CAR: Chapter 1

georg

2019-12-25

```
knitr::opts_chunk$set(error=TRUE)
library(knitr)
```

An R Companion to Applied Regression, 3rd Edition

J. Fox and S. Weisberg, Sage Publications

Script for Chapter 1

```
library("car")
```

```
## Loading required package: carData
```

Arithmetic in reverse order of precedence

```
2 + 3 # addition
```

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

```
2 - 3 # subtraction
```

```
## [1] -1
```

```
2*3   # multiplication
```

```
## [1] 6
```

```
2/3   # division
```

```
## [1] 0.6666667
```

```
2^3   # exponentiation
```

```
## [1] 8
```

```
4^2 - 3*2
```

```
## [1] 10
```

```
1 - 6 + 4
```

```
## [1] -1
```

```
(4^2) - (3*2)
```

```
## [1] 10
```

```r
(4 + 3)^2
```

```
## [1] 49
```

```r
4 + 3^2
```

```
## [1] 13
```

```r
-2--3
```

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

```r
-2 - -3
```

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

```r
log(100)
```

```
## [1] 4.60517
```

```r
log(100, base=10)
```

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

```r
log10(100) # equivalent
```

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

```r
log(100, b=10)
```

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

```r
args("log")
```

```
## function (x, base = exp(1))
## NULL
```

```r
log(100, 10)
```

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

```r
c(1, 2, 3, 4)
```

```
## [1] 1 2 3 4
```

```r
1:4 # integer sequence
```

```
## [1] 1 2 3 4
```

```r
4:1 # descending
```

```
## [1] 4 3 2 1
```

```r
-1:2 # negative to positive
```

```
## [1] -1  0  1  2
```

```r
seq(1, 4) # equivalent to 1:4
```

```
## [1] 1 2 3 4
```

```r
seq(2, 8, by=2) # specify interval between elements
```

```
## [1] 2 4 6 8
```

```r
seq(0, 1, by=0.1) # noninteger sequence
```

```
##  [1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
```

```r
seq(0, 1, length=11) # specify number of elements
```

```
##  [1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
```

```r
c(1, 2, 3, 4)/2
```

```
## [1] 0.5 1.0 1.5 2.0
```

```r
c(1, 2, 3, 4)/c(4, 3, 2, 1)
```

```
## [1] 0.2500000 0.6666667 1.5000000 4.0000000
```

```r
log(c(0.1, 1, 10, 100), base=10)
```

```
## [1] -1  0  1  2
```

```r
c(1, 2, 3, 4) + c(4, 3) # no warning
```

```
## [1] 5 5 7 7
```

```r
x <- c(1, 2, 3, 4) # assignment
x # print
```

```
## [1] 1 2 3 4
```

```r
x/2  # equivalent to c(1, 2, 3, 4)/2
```

```
## [1] 0.5 1.0 1.5 2.0
```

```r
(y <- sqrt(x))
```

```
## [1] 1.000000 1.414214 1.732051 2.000000
```

```r
set.seed(12345)
(x <- rnorm(100))  # 100 standard normal random numbers
```

```
##   [1]  0.58552882  0.70946602 -0.10930331 -0.45349717  0.60588746 -1.81795597
##   [7]  0.63009855 -0.27618411 -0.28415974 -0.91932200 -0.11624781  1.81731204
##  [13]  0.37062786  0.52021646 -0.75053199  0.81689984 -0.88635752 -0.33157759
##  [19]  1.12071265  0.29872370  0.77962192  1.45578508 -0.64432843 -1.55313741
##  [25] -1.59770952  1.80509752 -0.48164736  0.62037980  0.61212349 -0.16231098
##  [31]  0.81187318  2.19683355  2.04919034  1.63244564  0.25427119  0.49118828
##  [37] -0.32408658 -1.66205024  1.76773385  0.02580105  1.12851083 -2.38035806
##  [43] -1.06026555  0.93714054  0.85445172  1.46072940 -1.41309878  0.56740325
##  [49]  0.58318765 -1.30679883 -0.54038607  1.94769266  0.05359027  0.35166284
##  [55] -0.67097654  0.27795369  0.69117127  0.82379533  2.14506502 -2.34694398
##  [61]  0.14959198 -1.34253148  0.55330308  1.58996284 -0.58687959 -1.83237731
##  [67]  0.88813943  1.59348847  0.51685467 -1.29567168  0.05461558 -0.78464937
##  [73] -1.04935282  2.33051196  1.40270538  0.94260085  0.82625829 -0.81154049
##  [79]  0.47624828  1.02125841  0.64538307  1.04314355 -0.30436911  2.47711092
##  [85]  0.97122067  1.86709918  0.67204247 -0.30795338  0.53652372  0.82487007
##  [91] -0.96390148 -0.85508251  1.88694694 -0.39181937 -0.98063295  0.68733210
##  [97] -0.50504352  2.15771982 -0.59979756 -0.69454669
```

```r
summary(x)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

```
## -2.3804 -0.5901  0.4837  0.2452  0.9004  2.4771
```

```
(words <- c("To", "be", "or", "not", "to", "be"))
```

```
## [1] "To"  "be"  "or"  "not" "to"  "be"
```

```
paste(words, collapse=" ")
```

```
## [1] "To be or not to be"
```

```
(logical.values <- c(TRUE, TRUE, FALSE, TRUE))
```

```
## [1]  TRUE  TRUE FALSE  TRUE
```

```
!logical.values
```

```
## [1] FALSE FALSE  TRUE FALSE
```

```
sum(logical.values)
```

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

```
sum(!logical.values)
```

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

```
c("A", FALSE, 3.0)
```

```
## [1] "A"     "FALSE" "3"
```

```
c(10, FALSE, -6.5, TRUE)
```

```
## [1] 10.0  0.0 -6.5  1.0
```

```
x[12]             # 12th element
```

```
## [1] 1.817312
```

```
words[2]          # second element
```

```
## [1] "be"
```

```
logical.values[3] # third element
```

```
## [1] FALSE
```

```
x[6:15]       # elements 6 through 15
```

```
##  [1] -1.8179560  0.6300986 -0.2761841 -0.2841597 -0.9193220 -0.1162478
##  [7]  1.8173120  0.3706279  0.5202165 -0.7505320
```

```
x[c(1, 3, 5)] # 1st, 3rd, 5th elements
```

```
## [1]  0.5855288 -0.1093033  0.6058875
```

```
x[-(11:100)] # omit elements 11 through 100
```

```
##  [1]  0.5855288  0.7094660 -0.1093033 -0.4534972  0.6058875 -1.8179560
##  [7]  0.6300986 -0.2761841 -0.2841597 -0.9193220
```

```
v <- 1:4
v[c(TRUE, FALSE, FALSE, TRUE)]
```

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

```r
1 == 2
```

```
## [1] FALSE
```

```r
1 != 2
```

```
## [1] TRUE
```

```r
1 <= 2
```

```
## [1] TRUE
```

```r
1 < 1:3
```

```
## [1] FALSE  TRUE  TRUE
```

```r
3:1 > 1:3
```

```
## [1]  TRUE FALSE FALSE
```

```r
3:1 >= 1:3
```

```
## [1]  TRUE  TRUE FALSE
```

```r
TRUE & c(TRUE, FALSE)
```

```
## [1]  TRUE FALSE
```

```r
c(TRUE, FALSE, FALSE) | c(TRUE, TRUE, FALSE)
```

```
## [1]  TRUE  TRUE FALSE
```

```r
TRUE && FALSE
```

```
## [1] FALSE
```

```r
TRUE || FALSE
```

```
## [1] TRUE
```

```r
(z <- x[1:10]) # first 10 elements of x
```

```
##  [1]  0.5855288  0.7094660 -0.1093033 -0.4534972  0.6058875 -1.8179560
##  [7]  0.6300986 -0.2761841 -0.2841597 -0.9193220
```

```r
z < -0.5 # is each element less than -0.5?
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE
```

```r
z > 0.5  # is each element greater than 0.5
```

```
##  [1]  TRUE  TRUE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE
```

```r
z < -0.5 | z > 0.5  #  < and > are of higher precedence than |
```

```
##  [1]  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE FALSE FALSE  TRUE
```

```r
abs(z) > 0.5  # absolute value, equivalent to last expression
```

```
##  [1]  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE FALSE FALSE  TRUE
```

```r
z[abs(z) > 0.5] # values of z for which |z| > 0.5
```

```
## [1]  0.5855288  0.7094660  0.6058875 -1.8179560  0.6300986 -0.9193220
```

```r
z[!(abs(z) > 0.5)] # values z for which |z| <= 0.5
```

```
## [1] -0.1093033 -0.4534972 -0.2761841 -0.2841597
```

```r
mean(x)
```

```
## [1] 0.2451972
```

```r
sum(x)/length(x)
```

```
## [1] 0.2451972
```

```r
myMean <- function(x){
    sum(x)/length(x)
}

myMean(x)
```

```
## [1] 0.2451972
```

```r
y # from sqrt(c(1, 2, 3, 4))
```

```
## [1] 1.000000 1.414214 1.732051 2.000000
```

```r
myMean(y)
```

```
## [1] 1.536566
```

```r
myMean(1:100)
```

```
## [1] 50.5
```

```r
myMean(sqrt(1:100))
```

```
## [1] 6.714629
```

```r
mySD <- function(x){
    sqrt(sum((x - myMean(x))^2)/(length(x) - 1))
}
mySD(1:100)
```

```
## [1] 29.01149
```

```r
sd(1:100) # check
```

```
## [1] 29.01149
```

```r
mySD
```

```
## function(x){
##     sqrt(sum((x - myMean(x))^2)/(length(x) - 1))
## }
```

```r
myMean
```

```
## function(x){
##     sum(x)/length(x)
## }
## <bytecode: 0x0000000019063510>
```

```r
letters
```

```
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q" "r" "s"
## [20] "t" "u" "v" "w" "x" "y" "z"
```

```r
mySD(letters)
```

```
## Error in sum(x): invalid 'type' (character) of argument
```

```
head(Duncan, n=10)
```

```
##            type income education prestige
## accountant prof     62        86       82
## pilot      prof     72        76       83
## architect  prof     75        92       90
## author     prof     55        90       76
## chemist    prof     64        86       90
## minister   prof     21        84       87
## professor  prof     64        93       93
## dentist    prof     80       100       90
## reporter     wc     67        87       52
## engineer   prof     72        86       88
```

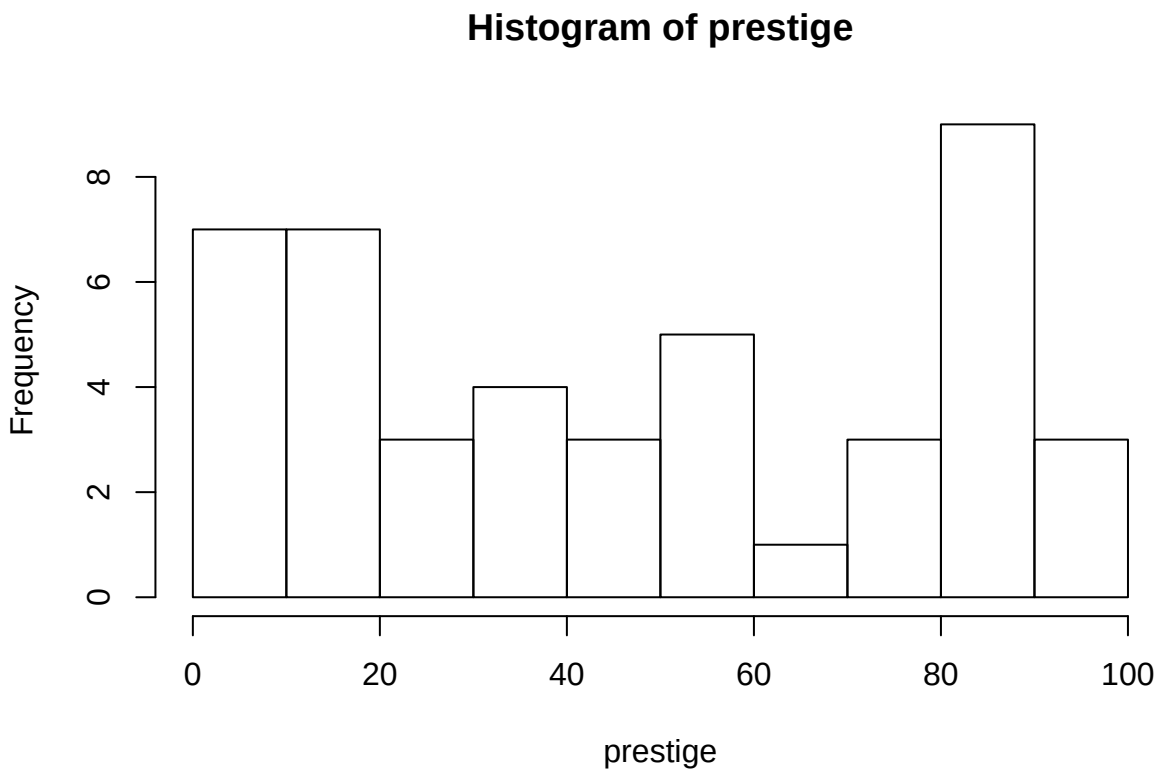
```
dim(Duncan)
```

```
## [1] 45  4
```

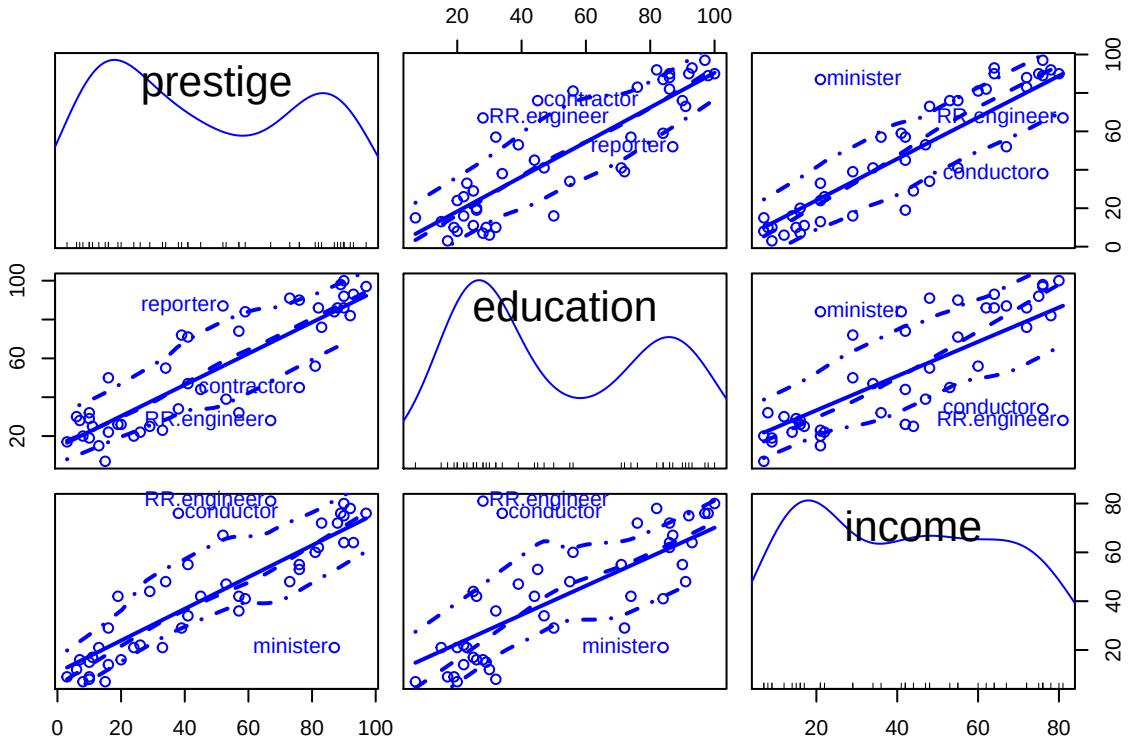
```
summary(Duncan)
```

```
##    type        income          education         prestige
##  bc  :21   Min.   : 7.00   Min.   :  7.00   Min.   : 3.00
##  prof:18   1st Qu.:21.00   1st Qu.: 26.00   1st Qu.:16.00
##  wc  : 6   Median :42.00   Median : 45.00   Median :41.00
##            Mean   :41.87   Mean   : 52.56   Mean   :47.69
##            3rd Qu.:64.00   3rd Qu.: 84.00   3rd Qu.:81.00
##            Max.   :81.00   Max.   :100.00   Max.   :97.00
```

```
with(Duncan, hist(prestige))
```




```
scatterplotMatrix( ~ prestige + education + income,
    id=list(n=3), data=Duncan)
```




```
(duncan.model <- lm(prestige ~ education + income, data=Duncan))
```

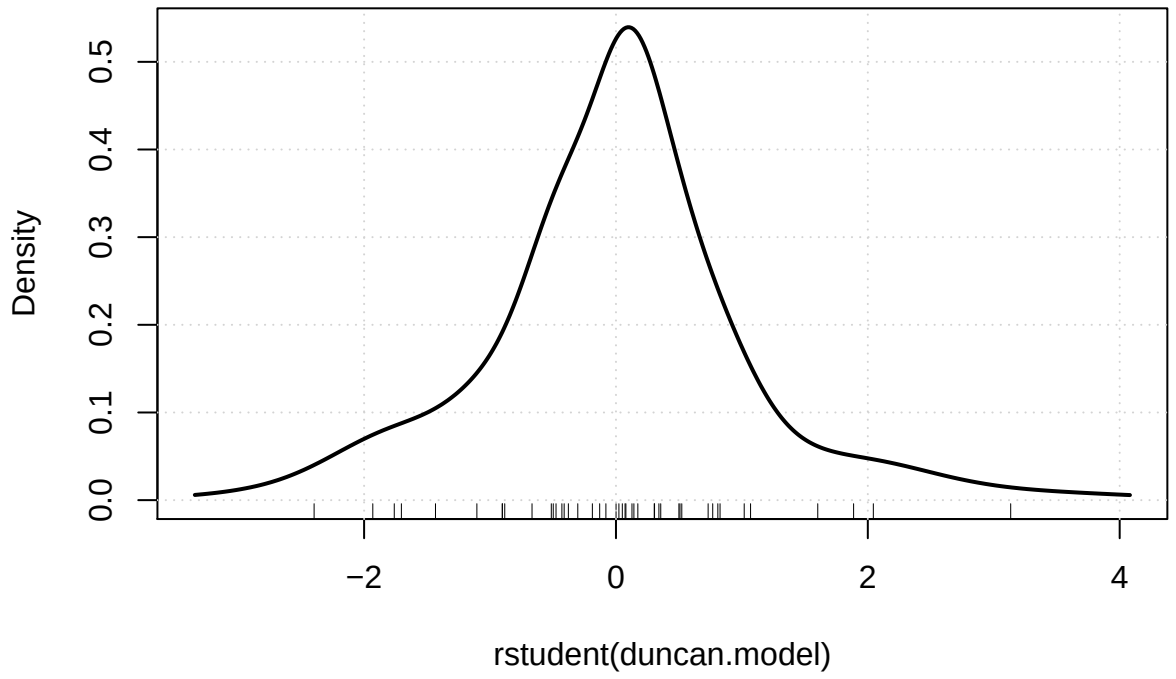
```
##
## Call:
## lm(formula = prestige ~ education + income, data = Duncan)
##
## Coefficients:
## (Intercept)    education       income
##     -6.0647       0.5458       0.5987
```

```
summary(duncan.model)
```

```
##
## Call:
## lm(formula = prestige ~ education + income, data = Duncan)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.538  -6.417   0.655   6.605  34.641
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.06466    4.27194  -1.420    0.163
## education    0.54583    0.09825   5.555 1.73e-06 ***
## income       0.59873    0.11967   5.003 1.05e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

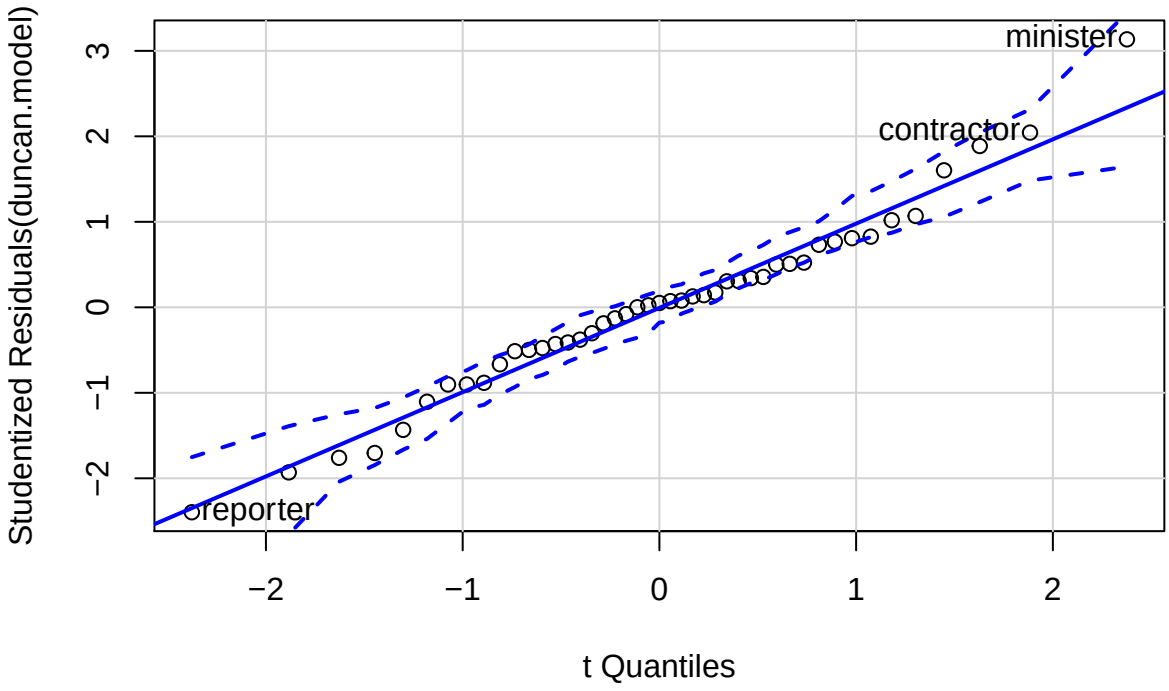
```
##
## Residual standard error: 13.37 on 42 degrees of freedom
## Multiple R-squared:  0.8282, Adjusted R-squared:   0.82
## F-statistic: 101.2 on 2 and 42 DF,  p-value: < 2.2e-16
```

```
densityPlot(rstudent(duncan.model))
```




```
set.seed(12345)

qqPlot(duncan.model, id=list(n=3))
```

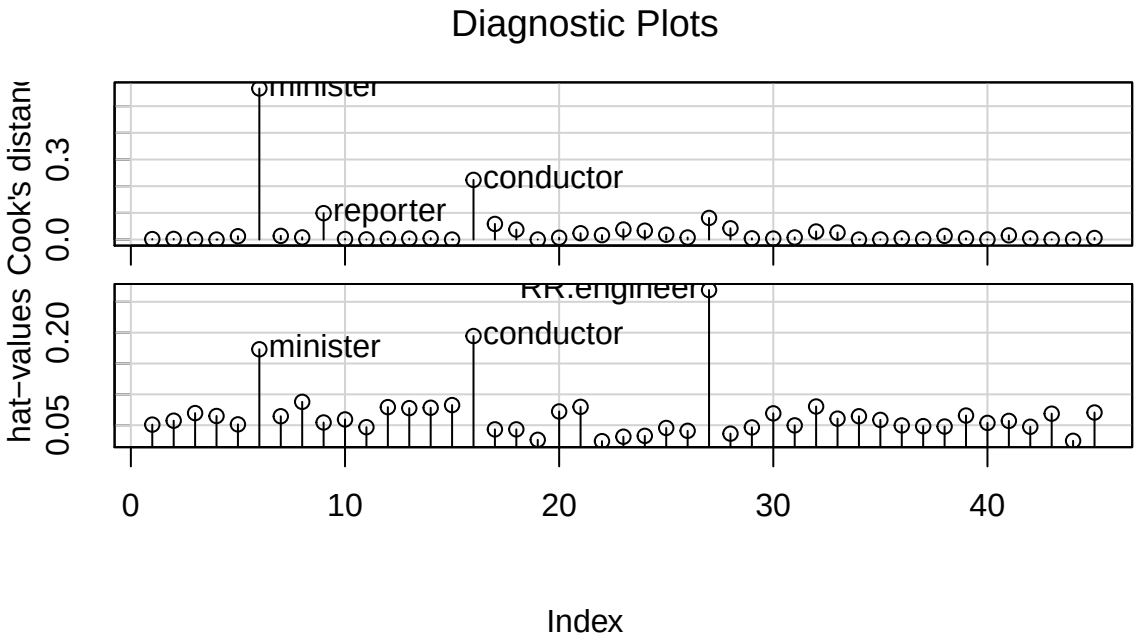



```
##   minister   reporter contractor
##          6          9         17
```

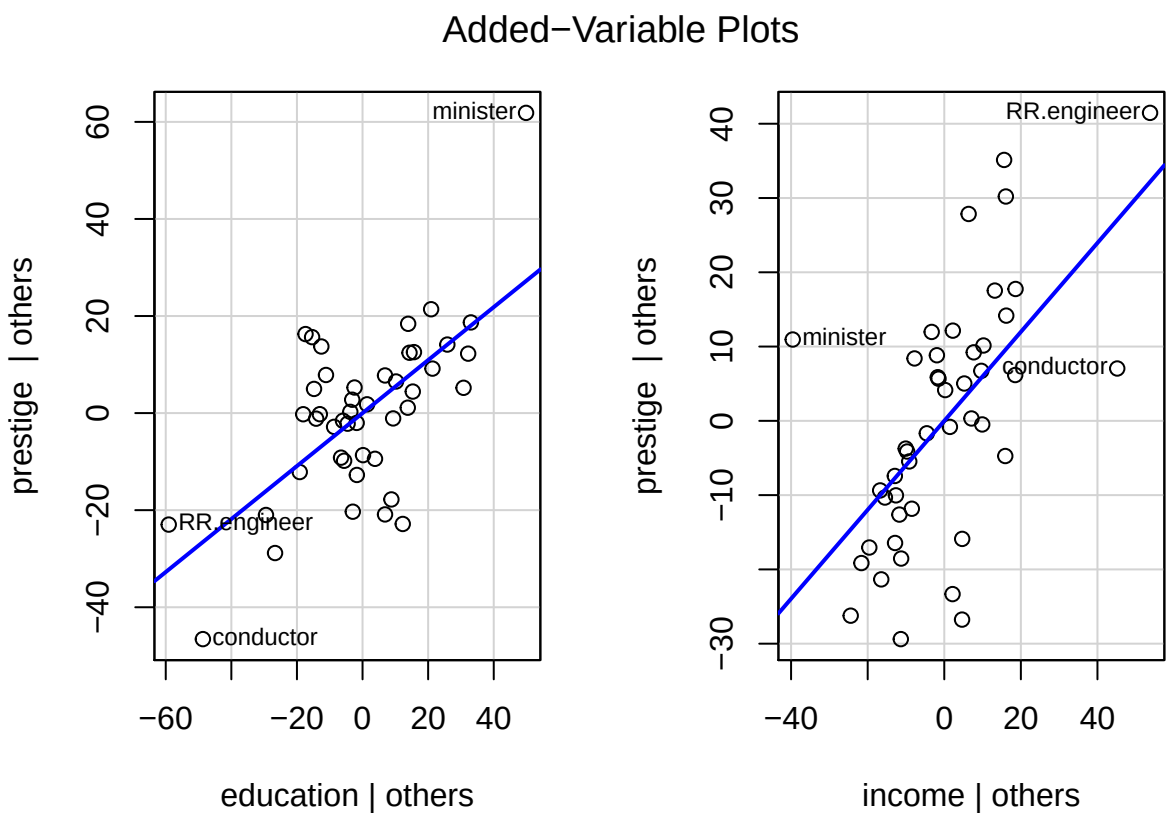
```
outlierTest(duncan.model)
```

```
## No Studentized residuals with Bonferroni p < 0.05
## Largest |rstudent|:
##          rstudent unadjusted p-value Bonferroni p
## minister 3.134519          0.0031772      0.14297
```

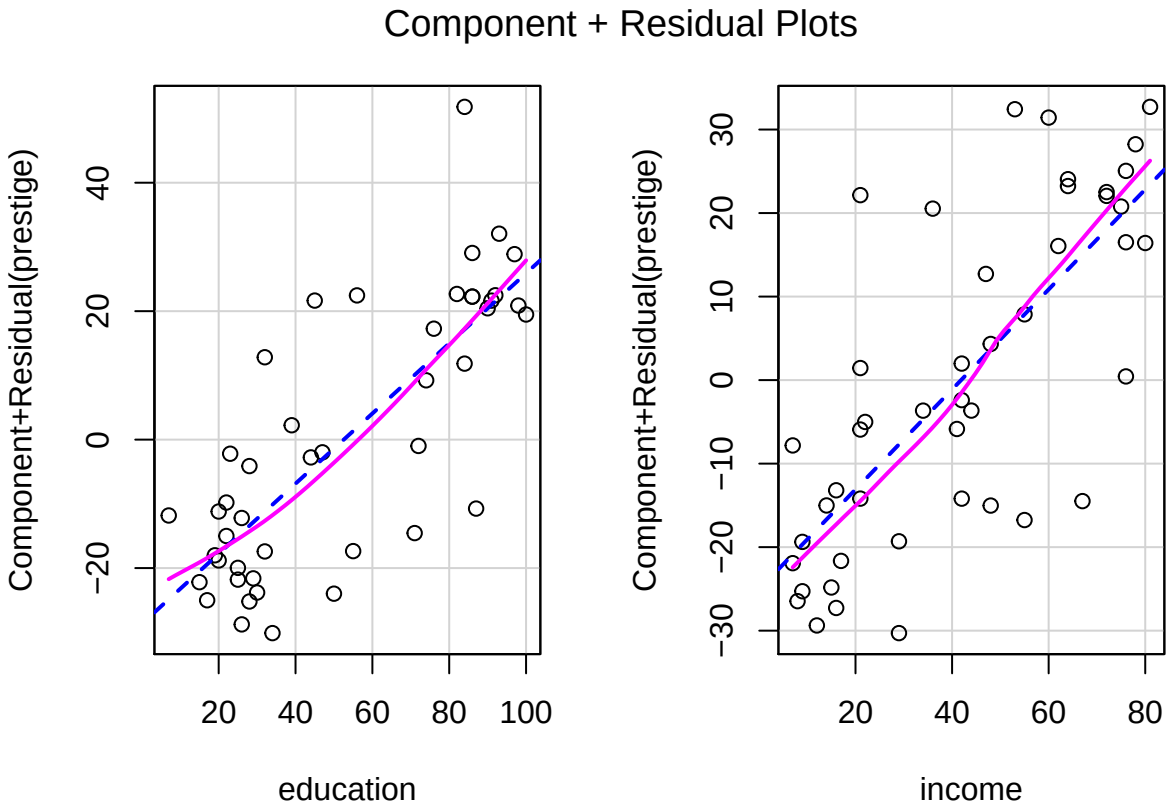
```
influenceIndexPlot(duncan.model, vars=c("Cook", "hat"),
    id=list(n=3))
```




```
avPlots(duncan.model,
    id=list(cex=0.75, n=3, method="mahal"))
```




```
crPlots(duncan.model)
```




```
ncvTest(duncan.model)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 0.3810967, Df = 1, p = 0.53702
```

```
ncvTest(duncan.model, var.formula= ~ income + education)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ income + education
## Chisquare = 0.6976023, Df = 2, p = 0.70553
```

```
whichNames(c("minister", "conductor"), Duncan)
```

```
##  minister conductor
##         6        16
```

```
duncan.model.2 <- update(duncan.model, subset=-c(6, 16))
summary(duncan.model.2)
```

```
##
## Call:
## lm(formula = prestige ~ education + income, data = Duncan, subset = -c(6,
##     16))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.612  -5.898   1.937   5.616  21.551
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.40899    3.65263  -1.755   0.0870 .
## education    0.33224    0.09875   3.364   0.0017 **
## income       0.86740    0.12198   7.111 1.31e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.42 on 40 degrees of freedom
## Multiple R-squared:  0.876,  Adjusted R-squared:  0.8698
## F-statistic: 141.3 on 2 and 40 DF,  p-value: < 2.2e-16
```

```
compareCoefs(duncan.model, duncan.model.2)
```

```
## Calls:
## 1: lm(formula = prestige ~ education + income, data = Duncan)
## 2: lm(formula = prestige ~ education + income, data = Duncan, subset = -c(6,
##    16))
##
##             Model 1 Model 2
## (Intercept)   -6.06   -6.41
## SE             4.27    3.65
##
## education    0.5458  0.3322
## SE           0.0983  0.0987
##
## income        0.599   0.867
## SE            0.120   0.122
##
```

```
summary(Duncan$type)
```

```
##   bc prof   wc
##   21   18    6
```

```
summary(Duncan$prestige)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    3.00   16.00   41.00   47.69   81.00   97.00
```

```
summary(Duncan)
```

```
##    type        income          education         prestige
##  bc  :21   Min.   : 7.00   Min.   :  7.00   Min.   : 3.00
##  prof:18   1st Qu.:21.00   1st Qu.: 26.00   1st Qu.:16.00
##  wc  : 6   Median :42.00   Median : 45.00   Median :41.00
##            Mean   :41.87   Mean   : 52.56   Mean   :47.69
##            3rd Qu.:64.00   3rd Qu.: 84.00   3rd Qu.:81.00
##            Max.   :81.00   Max.   :100.00   Max.   :97.00
```

```
summary(lm(prestige ~ education + income, data=Duncan))
```

```
##
## Call:
## lm(formula = prestige ~ education + income, data = Duncan)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -29.538  -6.417   0.655   6.605  34.641
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.06466    4.27194  -1.420    0.163
## education    0.54583    0.09825   5.555 1.73e-06 ***
## income       0.59873    0.11967   5.003 1.05e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.37 on 42 degrees of freedom
## Multiple R-squared:  0.8282, Adjusted R-squared:   0.82
## F-statistic: 101.2 on 2 and 42 DF,  p-value: < 2.2e-16
```

```
class(Duncan$type)
```

```
## [1] "factor"
```

```
class(Duncan$prestige)
```

```
## [1] "integer"
```

```
class(Duncan)
```

```
## [1] "data.frame"
```

```
duncan.model <- lm(prestige ~ education + income, data=Duncan)
class(duncan.model)
```

```
## [1] "lm"
```

```
summary
```

```
## function (object, ...)
## UseMethod("summary")
## <bytecode: 0x00000000120ce0e0>
## <environment: namespace:base>
```

```
args(summary.lm)
```

```
## function (object, correlation = FALSE, symbolic.cor = FALSE,
##     ...)
## NULL
```

```
mod.mroz <- glm(lfp ~ ., family=binomial, data=Mroz)
class(mod.mroz)
```

```
## [1] "glm" "lm"
```