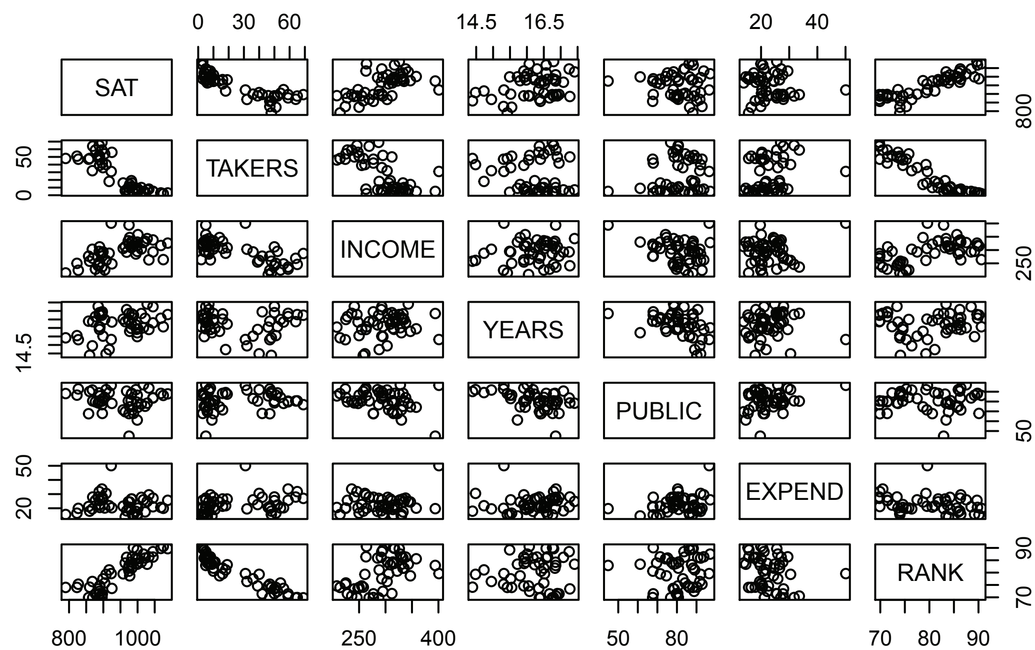
Model Selection

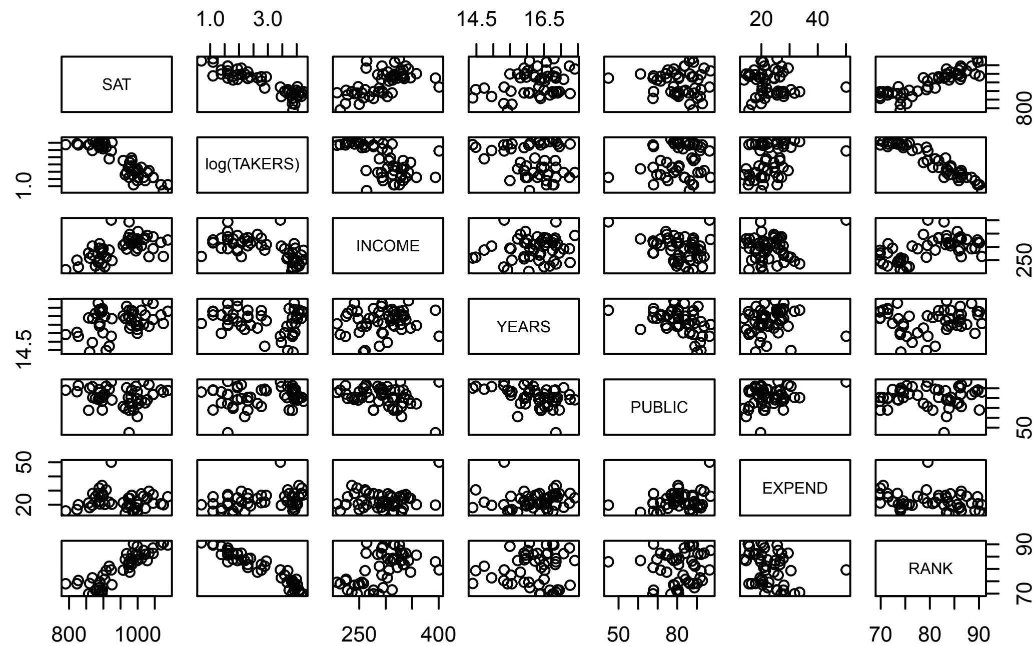
The file `case1201.csv` contains data on the average SAT scores by state. The states have been ordered by how well their students did on the SAT on average. Researchers have tried to explain the state by state differences in scores. The second column is the average SAT scores, along with six variables that may be associated with the SAT differences among states: percentage of the total eligible students who took the exam, median income of families of test takers, average number of years that the test takers had formal studies in social studies, natural sciences, humanities, percentage of test takers who attended public secondary schools, total state expenditure on secondary schools (dollars per student), and median percentile ranking of the test takers within their secondary school classes.

```
case1201 <- read.csv("case1201.csv")
```

```
pairs(SAT ~ TAKERS + INCOME + YEARS + PUBLIC + EXPEND + RANK, case1201)
```



```r
pairs(SAT ~ log(TAKERS) + INCOME + YEARS + PUBLIC + EXPEND + RANK, case1201)
```




```r
case1201 <- subset(case1201, STATE != "Alaska")
```

```r
library(broom)
fit <- lm(SAT ~ log(TAKERS) + INCOME + YEARS + PUBLIC + EXPEND + RANK, case1201)
#summary(fit)
tidy(fit)
```

```
##          term    estimate   std.error  statistic      p.value
## 1 (Intercept) 287.5241562 259.4170259  1.1083473 2.740186e-01
## 2 log(TAKERS) -30.2149086  14.7079011 -2.0543318 4.620090e-02
## 3      INCOME   0.1028689   0.1258568  0.8173485 4.183419e-01
## 4       YEARS  13.1073471   5.8798343  2.2292035 3.120553e-02
## 5      PUBLIC  -0.1011473   0.5105242 -0.1981245 8.439035e-01
## 6      EXPEND   3.9367309   0.8485803  4.6391964 3.404095e-05
## 7        RANK   5.2737739   2.2997494  2.2931950 2.690948e-02
```

```r
glance(fit)
```

```
##   r.squared adj.r.squared    sigma statistic      p.value df    logLik
## 1 0.9128002     0.9003431 22.57091  73.27545 1.198019e-20  7 -218.4677
##        AIC    BIC deviance df.residual
## 1 452.9355 468.07 21396.74          42
```

P-Value Model Selection

P-value model selection strategies choose which terms to include based on the significance of the terms using the relevant t-tests.

Backward Selection

Set a threshold α^*, then the backwards elimination algorithm is:

1. Begin with all possible predictors in the model.
2. Remove predictor with largest p-value above α^*.
3. Refit and repeat step 2 until all p-values below α^*.

Forward Selection

Set a threshold α^*, then the forward selection algorithm is:

1. Begin with no predictors in the model.
2. Of predictors not in the model, add predictor with smallest p-value below α^*.
3. Refit and repeat step 2 until no new p-values below α^*.

Stepwise Selection

Given thresholds α_F^* and α_B^*, then the stepwise selection algorithm is:

1. Begin with no predictors in the model.
2. Forward step: add predictor with smallest p-value below α_F^*.
3. Backward step: remove predictor with largest p-value above α_B^*.
4. Repeat steps 2–3 until convergence (or a max number of steps are reached).

α_F^* and α_B^* do not need to be the same.

The following example shows using backward selection with $\alpha^* = 0.05$.

```
fit1 <- update(fit, . ~ . - PUBLIC)
tidy(fit1)
```

```
##          term    estimate   std.error statistic      p.value
## 1 (Intercept) 291.1605132 255.8597988  1.137969 2.614344e-01
## 2 log(TAKERS) -31.1553044  13.7645661 -2.263443 2.871172e-02
## 3      INCOME   0.1134669   0.1126438  1.007306 3.194225e-01
## 4       YEARS  13.4920970   5.4875359  2.458680 1.804709e-02
## 5      EXPEND   3.8717915   0.7739292  5.002772 1.003109e-05
## 6        RANK   5.0600971   2.0083600  2.519517 1.554793e-02
```

```
fit2 <- update(fit1, . ~ . - INCOME)
tidy(fit2)
```

```
##          term   estimate   std.error statistic      p.value
## 1 (Intercept) 399.114660 232.3716497  1.717570 9.290657e-02
## 2 log(TAKERS) -38.100499  11.9152034 -3.197637 2.567643e-03
## 3       YEARS  13.147307   5.4777630  2.400123 2.068723e-02
## 4      EXPEND   3.995661   0.7642246  5.228385 4.519602e-06
## 5        RANK   4.400277   1.8988528  2.317335 2.520003e-02
```

```r
anova(fit2,fit)
```

```
## Analysis of Variance Table
##
## Model 1: SAT ~ log(TAKERS) + YEARS + EXPEND + RANK
## Model 2: SAT ~ log(TAKERS) + INCOME + YEARS + PUBLIC + EXPEND + RANK
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1     44 21922
## 2     42 21397  2    525.37 0.5156 0.6009
```

Model Selection Criterion

Recall the maximized log-likelihood of a fitted model,

$$L(\hat{\boldsymbol{\beta}}) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log\left(\frac{RSS}{n}\right) - \frac{n}{2}$$

Akaike Information Criterion

$$AIC = -2L(\hat{\boldsymbol{\beta}}) + 2p = n + n\log(2\pi) + n\log\left(\frac{RSS}{n}\right) + 2p$$

Thus for comparing models,

$$AIC \approx n\log\left(\frac{RSS}{n}\right) + 2p.$$

AIC quantifies the trade-off between a model which fits well and the number of model parameters. When selecting a model using AIC, we pick the model with the smallest AIC.

Bayesian Information Criterion

$$BIC = -2L(\hat{\boldsymbol{\beta}}) + \log(n)p = n + n\log(2\pi) + n\log\left(\frac{RSS}{n}\right) + \log(n)p.$$

Thus for comparing models,

$$BIC \approx n\log\left(\frac{RSS}{n}\right) + \log(n)p.$$

BIC also quantifies the trade-off between a model which fits well and the number of model parameters, however for a reasonable sample size, generally picks a smaller model than AIC. Again, for model selection use the model with the smallest BIC.

Adjusted R^2

Recall,

$$R^2 = 1 - \frac{SSE}{SST}.$$

Now define,

$$R_a^2 = 1 - \frac{SSE/(n-p)}{SST/(n-1)} = 1 - \left(\frac{n-1}{n-p}\right)(1 - R^2).$$

We can use R_a^2 to pick a model by selecting the model with the largest R_a^2.

PRESS

The Predicted Residual Sum-of-Squares (PRESS) statistic is

$$\text{PRESS} = \sum_{i=1}^{n} \left(y_i - \hat{y}_{[-i]}\right)^2 = \sum_{i=1}^{n} \left(\frac{\hat{e}_i}{1 - h_{ii}}\right)^2$$

where

- $\hat{y}_{[-i]} = \mathbf{x}_i \hat{\beta}_{[-i]}$ and $\hat{\beta}_{[-i]}$ is estimate of β without the i-th observation
- $\hat{e}_i$ is i-th estimated residual from full model
- h_{ii} is i-th leverage score from full model

PRESS can be used to select a model by picked the model with the lowest PRESS.

Continuing the SAT example, we can a backwards selection method with AIC.

```r
fit_back_aic <- step(fit, direction = "backward")
```

```
## Start:  AIC=311.88
## SAT ~ log(TAKERS) + INCOME + YEARS + PUBLIC + EXPEND + RANK
##
##               Df Sum of Sq   RSS    AIC
## - PUBLIC       1      20.0 21417 309.93
## - INCOME       1     340.3 21737 310.65
## <none>                     21397 311.88
## - log(TAKERS)  1    2150.0 23547 314.57
## - YEARS        1    2531.6 23928 315.36
## - RANK         1    2679.0 24076 315.66
## - EXPEND       1   10964.4 32361 330.15
##
## Step:  AIC=309.93
## SAT ~ log(TAKERS) + INCOME + YEARS + EXPEND + RANK
##
##               Df Sum of Sq   RSS    AIC
## - INCOME       1     505.4 21922 309.07
## <none>                     21417 309.93
## - log(TAKERS)  1    2551.7 23968 313.44
## - YEARS        1    3010.8 24428 314.37
## - RANK         1    3161.7 24578 314.67
## - EXPEND       1   12465.4 33882 330.40
##
## Step:  AIC=309.07
## SAT ~ log(TAKERS) + YEARS + EXPEND + RANK
##
##               Df Sum of Sq   RSS    AIC
## <none>                     21922 309.07
## - RANK         1    2675.5 24598 312.71
## - YEARS        1    2870.1 24792 313.10
## - log(TAKERS)  1    5094.3 27016 317.31
## - EXPEND       1   13619.6 35542 330.75
```

```r
tidy(fit_back_aic)
```

```
##          term   estimate   std.error statistic      p.value
## 1 (Intercept) 399.114660 232.3716497  1.717570 9.290657e-02
## 2 log(TAKERS) -38.100499  11.9152034 -3.197637 2.567643e-03
## 3       YEARS  13.147307   5.4777630  2.400123 2.068723e-02
## 4      EXPEND   3.995661   0.7642246  5.228385 4.519602e-06
## 5        RANK   4.400277   1.8988528  2.317335 2.520003e-02
```

```r
glance(fit_back_aic)
```

```
##   r.squared adj.r.squared    sigma statistic      p.value df   logLik
## 1 0.9106592     0.9025373 22.32106   112.124 1.762101e-22  5 -219.062
##        AIC     BIC deviance df.residual
## 1 450.1241 461.475  21922.1          44
```

We could have also gone forward using AIC.

```
fit_start <- lm(SAT ~ 1, case1201)
fit_forw_aic <- step(fit_start,
                SAT ~ log(TAKERS) + INCOME + YEARS + PUBLIC + EXPEND + RANK,
                direction = "forward")
```

```
## Start:  AIC=419.42
## SAT ~ 1
##
##               Df Sum of Sq    RSS    AIC
## + log(TAKERS)  1    199007  46369 339.78
## + RANK         1    190297  55079 348.21
## + INCOME       1    102026 143350 395.08
## + YEARS        1     26338 219038 415.85
## <none>                     245376 419.42
## + PUBLIC       1      1232 244144 421.17
## + EXPEND       1       386 244991 421.34
##
## Step:  AIC=339.78
## SAT ~ log(TAKERS)
##
##          Df Sum of Sq   RSS    AIC
## + EXPEND  1   20523.5 25846 313.14
## + YEARS   1    6363.5 40006 334.54
## <none>                46369 339.78
## + RANK    1     871.1 45498 340.85
## + INCOME  1     785.1 45584 340.94
## + PUBLIC  1     448.9 45920 341.30
##
## Step:  AIC=313.14
## SAT ~ log(TAKERS) + EXPEND
##
##          Df Sum of Sq   RSS    AIC
## + YEARS   1   1248.18 24598 312.71
## + RANK    1   1053.60 24792 313.10
## <none>                25846 313.14
## + INCOME  1     53.33 25793 315.04
## + PUBLIC  1      1.29 25845 315.13
##
## Step:  AIC=312.71
## SAT ~ log(TAKERS) + EXPEND + YEARS
##
##          Df Sum of Sq   RSS    AIC
## + RANK    1   2675.51 21922 309.07
## <none>                24598 312.71
## + PUBLIC  1    287.82 24310 314.13
## + INCOME  1     19.19 24578 314.67
##
## Step:  AIC=309.07
## SAT ~ log(TAKERS) + EXPEND + YEARS + RANK
##
##          Df Sum of Sq   RSS    AIC
## <none>                21922 309.07
```

```
## + INCOME  1    505.37 21417 309.93
## + PUBLIC  1    185.03 21737 310.65
```

```r
tidy(fit_forw_aic)
```

```
##          term   estimate   std.error statistic      p.value
## 1 (Intercept) 399.114660 232.3716497  1.717570 9.290657e-02
## 2 log(TAKERS) -38.100499  11.9152034 -3.197637 2.567643e-03
## 3      EXPEND   3.995661   0.7642246  5.228385 4.519602e-06
## 4       YEARS  13.147307   5.4777630  2.400123 2.068723e-02
## 5        RANK   4.400277   1.8988528  2.317335 2.520003e-02
```

```r
glance(fit_forw_aic)
```

```
##   r.squared adj.r.squared    sigma statistic      p.value df   logLik
## 1 0.9106592     0.9025373 22.32106   112.124 1.762101e-22  5 -219.062
##        AIC     BIC deviance df.residual
## 1 450.1241 461.475  21922.1          44
```

Or we could have used both directions with AIC.

```r
fit_both_aic <- step(fit, direction = "both")
```

```
## Start:  AIC=311.88
## SAT ~ log(TAKERS) + INCOME + YEARS + PUBLIC + EXPEND + RANK
##
##               Df Sum of Sq   RSS    AIC
## - PUBLIC       1      20.0 21417 309.93
## - INCOME       1     340.3 21737 310.65
## <none>                     21397 311.88
## - log(TAKERS)  1    2150.0 23547 314.57
## - YEARS        1    2531.6 23928 315.36
## - RANK         1    2679.0 24076 315.66
## - EXPEND       1   10964.4 32361 330.15
##
## Step:  AIC=309.93
## SAT ~ log(TAKERS) + INCOME + YEARS + EXPEND + RANK
##
##               Df Sum of Sq   RSS    AIC
## - INCOME       1     505.4 21922 309.07
## <none>                     21417 309.93
## + PUBLIC       1      20.0 21397 311.88
## - log(TAKERS)  1    2551.7 23968 313.44
## - YEARS        1    3010.8 24428 314.37
## - RANK         1    3161.7 24578 314.67
## - EXPEND       1   12465.4 33882 330.40
##
## Step:  AIC=309.07
## SAT ~ log(TAKERS) + YEARS + EXPEND + RANK
##
##               Df Sum of Sq   RSS    AIC
## <none>                     21922 309.07
```

```
## + INCOME       1     505.4 21417 309.93
## + PUBLIC       1     185.0 21737 310.65
## - RANK         1    2675.5 24598 312.71
## - YEARS        1    2870.1 24792 313.10
## - log(TAKERS)  1    5094.3 27016 317.31
## - EXPEND       1   13619.6 35542 330.75
```

```
tidy(fit_both_aic)
```

```
##          term   estimate   std.error statistic      p.value
## 1 (Intercept) 399.114660 232.3716497  1.717570 9.290657e-02
## 2 log(TAKERS) -38.100499  11.9152034 -3.197637 2.567643e-03
## 3       YEARS  13.147307   5.4777630  2.400123 2.068723e-02
## 4      EXPEND   3.995661   0.7642246  5.228385 4.519602e-06
## 5        RANK   4.400277   1.8988528  2.317335 2.520003e-02
```

```
glance(fit_both_aic)
```

```
##   r.squared adj.r.squared    sigma statistic      p.value df   logLik
## 1 0.9106592     0.9025373 22.32106   112.124 1.762101e-22  5 -219.062
##        AIC     BIC deviance df.residual
## 1 450.1241 461.475  21922.1          44
```

We could have also used BIC backwards, forwards, and in both directions. The code to do so follows. (Output omitted.)

```
n <- length(resid(fit))

#backwards
fit_back_bic <- step(fit, direction = "backward", k = log(n))
tidy(fit_back_bic)
glance(fit_back_bic)

#forwards
fit_forw_bic <- step(fit_start,
                SAT ~ log(TAKERS) + INCOME + YEARS + PUBLIC + EXPEND + RANK,
                direction = "forward", k = log(n))
tidy(fit_forw_bic)
glance(fit_forw_bic)

#stepwise
fit_both_bic <- step(fit, direction = "both", k = log(n))
tidy(fit_both_bic)
glance(fit_both_bic)
```

Using backwards, forwards, and stepwise procedures potentially ignore the "best" model. Using the `leaps` package, we can get AIC or BIC for every possible model.

```
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 3.1.3
```

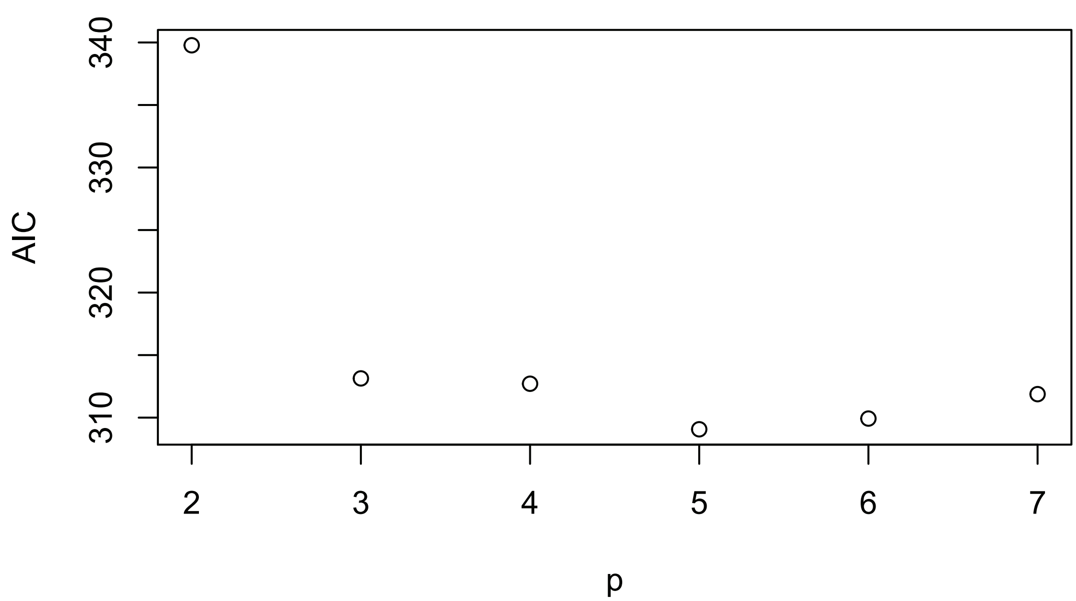
```
all_fits <- regsubsets(SAT ~
                       log(TAKERS) + INCOME + YEARS + PUBLIC + EXPEND + RANK,
                       data = case1201)

all_fits_sum <- summary(all_fits)
all_fits_sum$which
```

```
##   (Intercept) log(TAKERS) INCOME YEARS PUBLIC EXPEND  RANK
## 1        TRUE        TRUE  FALSE FALSE  FALSE  FALSE FALSE
## 2        TRUE        TRUE  FALSE FALSE  FALSE   TRUE FALSE
## 3        TRUE        TRUE  FALSE  TRUE  FALSE   TRUE FALSE
## 4        TRUE        TRUE  FALSE  TRUE  FALSE   TRUE  TRUE
## 5        TRUE        TRUE   TRUE  TRUE  FALSE   TRUE  TRUE
## 6        TRUE        TRUE   TRUE  TRUE   TRUE   TRUE  TRUE
```

We can then quickly obtain AIC, BIC and R_a^2 for the model with the smallest RSS of each size.

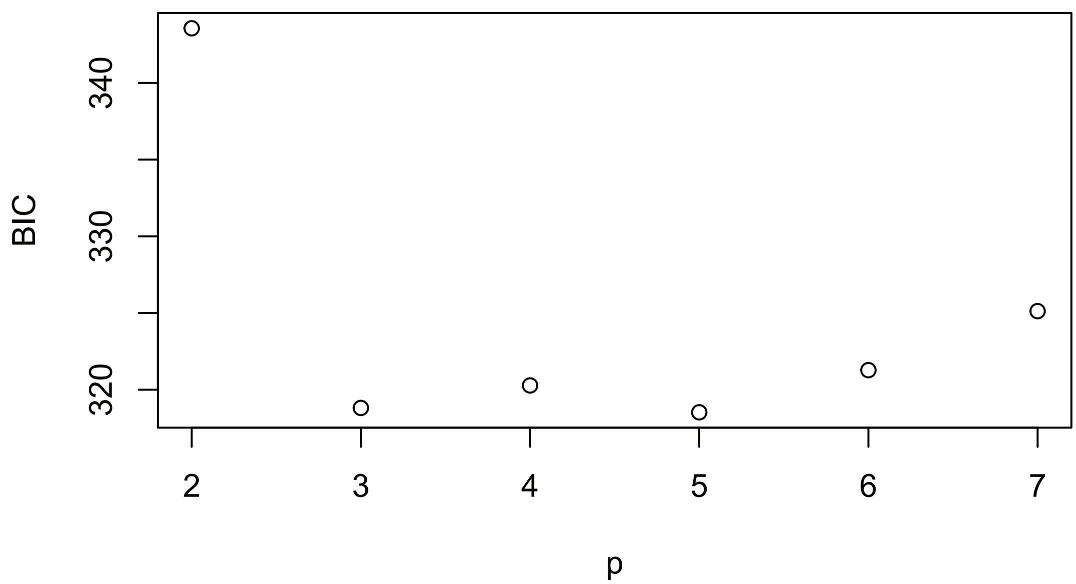
```
#AIC
p <- length(coef(fit))
n <- length(resid(fit))
AIC <- n*log(all_fits_sum$rss/n) + 2*(2:p)
plot(AIC ~ I(2:p), ylab = "AIC", xlab = "p")
```



```r
which.min(AIC)
```

```
## [1] 4
```

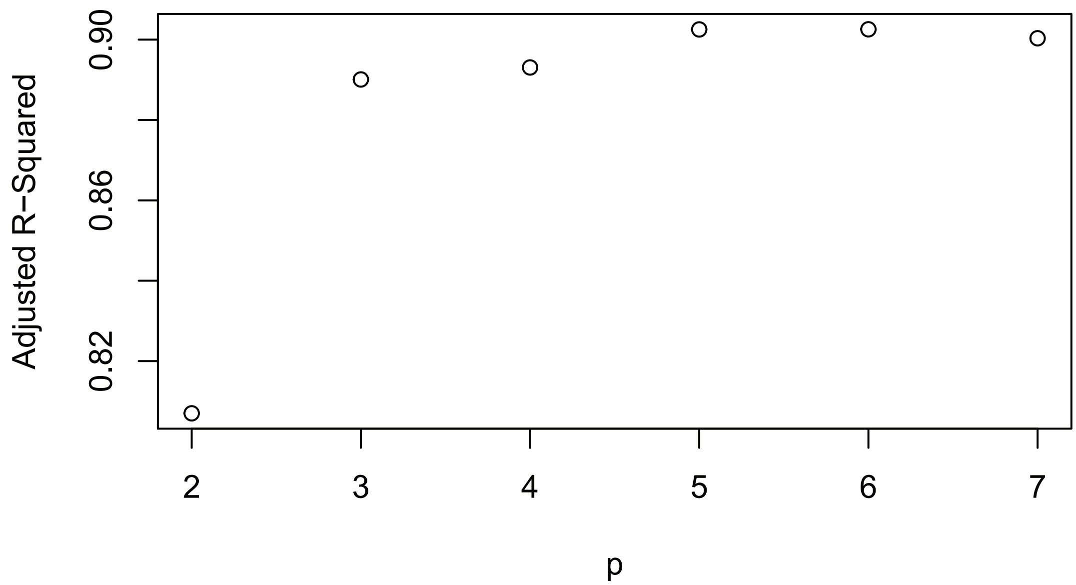
```r
#BIC
BIC <- n*log(all_fits_sum$rss/n) + log(n)*(2:p)
plot(BIC ~ I(2:p), ylab = "BIC", xlab = "p")
```




```r
which.min(BIC)
```

```
## [1] 4
```

```r
#R2adj
R_adj <- all_fits_sum$adjr2
plot(R_adj ~ I(2:p), ylab = "Adjusted R-Squared", xlab = "p")
```




```r
which.max(R_adj)
```

```
## [1] 5
```

```r
R_adj
```

```
## [1] 0.8070071 0.8900890 0.8930725 0.9025373 0.9025698 0.9003431
```

A study of the relation of amount of body fat (`bodyfat`) to several possible predictor variables, based on a sample of 20 healthy females 25–34 years old. The possible predictor variables are triceps skinfold thickness (`tricep`), thigh circumference (`thigh`), and midarm circumference (`midarm`).

```r
tricep <- c(19.5,24.7,30.7,29.8,19.1,25.6,31.4,27.9,22.1,25.5,
            31.1,30.4,18.7,19.7,14.6,29.5,27.7,30.2,22.7,25.2)
thigh <- c(43.1,49.8,51.9,54.3,42.2,53.9,58.5,52.1,49.9,53.5,
           56.6,56.7,46.5,44.2,42.7,54.4,55.3,58.6,48.2,51.0)
midarm <- c(29.1,28.2,37.0,31.1,30.9,23.7,27.6,30.6,23.2,24.8,
            30.0,28.3,23.0,28.6,21.3,30.1,25.7,24.6,27.1,27.5)
bodyfat <- c(11.9,22.8,18.7,20.1,12.9,21.7,27.1,25.4,21.3,19.3,
             25.4,27.2,11.7,17.8,12.8,23.9,22.6,25.4,14.8,21.1)
bodyfat <- data.frame(bodyfat, tricep, thigh, midarm)

rbind(
  summary( lm(bodyfat~tricep, data = bodyfat) )$adj.r.squared,
  summary( lm(bodyfat~thigh, data = bodyfat) )$adj.r.squared,
  summary( lm(bodyfat~midarm, data = bodyfat) )$adj.r.squared,
  summary( lm(bodyfat~tricep+thigh, data = bodyfat) )$adj.r.squared,
  summary( lm(bodyfat~tricep+midarm, data = bodyfat) )$adj.r.squared,
  summary( lm(bodyfat~thigh+midarm, data = bodyfat) )$adj.r.squared,
  summary( lm(bodyfat~tricep+thigh+midarm, data = bodyfat) )$adj.r.squared
)
```

```
##             [,1]
## [1,]  0.69504643
## [2,]  0.75832149
## [3,] -0.03413801
## [4,]  0.75194029
## [5,]  0.76100218
## [6,]  0.74932549
## [7,]  0.76411328
```

```r
fit <- lm(bodyfat ~ tricep + thigh + midarm, data = bodyfat)
all_fits <- regsubsets(bodyfat ~ tricep + thigh + midarm, data = bodyfat)
all_fits_sum <- summary(all_fits)
all_fits_sum$which
```

```
##   (Intercept) tricep thigh midarm
## 1        TRUE  FALSE  TRUE  FALSE
## 2        TRUE   TRUE FALSE   TRUE
## 3        TRUE   TRUE  TRUE   TRUE
```

```r
all_fits_sum$adjr2
```

```
## [1] 0.7583215 0.7610022 0.7641133
```

```r
fit_back <- step(fit, dir="backward")
```

```
## Start:  AIC=39.87
## bodyfat ~ tricep + thigh + midarm
```

```
##
##          Df Sum of Sq     RSS    AIC
## - thigh   1    7.5293 105.934 39.342
## <none>                98.405 39.867
## - midarm  1   11.5459 109.951 40.086
## - tricep  1   12.7049 111.110 40.296
##
## Step:  AIC=39.34
## bodyfat ~ tricep + midarm
##
##          Df Sum of Sq    RSS    AIC
## <none>               105.93 39.342
## - midarm  1     37.19 143.12 43.359
## - tricep  1    379.40 485.34 67.782
```

```r
summary(fit_back)
```

```
##
## Call:
## lm(formula = bodyfat ~ tricep + midarm, data = bodyfat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.8794 -1.9627  0.3811  1.2688  3.8942
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   6.7916     4.4883   1.513   0.1486
## tricep        1.0006     0.1282   7.803 5.12e-07 ***
## midarm       -0.4314     0.1766  -2.443   0.0258 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.496 on 17 degrees of freedom
## Multiple R-squared:  0.7862, Adjusted R-squared:  0.761
## F-statistic: 31.25 on 2 and 17 DF,  p-value: 2.022e-06
```

```r
fit_forw <- step(lm(bodyfat ~ 1, data = bodyfat),
                 bodyfat~tricep+thigh+midarm, dir="forward")
```

```
## Start:  AIC=66.19
## bodyfat ~ 1
##
##          Df Sum of Sq    RSS    AIC
## + thigh   1    381.97 113.42 38.708
## + tricep  1    352.27 143.12 43.359
## <none>               495.39 66.192
## + midarm  1     10.05 485.34 67.782
##
## Step:  AIC=38.71
## bodyfat ~ thigh
##
##          Df Sum of Sq    RSS    AIC
```

```
## <none>                 113.42 38.708
## + tricep  1    3.4729 109.95 40.086
## + midarm  1    2.3139 111.11 40.296
```

```r
summary(fit_forw)
```

```
##
## Call:
## lm(formula = bodyfat ~ thigh, data = bodyfat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4949 -1.5671  0.1241  1.3362  4.4084
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -23.6345     5.6574  -4.178 0.000566 ***
## thigh         0.8565     0.1100   7.786  3.6e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.51 on 18 degrees of freedom
## Multiple R-squared:  0.771,  Adjusted R-squared:  0.7583
## F-statistic: 60.62 on 1 and 18 DF,  p-value: 3.6e-07
```

```r
rbind(
  extractAIC( lm(bodyfat~tricep, data = bodyfat) ),
  extractAIC( lm(bodyfat~thigh, data = bodyfat) ),
  extractAIC( lm(bodyfat~midarm, data = bodyfat) ),
  extractAIC( lm(bodyfat~tricep+thigh, data = bodyfat) ),
  extractAIC( lm(bodyfat~tricep+midarm, data = bodyfat) ),
  extractAIC( lm(bodyfat~thigh+midarm, data = bodyfat) ),
  extractAIC( lm(bodyfat~tricep+thigh+midarm, data = bodyfat) )
)
```

```
##      [,1]     [,2]
## [1,]    2 43.35898
## [2,]    2 38.70796
## [3,]    2 67.78226
## [4,]    3 40.08601
## [5,]    3 39.34171
## [6,]    3 40.29573
## [7,]    4 39.86716
```

```r
n <- length(resid(fit))
p <- length(coef(fit))
AIC <- n*log(all_fits_sum$rss/n) + 2*(2:p)
AIC
```

```
## [1] 38.70796 39.34171 39.86716
```

```r
fit1 <- lm(bodyfat~tricep, data = bodyfat)
fit2 <- lm(bodyfat~thigh, data = bodyfat)
fit3 <- lm(bodyfat~midarm, data = bodyfat)
fit4 <- lm(bodyfat~tricep+thigh, data = bodyfat)
fit5 <- lm(bodyfat~tricep+midarm, data = bodyfat)
fit6 <- lm(bodyfat~thigh+midarm, data = bodyfat)
fit7 <- lm(bodyfat~tricep+thigh+midarm, data = bodyfat)


rbind(
  sum( (resid(fit1) / (1 - hatvalues(fit1)))^2 ),
  sum( (resid(fit2) / (1 - hatvalues(fit2)))^2 ),
  sum( (resid(fit3) / (1 - hatvalues(fit3)))^2 ),
  sum( (resid(fit4) / (1 - hatvalues(fit4)))^2 ),
  sum( (resid(fit5) / (1 - hatvalues(fit5)))^2 ),
  sum( (resid(fit6) / (1 - hatvalues(fit6)))^2 ),
  sum( (resid(fit7) / (1 - hatvalues(fit7)))^2 )
)
```

```
##          [,1]
## [1,] 177.7846
## [2,] 134.4416
## [3,] 605.0455
## [4,] 154.4736
## [5,] 148.3335
## [6,] 155.4313
## [7,] 160.7366
```