27.0511     5.3958  -5.013 7.43e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.396 on 502 degrees of freedom
## Multiple R-squared:  0.6578,Adjusted R-squared:  0.6558
## F-statistic: 321.7 on 3 and 502 DF,  p-value: < 2.2e-16
```

Comparison between `m2` and `m3`:

- Comparison using F statistic.

```r
anova(m2,m3)

## Analysis of Variance Table
##
## Model 1: medv ~ lstat + I(lstat^2)
## Model 2: medv ~ poly(lstat, 3)
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)
## 1    503 15347
## 2    502 14616  1    731.76 25.134 7.428e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

How can we comment on the result?

- Comparison using $\overline{R}^2$.
  From the outputs of the two models we can conclude that $\overline{R}^2$ increases from 0.6392883 for `m2` to 0.6558029 for `m3`.

```r
## extract the value from the summary of the model
summary(m2)$adj.r.squared

## [1] 0.6392883

summary(m3)$adj.r.squared

## [1] 0.6558029
```

  Thus it is justified moving to `m3`.

- Comparison using AIC.
  Compute the values of AIC for the two models

```r
## extract the value of the log-likelihood
logLik(m2)

## 'log Lik.' -1581.258 (df=4)

aic.m2 <- 2*2 - 2*logLik(m2)
aic.m2

## 'log Lik.' 3166.516 (df=4)

logLik(m3)
```

```
## 'log Lik.' -1568.898 (df=5)

aic.m3 <- 2*3 - 2*logLik(m3)
aic.m3

## 'log Lik.' 3143.796 (df=5)
```

The reduction of AIC value suggests it is preferable to move to `m3`. Using `R` functionalities, consider that function `glm()` provides the value of AIC. In order to estimate a linear regression model with `glm()` we do not have to specify option `family`

```
m2.glm <- glm(medv ~ lstat + I(lstat^2), data=Boston)
m3.glm <- glm(medv ~ lstat + I(lstat^2) + I(lstat^3), data=Boston)
```

Then, to extract the value of AIC

```
m2.glm$aic - m3.glm$aic

## [1] 22.72039

## or
extractAIC(m2) - extractAIC(m3)

## [1] -1.00000 22.72039
```

- Calculate the test error rate using CV.
  The function performing cross validation is inside library `boot`

```
library(boot)
```

Compute the *k*-fold CV, with $k = 10$

```
## fix the seed
set.seed(123)
cv.err.m2 <- cv.glm(Boston, m2.glm, K=10)
names(cv.err.m2)

## [1] "call"  "K"     "delta" "seed"
```

Function `cv.glm` calculates the estimate of the prediction error for models belonging to the `glm` class. The output contains the value of *k*, the estimate of the error and the list of seeds used.

```
cv.err.m2$delta

## [1] 30.64750 30.63077
```

The second number is the estimate corrected for the possible bias.

```
##comparison with m3.glm
## fix the seed
set.seed(123)
cv.err.m3 <- cv.glm(Boston, m3.glm, K=10)
cv.err.m3$delta

## [1] 29.32153 29.29819
```

The value of MSE provided by cross validation suggests to choose model `m3`. The LOOCV estimate is obtained without specifying K in the calling

```
cv.glm(Boston, m2.glm)$delta

## [1] 30.73622 30.73581

cv.glm(Boston, m3.glm)$delta

## [1] 29.42262 29.42207
```

We confirm the previous result. Note that in this case we don't need to fix the seed as the procedure uses all the data, inserting one observation at a time in the test set.

Compare the models with polynomial in `lstat` up to degree 6: the comparison can be performed in terms of $\overline{R}^2$ and AIC, in sequence. Hereafter, you can find the code useful to obtain the results quickly...but the same results can be obtained one at a time as done previously when comparing `m2` e `m3`

```
## Values of adjusted r2 and AIC for models with polynomials
## from degree 1 to degree 6
## Construct 2 vectors, at the moment with elements equal to 0
## They will contain the estimated values
adj.r2 <- rep(0, 6)
aic <- rep(0, 6)
## For each degree of the polynomial fit the model and
## save the corresponding value of adj.r2 and AIC
for(i in 1:6){
        m <- lm(medv ~ poly(lstat, i), data=Boston)
        adj.r2[i] <- summary(m)$adj.r.squared
        aic[i] <- 2*(i+2) - 2*logLik(m)
}
adj.r2

## [1] 0.5432418 0.6392883 0.6558029 0.6704019 0.6785066 0.6788660

aic

## [1] 3288.975 3170.516 3147.796 3126.856 3115.247 3115.668
```

The procedure suggests to choose the model with the polynomial of degree 6 on the basis of $\overline{R}^2$ and the model with the polynomial of degree 5 on the basis of AIC. Compare the models using CV.

```
m5.glm <- glm(medv ~ poly(lstat, 5), data=Boston)
m6.glm <- glm(medv ~ poly(lstat, 6), data=Boston)
set.seed(123)
cv.err.m5 <- cv.glm(Boston, m5.glm, K=10)$delta
cv.err.m5

## [1] 27.77439 27.72378

set.seed(123)
cv.err.m6 <- cv.glm(Boston, m6.glm, K=10)$delta
cv.err.m6

## [1] 27.73294 27.68008
```

Using CV, the best choice seems to be the model with the polynomial of degree 5, although the difference between the two models is slight. Given the slight difference, the model with the polynomial of degree 5 is preferable in terms of simplicity, without a substantial loss of performance.

2 Hitters dataset

The dataset is in library `ISLR` and they refer to information about Major League Baseball from the 1986 and 1987 seasons.

```
library(ISLR)
data(Hitters)
dim(Hitters)

## [1] 322  20
```

```
names(Hitters)

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

The aim is to evaluate and predict the `Salary` (in thousands of dollars) of the players using information about their performance in the previous seasons.
A first look at the data shows that there missing data, indicated as `NA`

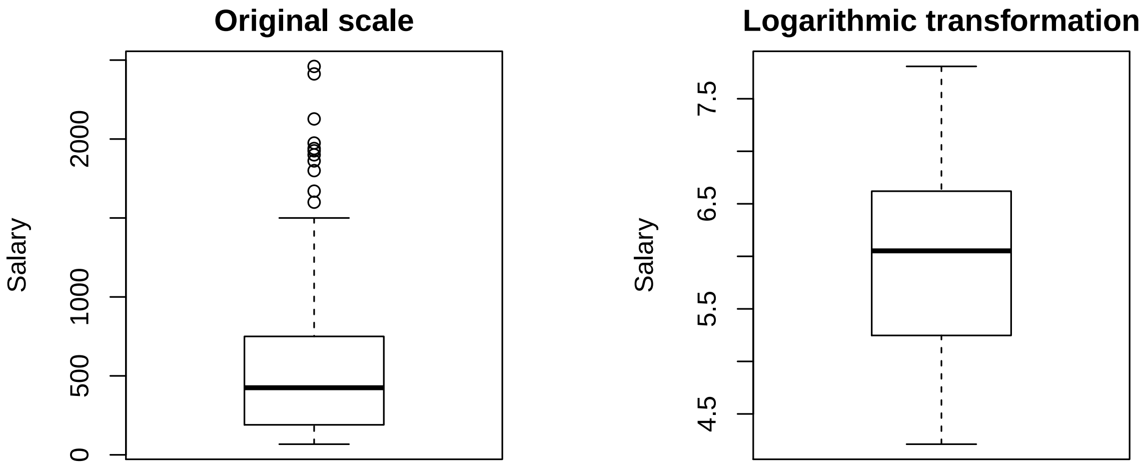
```
summary(Hitters)

##      AtBat            Hits         HmRun            Runs             RBI
##  Min.   : 16.0   Min.   :  1   Min.   : 0.00   Min.   :  0.00   Min.   :  0.00
##  1st Qu.:255.2   1st Qu.: 64   1st Qu.: 4.00   1st Qu.: 30.25   1st Qu.: 28.00
##  Median :379.5   Median : 96   Median : 8.00   Median : 48.00   Median : 44.00
##  Mean   :380.9   Mean   :101   Mean   :10.77   Mean   : 50.91   Mean   : 48.03
##  3rd Qu.:512.0   3rd Qu.:137   3rd Qu.:16.00   3rd Qu.: 69.00   3rd Qu.: 64.75
##  Max.   :687.0   Max.   :238   Max.   :40.00   Max.   :130.00   Max.   :121.00
##
##      Walks            Years            CAtBat            CHits            CHmRun
##  Min.   :  0.00   Min.   : 1.000   Min.   :   19.0   Min.   :   4.0   Min.   :  0.00
##  1st Qu.: 22.00   1st Qu.: 4.000   1st Qu.:  816.8   1st Qu.: 209.0   1st Qu.: 14.00
##  Median : 35.00   Median : 6.000   Median : 1928.0   Median : 508.0   Median : 37.50
##  Mean   : 38.74   Mean   : 7.444   Mean   : 2648.7   Mean   : 717.6   Mean   : 69.49
##  3rd Qu.: 53.00   3rd Qu.:11.000   3rd Qu.: 3924.2   3rd Qu.:1059.2   3rd Qu.: 90.00
##  Max.   :105.00   Max.   :24.000   Max.   :14053.0   Max.   :4256.0   Max.   :548.00
##
##      CRuns             CRBI             CWalks        League  Division    PutOuts
##  Min.   :   1.0   Min.   :   0.00   Min.   :   0.00   A:175   E:157    Min.   :   0.0
##  1st Qu.: 100.2   1st Qu.:  88.75   1st Qu.:  67.25   N:147   W:165    1st Qu.: 109.2
##  Median : 247.0   Median : 220.50   Median : 170.50                    Median : 212.0
##  Mean   : 358.8   Mean   : 330.12   Mean   : 260.24                    Mean   : 288.9
##  3rd Qu.: 526.2   3rd Qu.: 426.25   3rd Qu.: 339.25                    3rd Qu.: 325.0
##  Max.   :2165.0   Max.   :1659.00   Max.   :1566.00                    Max.   :1378.0
##
##     Assists          Errors          Salary       NewLeague
##  Min.   :  0.0   Min.   : 0.00   Min.   :  67.5   A:176
##  1st Qu.:  7.0   1st Qu.: 3.00   1st Qu.: 190.0   N:146
##  Median : 39.5   Median : 6.00   Median : 425.0
##  Mean   :106.9   Mean   : 8.04   Mean   : 535.9
##  3rd Qu.:166.0   3rd Qu.:11.00   3rd Qu.: 750.0
##  Max.   :492.0   Max.   :32.00   Max.   :2460.0
##                                  NA's   :59
```

How to handle missing data is outside the aim of the course: thus, we will simply eliminate the missing data and we will work with the complete rows of the dataset.

```
## how many missing data?
sum(is.na(Hitters))

## [1] 59

## clean the dataset
hitters <- na.omit(Hitters)
dim(hitters)

## [1] 263  20

sum(is.na(hitters))

## [1] 0

## ok!
```

Preliminary graphical analysis of the response variable

```
par(mfrow=c(1,2))
boxplot(hitters$Salary, ylab='Salary', main='Original scale')
boxplot(log(hitters$Salary), ylab='Salary', main='Logarithmic transformation')
```



Choose the logarithmic transformation.

```r
hitters$Salary <- log(hitters$Salary)
```

Given the large number of variables, we can try with an automatic variable selection approach. Start with the *forward regression*. Function `regsubsets()` implemented in library `leaps` allows to apply stepwise regression for linear models using different criteria using a pre-specified number of covariates. Syntax is similar to that used in `lm()`.

```r
library(leaps)
m.forward <- regsubsets(Salary ~ ., data=hitters, nvmax=19, method='forward')
```

The option `nvmax` specifies the maximum number of covariates included in the model. Default to 8.

```r
summary(m.forward)

## 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 CRBI CWalks
## 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 ) "*"   "*"  "*"   "*"  "*" "*"   "*"   "*"    "*"   "*"    "*"   "*"  "*"
##           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 ) "*"     "*"       "*"     "*"     "*"    "*"
```

The output shows the best models with a fixed number of covariates on the basis of RSS. The variables chosen in each model are identified through an asterisk. For example, the best model with 3 covariates includes `Hits`, `CRBI` and `PutOuts`. The `summary` of the object contains all the following information

```
names(summary(m.forward))
```

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

In particular, it contains the values of RSS, R^2, $\overline{R}^2$ and BIC for the fitted models.

```
summary(m.forward)$rss

##  [1] 127.24377 107.31192 104.19696 101.89625  99.82172  98.31075  96.38157  94.52650
##  [9]  93.75989  93.18646  92.70504  92.43868  91.62009  91.57899  91.51304  91.49201
## [17]  91.46237  91.44653  91.44630

which.min(summary(m.forward)$rss)

## [1] 19
```

The model with smallest RSS is the model number 19, that is, the model with the following covariates

```
coef(m.forward, 19)

##   (Intercept)         AtBat          Hits         HmRun          Runs           RBI
##  4.618143e+00 -2.983835e-03  1.308450e-02  1.179338e-02 -1.419299e-03 -1.675456e-03
##         Walks         Years        CAtBat         CHits        CHmRun         CRuns
##  1.095506e-02  5.696428e-02  1.282973e-04 -4.413856e-04 -7.808898e-05  1.512926e-03
##          CRBI        CWalks       LeagueN     DivisionW       PutOuts       Assists
##  1.311826e-04 -1.465848e-03  2.824751e-01 -1.656435e-01  3.389042e-04  6.213909e-04
##        Errors    NewLeagueN
## -1.196690e-02 -1.741606e-01
```

all the available covariates. Using BIC, instead, the best model includes the following covariates

```
which.min(summary(m.forward)$bic)

## [1] 4

coef(m.forward, 4)

## (Intercept)        Hits       Years       CRuns     PutOuts
## 4.432357335 0.006780439 0.053285280 0.000753155 0.000351109
```

Variance/covariance matrix of the estimators

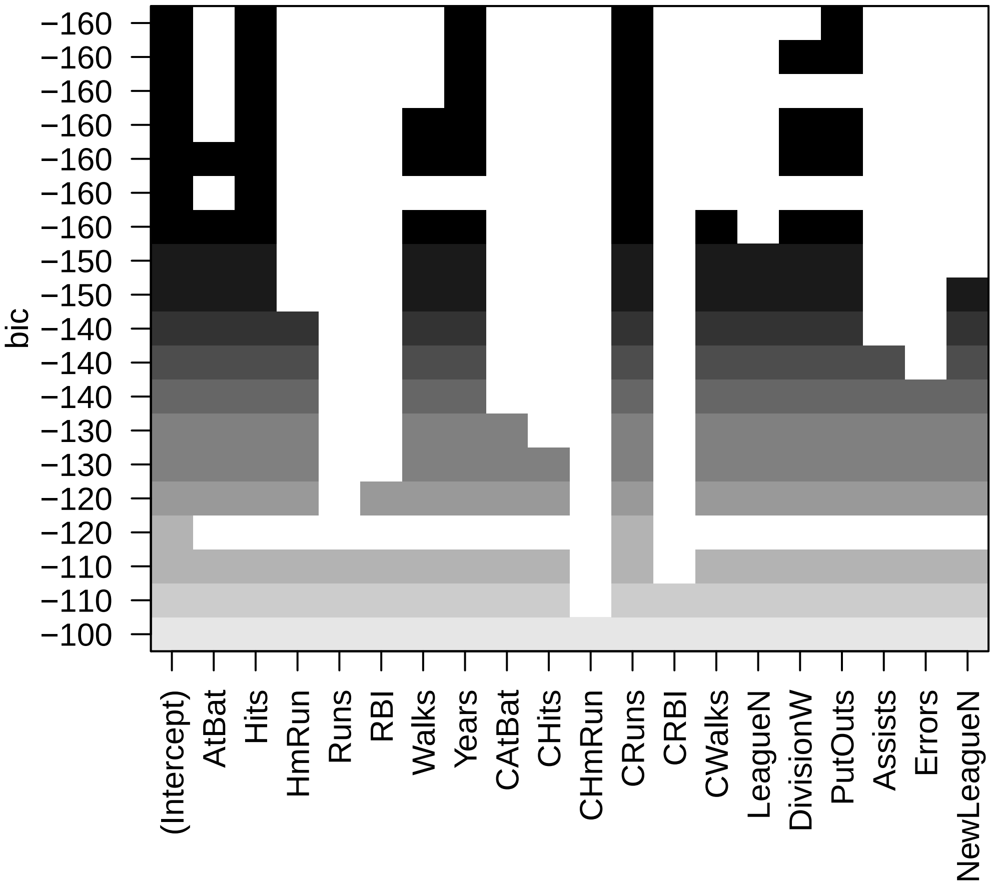
```
vcov(m.forward, 4)

##               (Intercept)          Hits         Years         CRuns       PutOuts
## (Intercept)  1.862590e-02 -1.062232e-04 -1.640705e-03  1.943002e-05 -2.381408e-06
## Hits        -1.062232e-04  1.009909e-06  7.151825e-06 -1.217265e-07 -3.782685e-08
## Years       -1.640705e-03  7.151825e-06  3.405174e-04 -4.557830e-06  8.985931e-08
## CRuns        1.943002e-05 -1.217265e-07 -4.557830e-06  7.558336e-08 -9.657864e-10
## PutOuts     -2.381408e-06 -3.782685e-08  8.985931e-08 -9.657864e-10  2.116213e-08
```

Standard error

```
sqrt(diag(vcov(m.forward, 4)))

##  (Intercept)         Hits        Years        CRuns      PutOuts
## 0.1364767271 0.0010049422 0.0184531134 0.0002749243 0.0001454721
```

The default graphics provided by functionalities within library `leaps` shows the selection of the covariates according to different criteria. BIC is the default selection

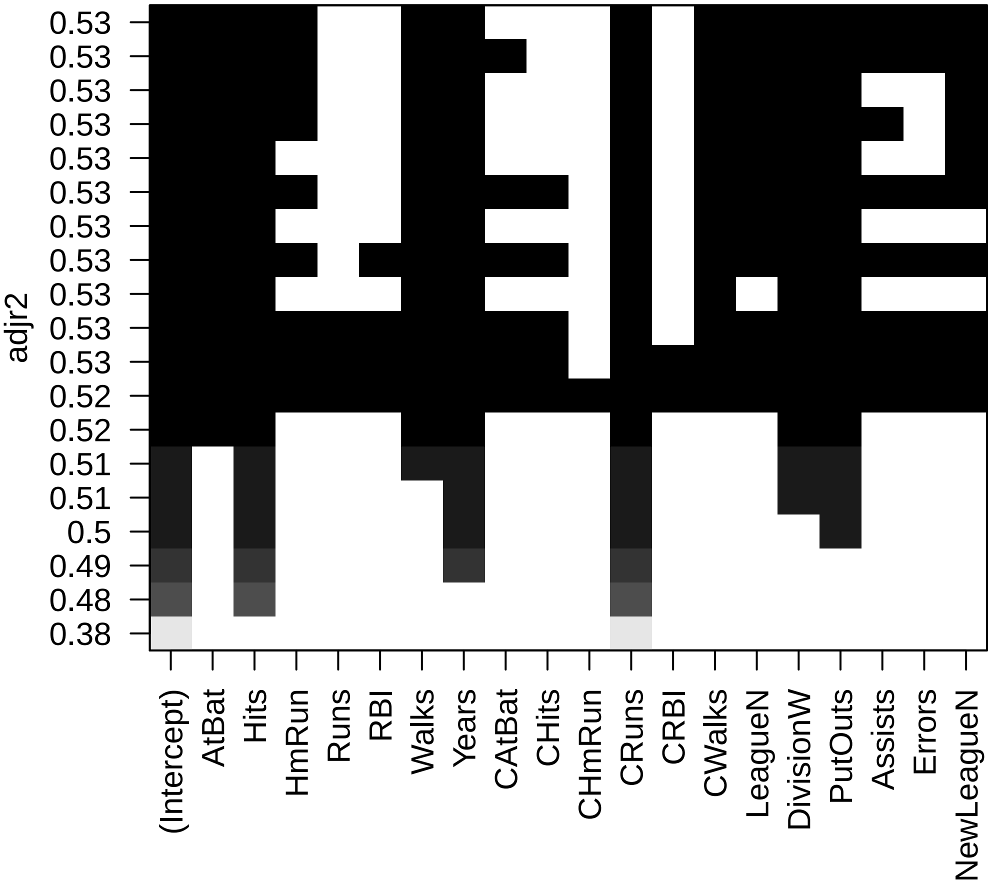
```
plot(m.forward)
```



The covariates remaining in the best model, the one with the smallest BIC, are indicated in black in the upper part of the panel. There are 4 covariates (intercept excluded), as concluded before.

The selection according to other criteria is possible using an appropriate specification, for example

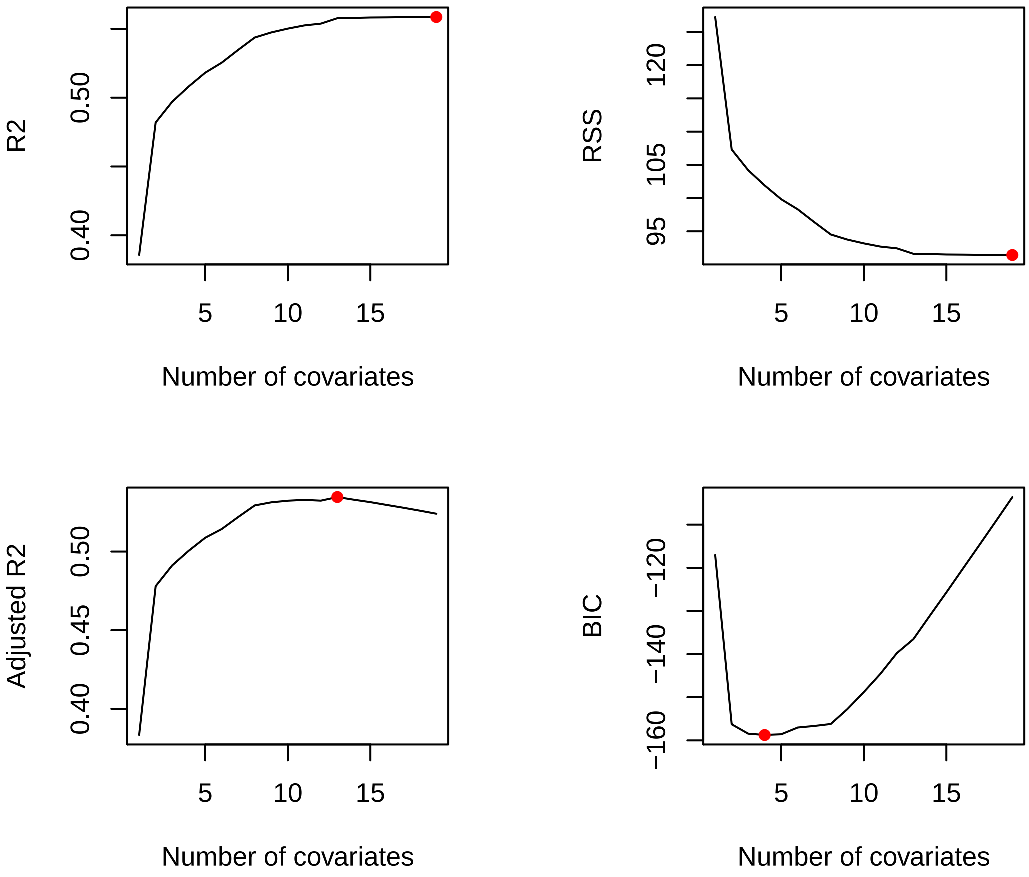
```
plot(m.forward, scale='adjr2')
```



The ranking of the models according to different criteria can be graphically visualized as follows

```
par(mfrow=c(2,2))
## R2
plot(summary(m.forward)$rsq, xlab='Number of covariates', ylab='R2', type='l')
## add on the indication of the best model
max.rsq <- which.max(summary(m.forward)$rsq)
points(max.rsq, summary(m.forward)$rsq[max.rsq], col='red', pch=16)
## RSS
```

```
plot(summary(m.forward)$rss, xlab='Number of covariates', ylab='RSS', type='l')
min.rss <- which.min(summary(m.forward)$rss)
points(min.rss, summary(m.forward)$rss[min.rss], col='red', pch=16)
## Adjusted R2
plot(summary(m.forward)$adjr2, xlab='Number of covariates',
        ylab='Adjusted R2', type='l')
max.adjr2 <- which.max(summary(m.forward)$adjr2)
points(max.adjr2, summary(m.forward)$adjr2[max.adjr2], col='red', pch=16)
## BIC
plot(summary(m.forward)$bic, xlab='Number of covariates', ylab='BIC', type='l')
min.bic <- which.min(summary(m.forward)$bic)
points(min.bic, summary(m.forward)$bic[min.bic], col='red', pch=16)
```

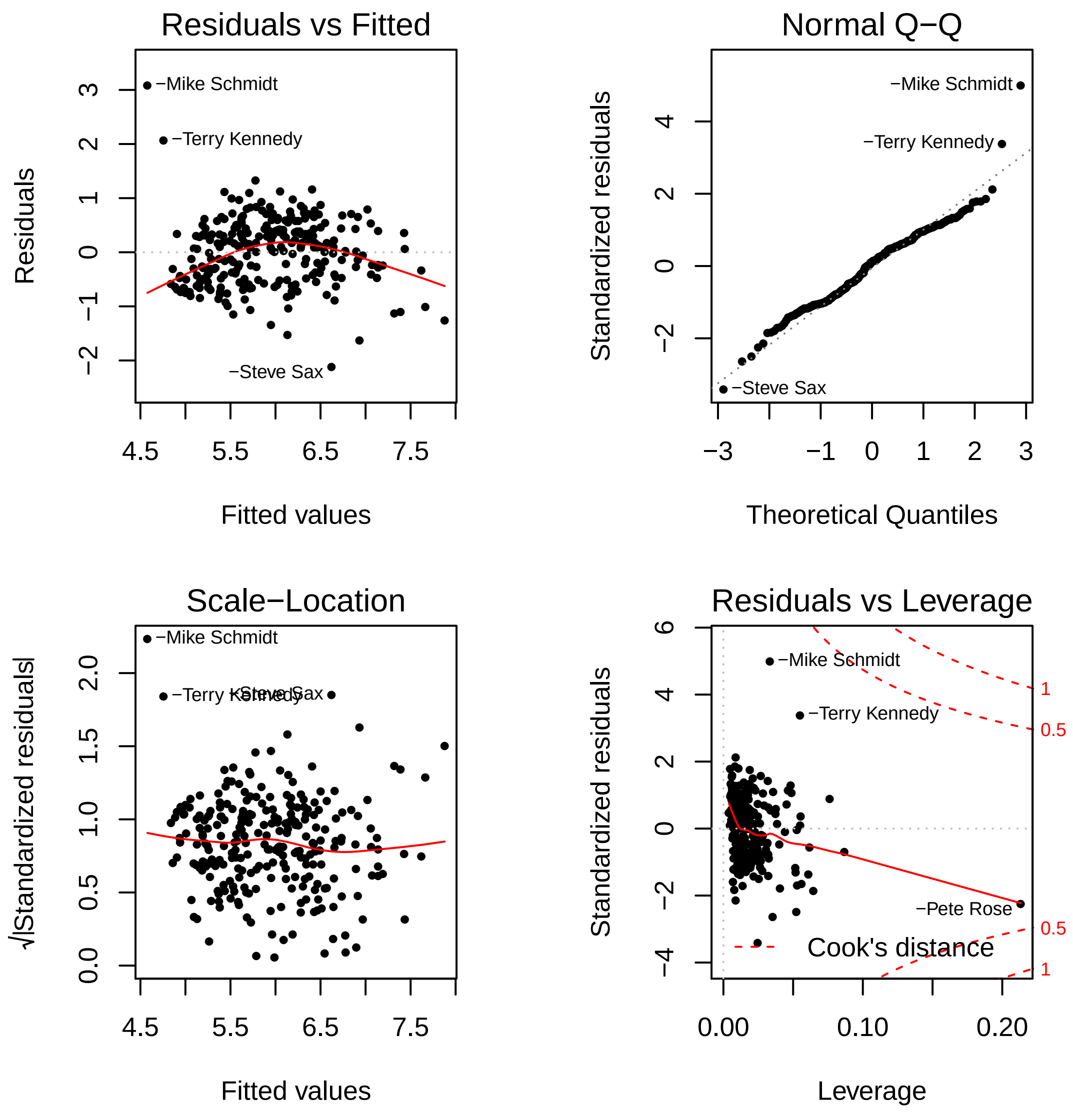


Function `which.max()` and function `which.min()` give the position of the maximum and the minimum of a vector.
After automatic selection, we could estimate the model with `lm()` inserting the covariates selected in `m.forward` and proceed with standard analyses (p-value, confidence intervals, residual analysis, ...).
For example, with BIC selection

```
model.bic <- lm(Salary ~ Hits + Years + CRuns + PutOuts, data=hitters)
summary(model.bic)

##
## Call:
## lm(formula = Salary ~ Hits + Years + CRuns + PutOuts, data = hitters)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.12133 -0.47896  0.06933  0.41313  3.08501
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.4323573  0.1364767  32.477  < 2e-16 ***
## Hits        0.0067804  0.0010049   6.747 9.88e-11 ***
## Years       0.0532853  0.0184531   2.888  0.00421 **
## CRuns       0.0007532  0.0002749   2.739  0.00658 **
## PutOuts     0.0003511  0.0001455   2.414  0.01649 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6284 on 258 degrees of freedom
## Multiple R-squared:  0.5081,Adjusted R-squared:  0.5005
## F-statistic: 66.63 on 4 and 258 DF,  p-value: < 2.2e-16
```

Residual analysis, using the default plots in R

```
par(mfrow=c(2,2))
plot(model.bic, pch=16, cex=0.7)
```

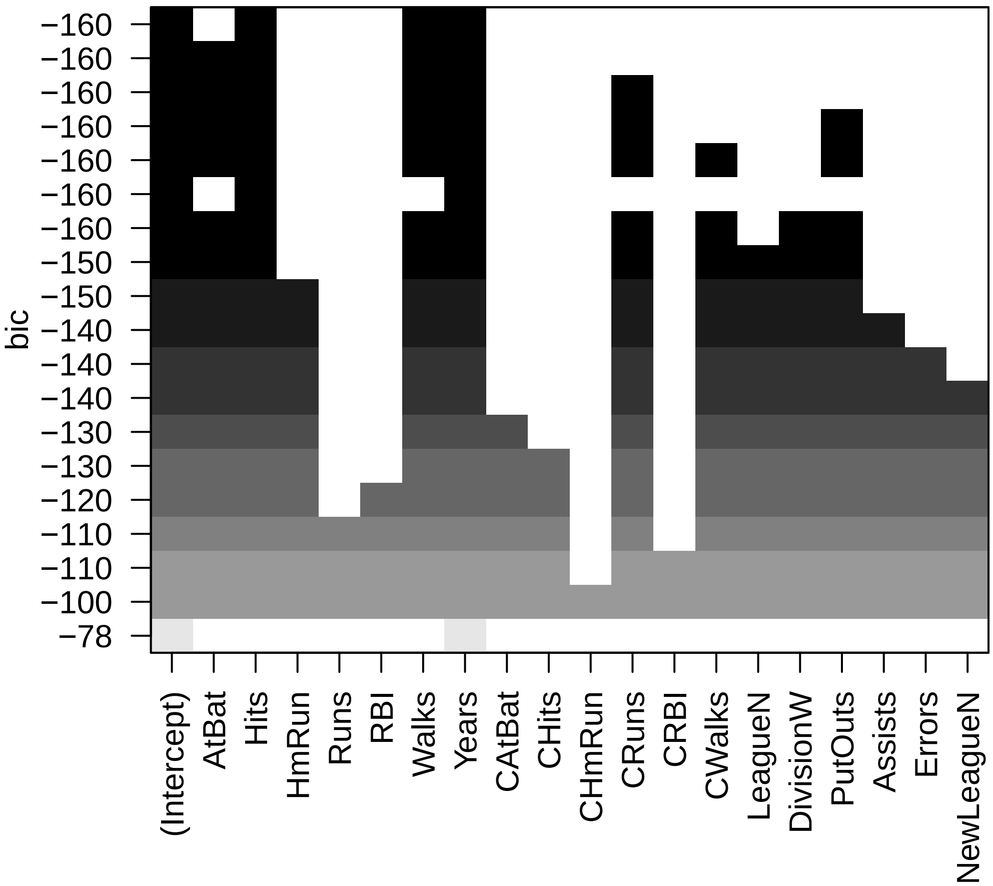


Comments?

Backward selection

```
m.backward <- regsubsets(Salary ~ ., data=hitters, nvmax=19, method='backward')
```

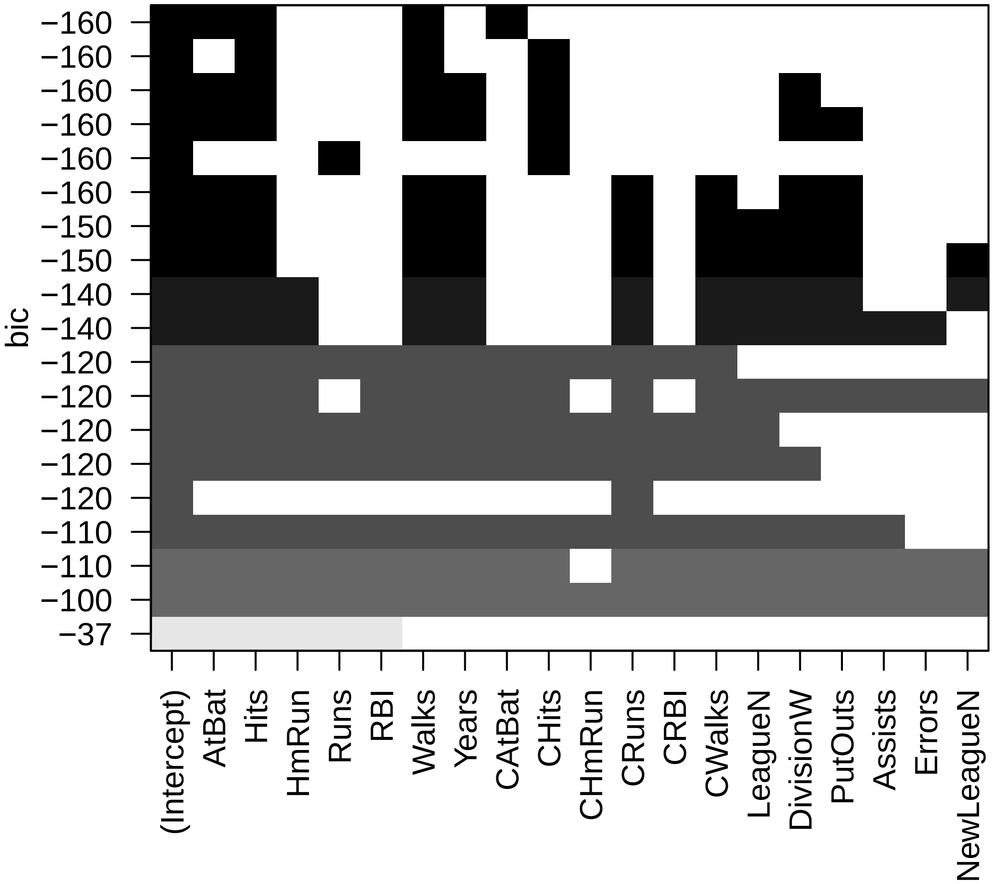
```
plot(m.backward)
```



As you can see, forward and backward selection do not necessarily choose the same models.
Mixed selection

```
m.seqrep <- regsubsets(Salary ~ ., data=hitters, nvmax=19, method='seqrep')
```

```
plot(m.seqrep)
```


bic
−160
−160
−160
−160
−160
−160
−150
−150
−140
−140
−120
−120
−120
−120
−120
−110
−110
−100
−37
(Intercept)
AtBat
Hits
HmRun
Runs
RBI
Walks
Years
CAtBat
CHits
CHmRun
CRuns
CRBI
CWalks
LeagueN
DivisionW
PutOuts
Assists
Errors
NewLeagueN