ISLR CH3 Applied Exercises

```
auto.df = read.csv("/Users/shahrdadshadab/env/my-R-project/ISLR/Data/Auto.csv",
                   header=T, stringsAsFactors = F, na.strings = "?")

str(auto.df)
```

```
## 'data.frame':    397 obs. of  9 variables:
##  $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
##  $ cylinders   : int  8 8 8 8 8 8 8 8 8 8 ...
##  $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ horsepower  : int  130 165 150 150 140 198 220 215 225 190 ...
##  $ weight      : int  3504 3693 3436 3433 3449 4341 4354 4312 4425 3850 ...
##  $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
##  $ year        : int  70 70 70 70 70 70 70 70 70 70 ...
##  $ origin      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ name        : chr  "chevrolet chevelle malibu" "buick skylark 320" "plymouth satellite" "amc rebel
```

```
auto.lm = lm(mpg ~ horsepower, data = auto.df)
summary.lm = summary(auto.lm)
summary.lm
```

```
##
## Call:
## lm(formula = mpg ~ horsepower, data = auto.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.5710  -3.2592  -0.3435   2.7630  16.9240
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 39.935861   0.717499   55.66   <2e-16 ***
## horsepower  -0.157845   0.006446  -24.49   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.906 on 390 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.6059, Adjusted R-squared:  0.6049
## F-statistic: 599.7 on 1 and 390 DF,  p-value: < 2.2e-16
```

```
print("--------- Summry is S3 object of class summary.lm ------------")
```

```
## [1] "--------- Summry is S3 object of class summary.lm ------------"
```

```
print("attributes of summary")
```

```
## [1] "attributes of summary"
```

```r
str(attributes(summary.lm))
```

```
## List of 2
##  $ names: chr [1:12] "call" "terms" "residuals" "coefficients" ...
##  $ class: chr "summary.lm"
```

```r
sprintf("type of summary: %s ", typeof(summary.lm))
```

```
## [1] "type of summary: list "
```

```r
sprintf("names of summary:")
```

```
## [1] "names of summary:"
```

```r
names(summary.lm)
```

```
##  [1] "call"          "terms"         "residuals"     "coefficients"
##  [5] "aliased"       "sigma"         "df"            "r.squared"
##  [9] "adj.r.squared" "fstatistic"    "cov.unscaled"  "na.action"
```

```r
fstatistic <- summary.lm$fstatistic

"-------- Calculate overall p-value using F distribution-----"
```

```
## [1] "-------- Calculate overall p-value using F distribution-----"
```

```r
str(attributes(fstatistic))
```

```
## List of 1
##  $ names: chr [1:3] "value" "numdf" "dendf"
```

```r
sprintf("Type of fstatistic: %s", typeof(fstatistic))
```

```
## [1] "Type of fstatistic: double"
```

```r
stopifnot( fstatistic["value"] == fstatistic[[1]])
fstatistic["value"]
```

```
##    value
## 599.7177
```

```r
fstatistic[[1]]
```

```
## [1] 599.7177
```

```r
fstatisticValue <- fstatistic[["value"]]
fstatisticNumDegreesOfFreedom <- fstatistic[["numdf"]]
fstatisticDenDegreesOfFreedom <- fstatistic[["dendf"]]
overallPValue = pf(fstatisticValue, fstatisticNumDegreesOfFreedom,
                   fstatisticDenDegreesOfFreedom,lower.tail = FALSE)

"-------- Type differences --------------"
```

```
## [1] "-------- Type differences --------------"
```

```r
typeof(summary.lm[["r.squared"]]) # a double
```

```
## [1] "double"
```

```r
typeof(summary.lm["r.squared"]) # a list
```

```
## [1] "list"
```

```r
stopifnot(summary.lm[["r.squared"]] == summary.lm$r.squared)

rsquared <- summary.lm$r.squared
rse <- summary.lm$sigma # Clearly RSE is an estimate of population vriance
coefficients <- summary.lm$coefficients
sprintf("Type of coefficients: %s", typeof(coefficients))
```

```
## [1] "Type of coefficients: double"
```

```r
sprintf("type of r.squared in summary: %s", typeof(rsquared))
```

```
## [1] "type of r.squared in summary: double"
```

```r
sprintf("R squared: %f", rsquared)
```

```
## [1] "R squared: 0.605948"
```

```r
sprintf("RSE (Standard deviation from population regression line) = %f", rse)
```

```
## [1] "RSE (Standard deviation from population regression line) = 4.905757"
```

```r
typeof(summary.lm$sigma) # double
```

```
## [1] "double"
```

```r
typeof(summary.lm ["sigma"]) # list
```

```
## [1] "list"
```

```r
percentageError = summary.lm$sigma/mean(auto.df$mpg)
sprintf(" i) Yes: F-statistics %.4f > 1 and overall p-values %.4f < 0.05",
        fstatisticValue, overallPValue)
```

```
## [1] " i) Yes: F-statistics 599.7177 > 1 and overall p-values 0.0000 < 0.05"
```

```r
sprintf(" ii) Deviation from population regression line is: %.4f%%
        and variablity explained by horsepower: %.f%%",
        floor(percentageError*100), floor(rsquared*100))
```

```
## [1] " ii) Deviation from population regression line is: 20.0000% \n        and variablity explained
```

```r
sprintf(" iii) Reltionship is negative:")
```

```
## [1] " iii) Reltionship is negative:"
```

```r
coefficients
```

```
##               Estimate  Std. Error   t value      Pr(>|t|)
## (Intercept) 39.9358610 0.717498656  55.65984 1.220362e-187
## horsepower  -0.1578447 0.006445501 -24.48914  7.031989e-81
```

```r
sprintf("Confedence interval for coefficients: ")
```

```
## [1] "Confedence interval for coefficients: "
```

```r
confint(summary.lm)
```

```
##      2.5 % 97.5 %
```

```r
sprintf(" iv) Predict mpg associated with a horsepower of 98 and
        show predicted value, confidence interval")
```

```
## [1] " iv) Predict mpg associated with a horsepower of 98 and \n        show predicted value, confide
```

```
predict(auto.lm, data.frame(horsepower = c(98)) , interval = "confidence")
```

```
##        fit      lwr      upr
## 1 24.46708 23.97308 24.96108
```

```
sprintf(" Predict mpg associated with a horsepower of 98 and show predicted
        value, prediction interval")
```

```
## [1] " Predict mpg associated with a horsepower of 98 and show predicted \n        value, prediction i
```

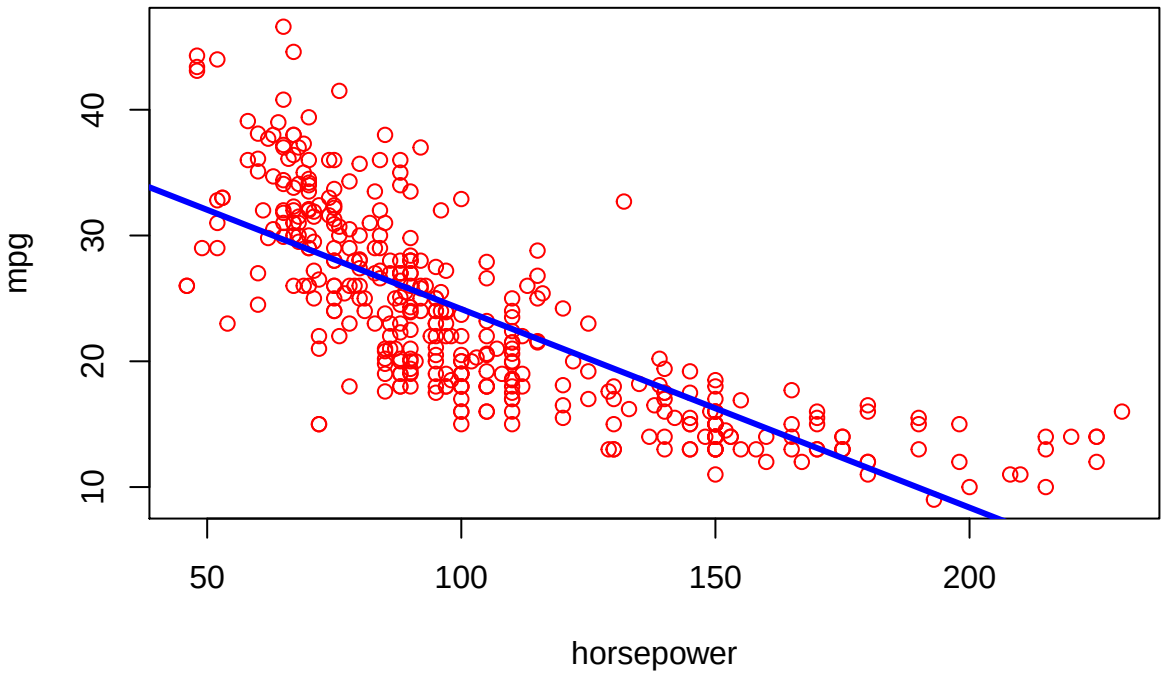
```
predict(auto.lm, data.frame(horsepower = c(98)) , interval = "prediction")
```

```
##        fit     lwr      upr
## 1 24.46708 14.8094 34.12476
```

```
sprintf("b) plot response and predictor:")
```

```
## [1] "b) plot response and predictor:"
```

```
plot(auto.df$mpg ~ auto.df$horsepower, col="red",xlab="horsepower",ylab="mpg") +
  abline(auto.lm, lwd=3, col="blue")
```

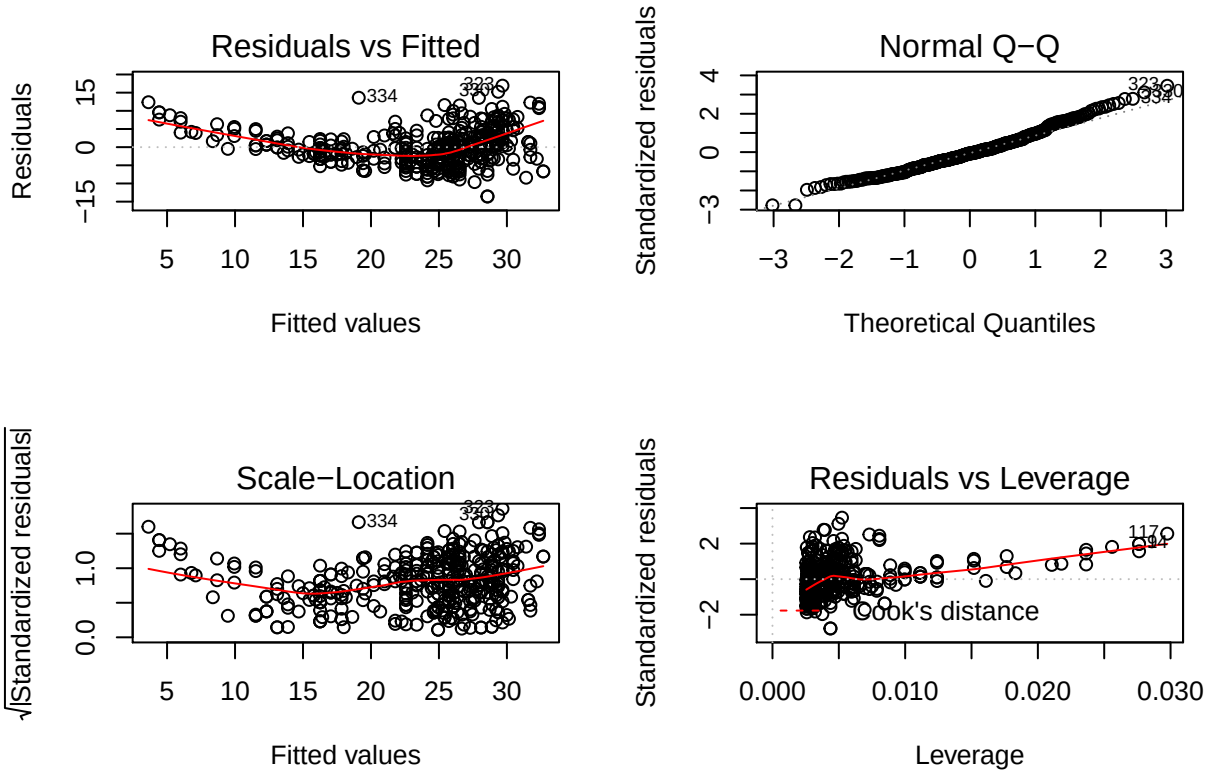



```
## integer(0)
```

```
sprintf("c) create diagnostic plots: ")
```

```
## [1] "c) create diagnostic plots: "
```

```
par(mfrow=c(2,2))
plot(auto.lm)
```




```
par(mfrow=c(1,1))
sprintf("Problems with the fit are : 1_ dependency is not linear,
        2_ heteroscedastisity, 3_ high leverage, 4_ high leverage outlier")
```

```
## [1] "Problems with the fit are : 1_ dependency is not linear, \n        2_ heteroscedastisity, 3_ hi
```

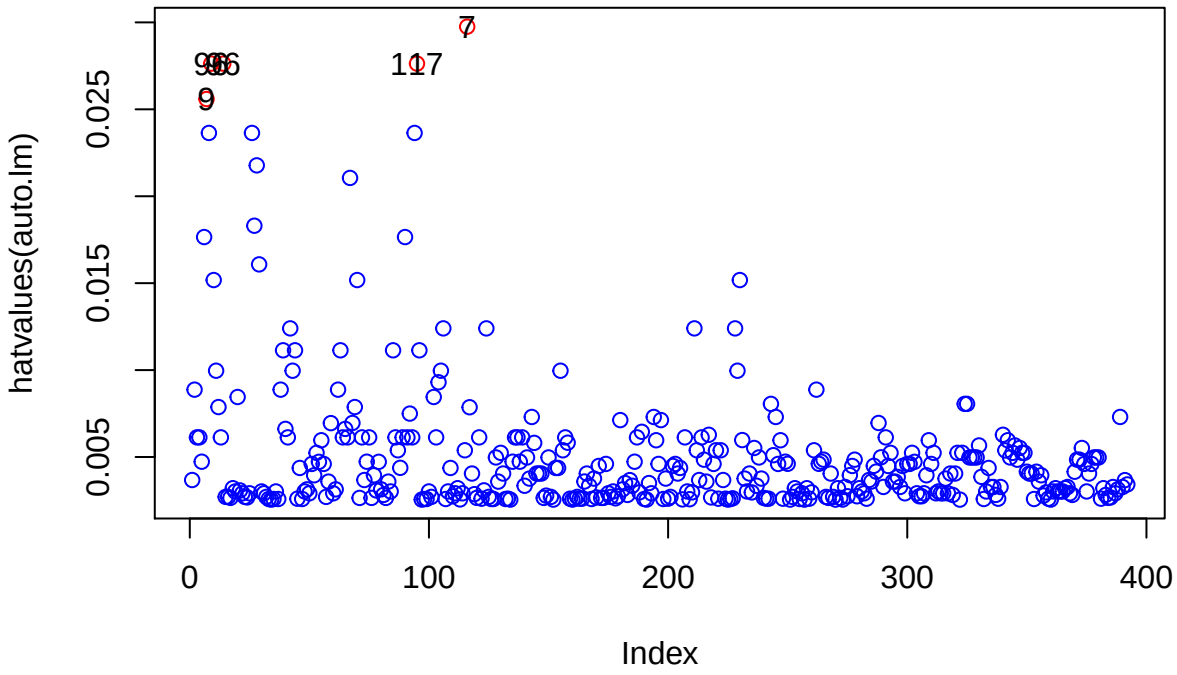
```
# find which observation in dataframe has rstudent > 3
sprintf (" rows in dataframe with rstudent >= 3:")
```

```
## [1] " rows in dataframe with rstudent >= 3:"
```

```
which(rstudent(auto.lm)>=3)
```

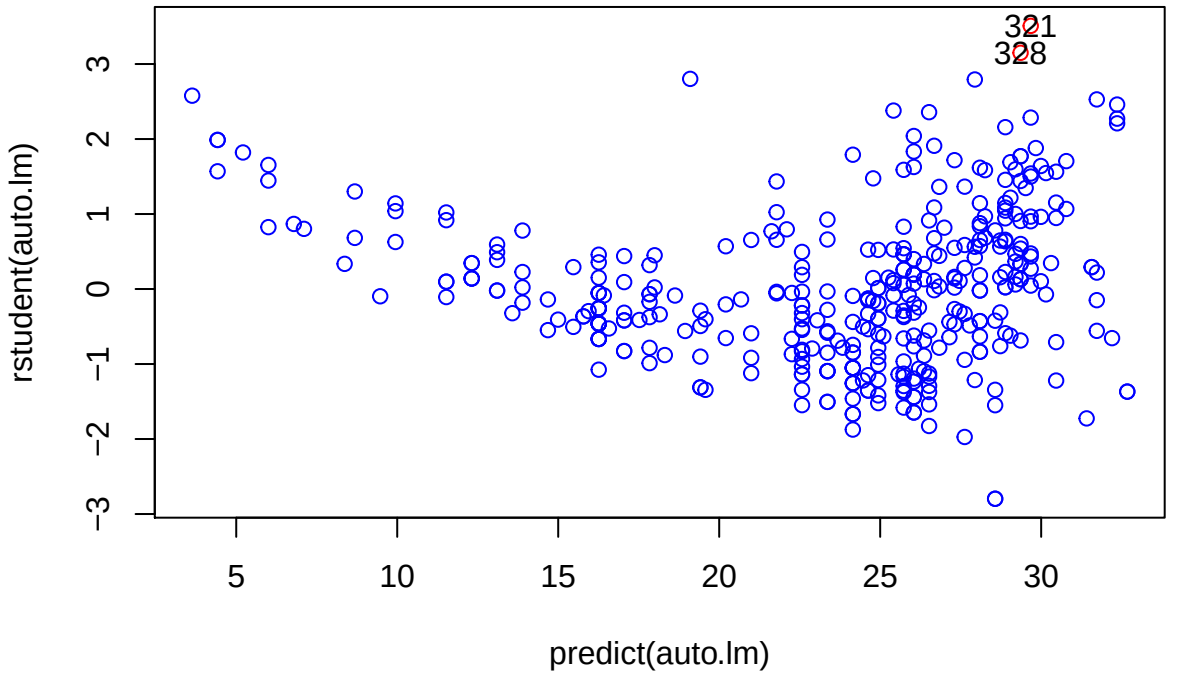
```
## 323 330
## 321 328
```

```
#plot high leverage points (hatmatrix)
n <- nrow(auto.df)
p <- ncol(auto.df)
plot(hatvalues(auto.lm), col=ifelse(hatvalues(auto.lm) > (p+1)/n,
                                    "red", "blue") ) +
  text(hatvalues(auto.lm),
       labels=ifelse(hatvalues(auto.lm) > (p+1)/n,
                     names(which(hatvalues(auto.lm) > (p+1)/n)), ""))
```



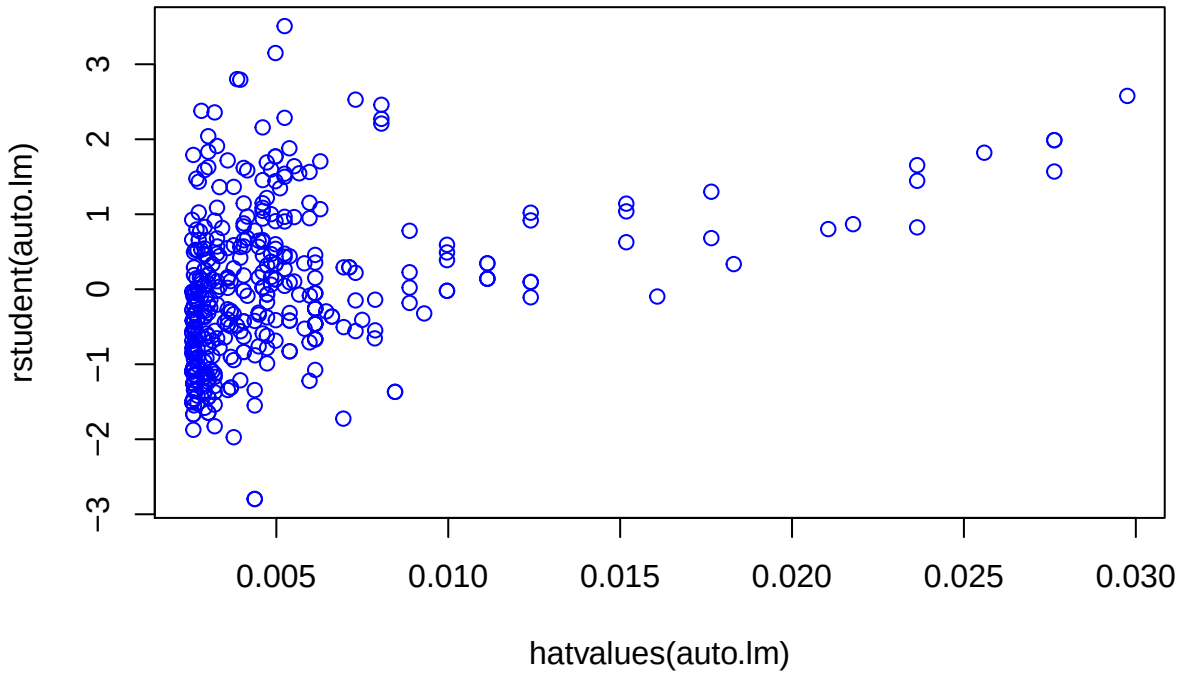

```
## integer(0)
```

```
#plot outliers against student residual (rstudent)
plot(predict(auto.lm), rstudent(auto.lm),
     col=ifelse(rstudent(auto.lm) >= 3, "red", "blue")) +
  text(predict(auto.lm), rstudent(auto.lm),
       labels = ifelse(rstudent(auto.lm) >= 3,
                       which(rstudent(auto.lm) >= 3), ""))
```




```
## integer(0)
```

```
#plot hatvalues against outliers
plot(hatvalues(auto.lm), rstudent(auto.lm),
     col=ifelse( hatvalues(auto.lm) > (p+1)/n || rstudent(auto.lm) >= 3,
                 "red", "blue") )
```




```
auto.df = read.csv("/Users/shahrdadshadab/env/my-R-project/ISLR/Data/Auto.csv",
                   header=T,  na.strings = "?")

sprintf("a) scatter plot matrix")
```

```
## [1] "a) scatter plot matrix"
```

```
my_cols <- c("#00AFBB", "#E7B800", "#FC4E07")
pairs(auto.df, pch = 19,  cex = 0.5,
      col = my_cols[auto.df$origin],
      lower.panel=NULL)
# pairs(auto.df, pch = 19, lower.panel = NULL)

sprintf("b) compute the matrix of correlations between the variables:")
```

```
## [1] "b) compute the matrix of correlations between the variables:"
```

```
# First remove "name" column from dataframe
drops <- c("name")
newDf <- auto.df [, !(names(auto.df) %in% drops)]
str(newDf)
```

```
## 'data.frame':    397 obs. of  8 variables:
##  $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
##  $ cylinders   : int  8 8 8 8 8 8 8 8 8 8 ...
##  $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ horsepower  : int  130 165 150 150 140 198 220 215 225 190 ...
##  $ weight      : int  3504 3693 3436 3433 3449 4341 4354 4312 4425 3850 ...
##  $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
```

```
##  $ year         : int  70 70 70 70 70 70 70 70 70 70 ...
##  $ origin       : int  1 1 1 1 1 1 1 1 1 1 ...
```

```
# Now find a subset of records that have at least one NA
dfSubsetWithNa <- newDf[rowSums(is.na(newDf)) > 0,]
head(dfSubsetWithNa)
```

```
##      mpg cylinders displacement horsepower weight acceleration year origin
## 33  25.0         4           98         NA   2046         19.0   71      1
## 127 21.0         6          200         NA   2875         17.0   74      1
## 331 40.9         4           85         NA   1835         17.3   80      2
## 337 23.6         4          140         NA   2905         14.3   80      1
## 355 34.5         4          100         NA   2320         15.8   81      2
```

```
# Check a particular column that has NA and include the record
dfSubsetWithNaInOneCol <- newDf[is.na(newDf$horsepower), ]
head(dfSubsetWithNaInOneCol)
```

```
##      mpg cylinders displacement horsepower weight acceleration year origin
## 33  25.0         4           98         NA   2046         19.0   71      1
## 127 21.0         6          200         NA   2875         17.0   74      1
## 331 40.9         4           85         NA   1835         17.3   80      2
## 337 23.6         4          140         NA   2905         14.3   80      1
## 355 34.5         4          100         NA   2320         15.8   81      2
```

```
# get a subset of records in dataframe with no NA in any column:
dfSubsetWithNoNa <- newDf[rowSums(is.na(newDf)) == 0, ]
head(dfSubsetWithNoNa)
```

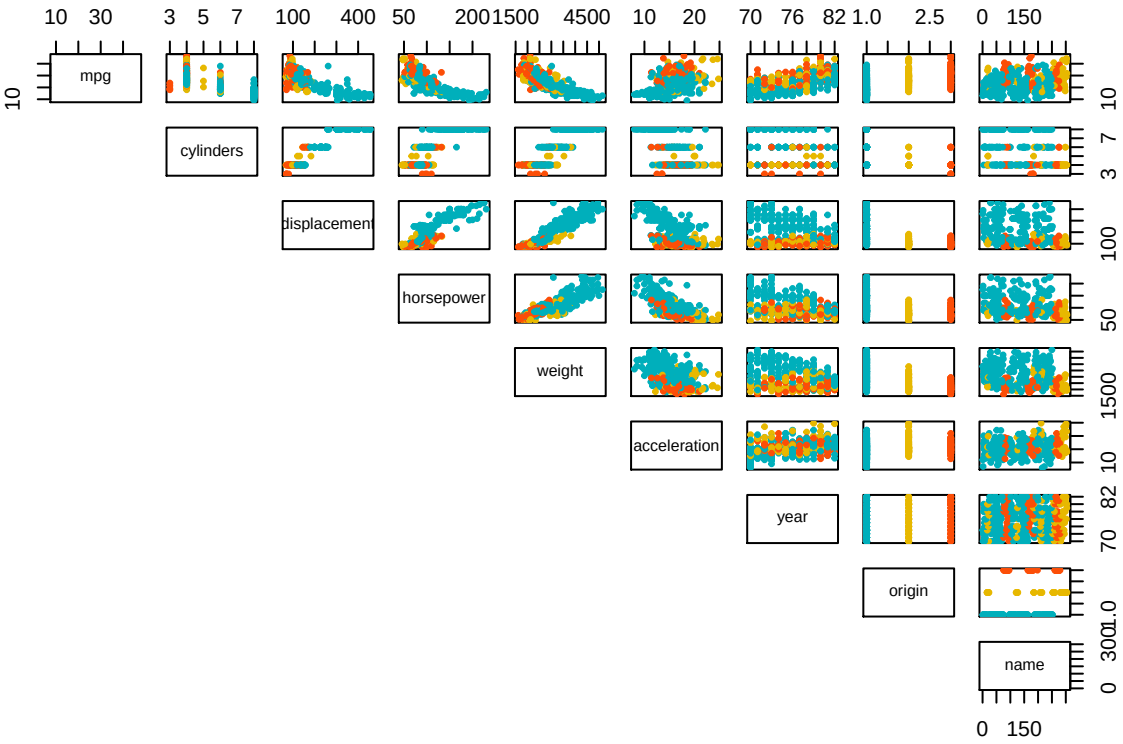
```
##   mpg cylinders displacement horsepower weight acceleration year origin
## 1  18         8          307        130   3504         12.0   70      1
## 2  15         8          350        165   3693         11.5   70      1
## 3  18         8          318        150   3436         11.0   70      1
## 4  16         8          304        150   3433         12.0   70      1
## 5  17         8          302        140   3449         10.5   70      1
## 6  15         8          429        198   4341         10.0   70      1
```

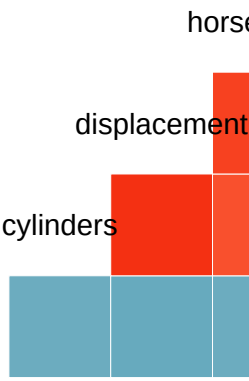
```
cor(dfSubsetWithNoNa[,])
```

```
##                     mpg  cylinders displacement horsepower     weight
## mpg           1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442
## cylinders    -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273
## displacement -0.8051269  0.9508233    1.0000000  0.8972570  0.9329944
## horsepower   -0.7784268  0.8429834    0.8972570  1.0000000  0.8645377
## weight       -0.8322442  0.8975273    0.9329944  0.8645377  1.0000000
## acceleration  0.4233285 -0.5046834   -0.5438005 -0.6891955 -0.4168392
## year          0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199
## origin        0.5652088 -0.5689316   -0.6145351 -0.4551715 -0.5850054
##              acceleration       year     origin
## mpg             0.4233285  0.5805410  0.5652088
## cylinders      -0.5046834 -0.3456474 -0.5689316
## displacement   -0.5438005 -0.3698552 -0.6145351
## horsepower     -0.6891955 -0.4163615 -0.4551715
## weight         -0.4168392 -0.3091199 -0.5850054
## acceleration    1.0000000  0.2903161  0.2127458
## year            0.2903161  1.0000000  0.1815277
## origin          0.2127458  0.1815277  1.0000000
```

```
# use GGally to visualize the correlation between all predictors in
GGally::ggcorr(dfSubsetWithNoNa)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```






```
# fit a model
df.lm <- lm (mpg ~ ., data = dfSubsetWithNoNa)
summary.ml <- summary(df.lm)

# overall p-value
fstatistic = summary.ml[["fstatistic"]]
fstatisticValue <- fstatistic[["value"]]
fstatisticNumDegreesOfFreedom <- fstatistic[["numdf"]]
fstatisticDenDegreesOfFreedom <- fstatistic[["dendf"]]
overallPValue = pf(fstatisticValue, fstatisticNumDegreesOfFreedom,
                   fstatisticDenDegreesOfFreedom,lower.tail = FALSE)


sprintf("i) Yes because: F-statistics %.4f,  p-value: %.4f < 2.2e-16: ",
        fstatisticValue, overallPValue)
```

```
## [1] "i) Yes because: F-statistics 252.4280,  p-value: 0.0000 < 2.2e-16: "
```

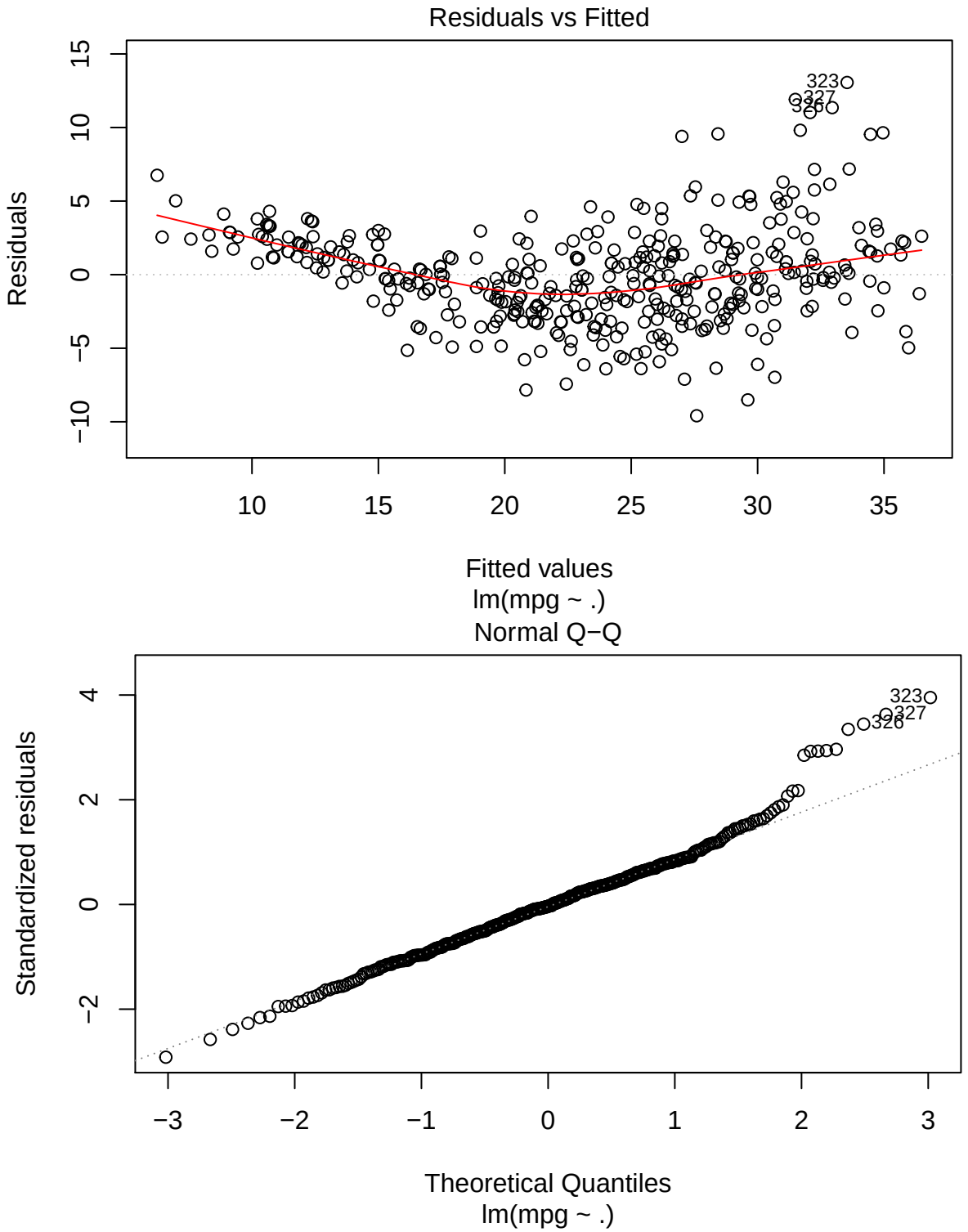
```
sprintf("ii) displacement, weight, year, origin")
```

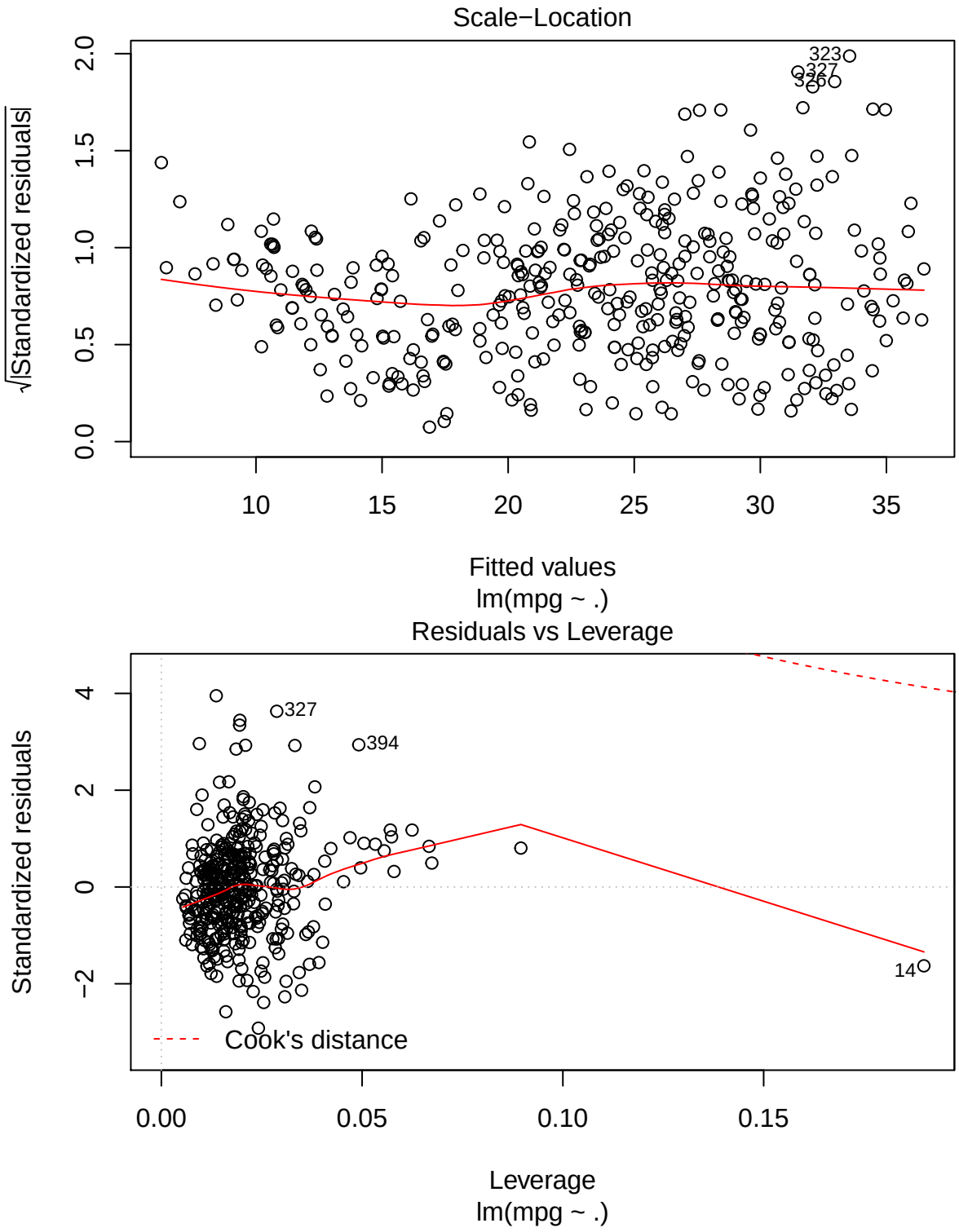
```
## [1] "ii) displacement, weight, year, origin"
```

```
sprintf("iii) miles per galon increases as year goes by")
```

```
## [1] "iii) miles per galon increases as year goes by"
```

```
# diagnostic plots for linear regression fit:
plot(df.lm)
```






```
sprintf("d) nonlinearity and heteroschedasticity")
```

```
## [1] "d) nonlinearity and heteroschedasticity"
```

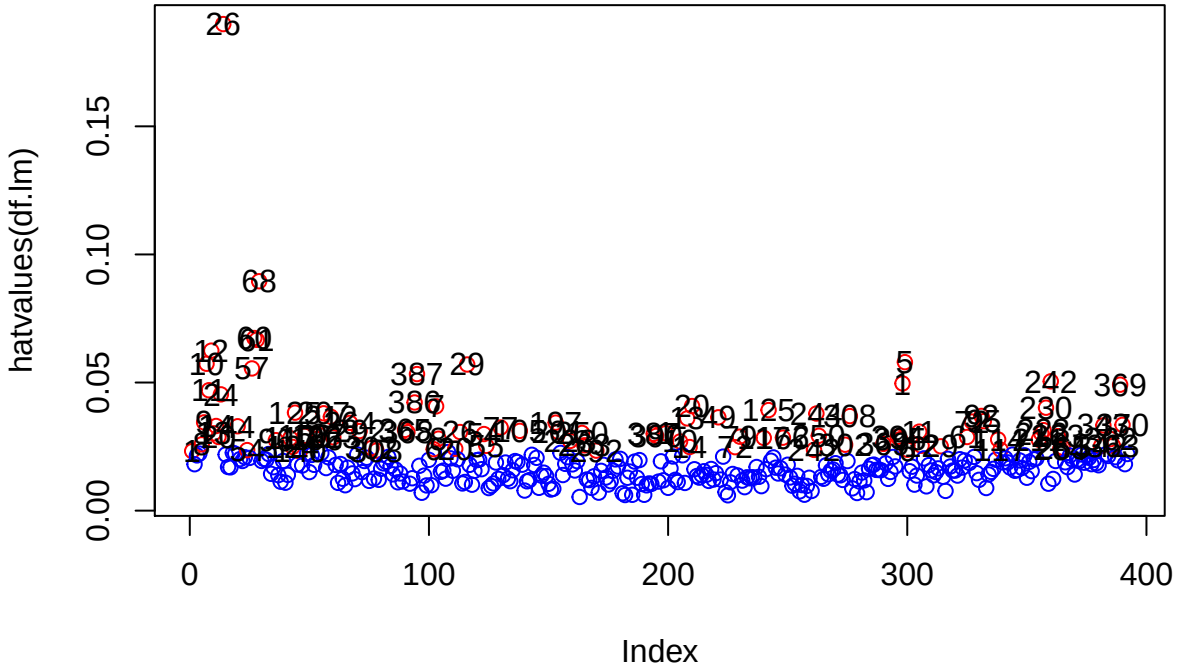
```
sprintf(" High leverage points are rows 327 and 394
        because there are above cook's line")
```

```
## [1] " High leverage points are rows 327 and 394\n        because there are above cook's line"
```

```
sprintf(" Also 323 and 327 are the outliers, thus 327
        is high levrage and outlier")
```

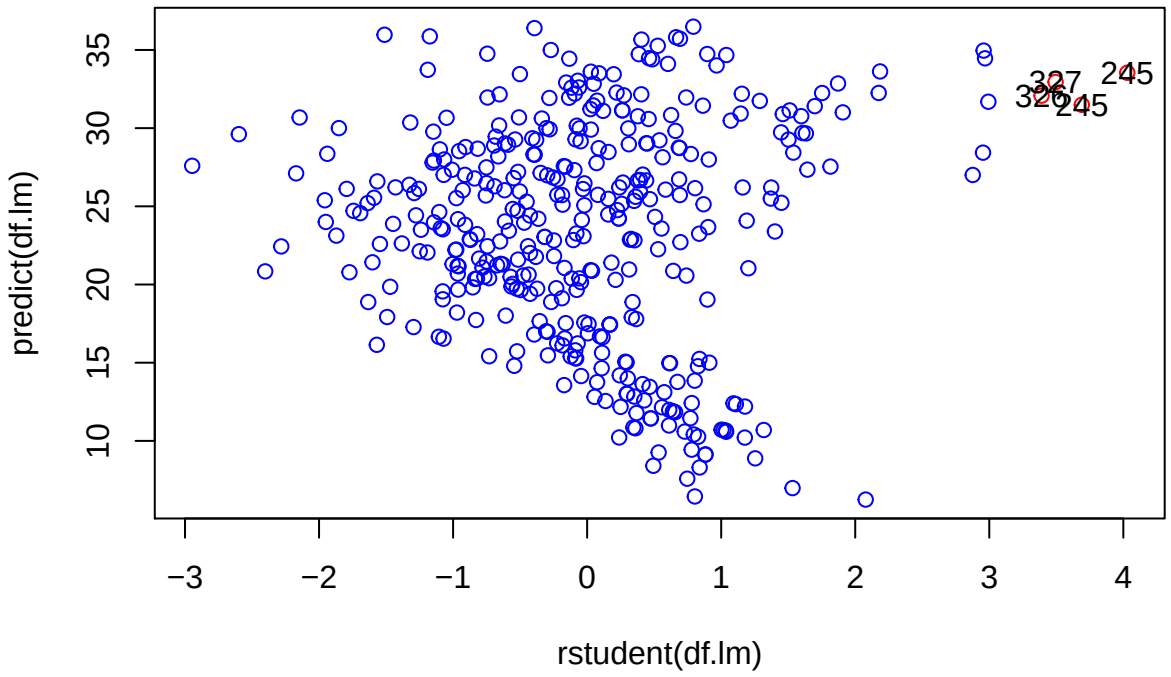
```
## [1] " Also 323 and 327 are the outliers, thus 327 \n        is high levrage and outlier"
```

```
# plot high leverage points
n <- nrow(dfSubsetWithNoNa)
p <- ncol(dfSubsetWithNoNa)
plot(hatvalues(df.lm), col=ifelse(hatvalues(df.lm) > (p+1)/n, "red", "blue"))+
  text(hatvalues(df.lm), labels=ifelse(hatvalues(df.lm) > (p+1)/n,
                                       names(which(hatvalues(df.lm) > (p+1)/n)),
                                       ""))
```




```
## integer(0)
```

```
# plot outliers
plot(rstudent(df.lm), predict(df.lm), col=ifelse(rstudent(df.lm)>3,
                                                 "red", "blue"))+
  text(rstudent(df.lm), predict(df.lm),
       labels = ifelse(rstudent(df.lm)>3,names(which(rstudent(df.lm)>3)), ""))
```




```
## integer(0)
```

```
sprintf("e) find all possible interactions first")
```

```
## [1] "e) find all possible interactions first"
```

```
"fit a model with all interactions"
```

```
## [1] "fit a model with all interactions"
```

```
df.lm2 <- lm (mpg ~ .^2, data = dfSubsetWithNoNa)
summary.ml2 <- summary(df.lm2)
summary.ml2
```

```
##
## Call:
## lm(formula = mpg ~ .^2, data = dfSubsetWithNoNa)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.6303 -1.4481  0.0596  1.2739 11.1386
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 3.548e+01  5.314e+01   0.668  0.50475
## cylinders                   6.989e+00  8.248e+00   0.847  0.39738
## displacement               -4.785e-01  1.894e-01  -2.527  0.01192 *
## horsepower                  5.034e-01  3.470e-01   1.451  0.14769
## weight                      4.133e-03  1.759e-02   0.235  0.81442
## acceleration               -5.859e+00  2.174e+00  -2.696  0.00735 **
## year                        6.974e-01  6.097e-01   1.144  0.25340
## origin                     -2.090e+01  7.097e+00  -2.944  0.00345 **
## cylinders:displacement     -3.383e-03  6.455e-03  -0.524  0.60051
## cylinders:horsepower        1.161e-02  2.420e-02   0.480  0.63157
## cylinders:weight            3.575e-04  8.955e-04   0.399  0.69000
```

```
## cylinders:acceleration     2.779e-01  1.664e-01   1.670  0.09584 .
## cylinders:year            -1.741e-01  9.714e-02  -1.793  0.07389 .
## cylinders:origin           4.022e-01  4.926e-01   0.816  0.41482
## displacement:horsepower   -8.491e-05  2.885e-04  -0.294  0.76867
## displacement:weight        2.472e-05  1.470e-05   1.682  0.09342 .
## displacement:acceleration -3.479e-03  3.342e-03  -1.041  0.29853
## displacement:year          5.934e-03  2.391e-03   2.482  0.01352 *
## displacement:origin        2.398e-02  1.947e-02   1.232  0.21875
## horsepower:weight         -1.968e-05  2.924e-05  -0.673  0.50124
## horsepower:acceleration   -7.213e-03  3.719e-03  -1.939  0.05325 .
## horsepower:year           -5.838e-03  3.938e-03  -1.482  0.13916
## horsepower:origin          2.233e-03  2.930e-02   0.076  0.93931
## weight:acceleration        2.346e-04  2.289e-04   1.025  0.30596
## weight:year               -2.245e-04  2.127e-04  -1.056  0.29182
## weight:origin             -5.789e-04  1.591e-03  -0.364  0.71623
## acceleration:year          5.562e-02  2.558e-02   2.174  0.03033 *
## acceleration:origin        4.583e-01  1.567e-01   2.926  0.00365 **
## year:origin                1.393e-01  7.399e-02   1.882  0.06062 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.695 on 363 degrees of freedom
## Multiple R-squared:  0.8893, Adjusted R-squared:  0.8808
## F-statistic: 104.2 on 28 and 363 DF,  p-value: < 2.2e-16
```

```
# Let's find statistically significant  coefficients with p-value less than 0.05
library(broom)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
coeffs <- tidy(df.lm2) %>%
  select(term, estimate, std.error, statistic, p.value) %>%
  filter(p.value <= 0.05)

coeffs
```

```
## # A tibble: 6 x 5
##   term                 estimate std.error statistic p.value
##   <chr>                   <dbl>     <dbl>     <dbl>   <dbl>
## 1 displacement         -0.479     0.189       -2.53 0.0119
## 2 acceleration         -5.86      2.17        -2.70 0.00735
## 3 origin              -20.9       7.10        -2.94 0.00345
## 4 displacement:year     0.00593   0.00239      2.48 0.0135
## 5 acceleration:year     0.0556    0.0256       2.17 0.0303
## 6 acceleration:origin   0.458     0.157        2.93 0.00365
```

```r
sprintf("f) try different transformation on statistically significant variables")
```

```
## [1] "f) try different transformation on statistically significant variables"
```

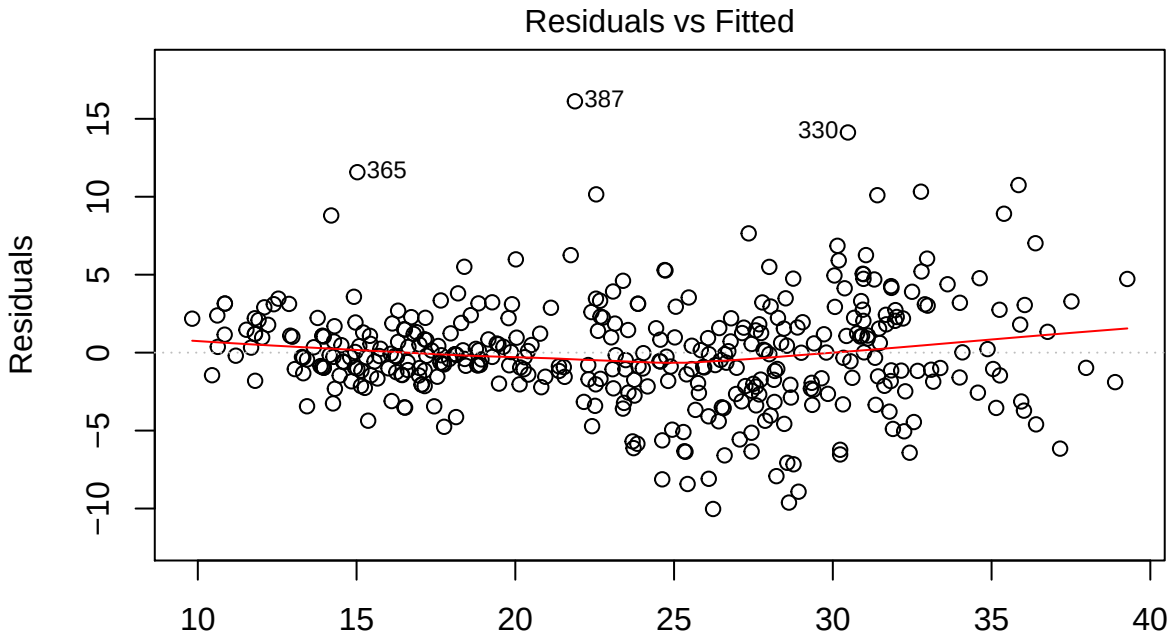
```r
"Fit a model with only those statistically significant variables"
```

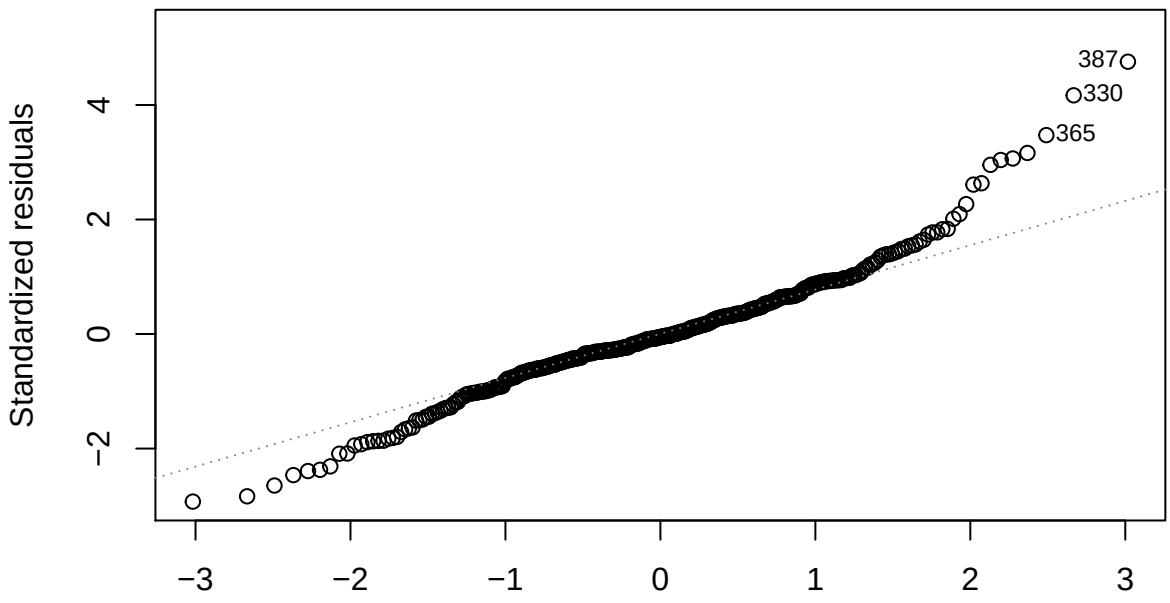
```
## [1] "Fit a model with only those statistically significant variables"
```

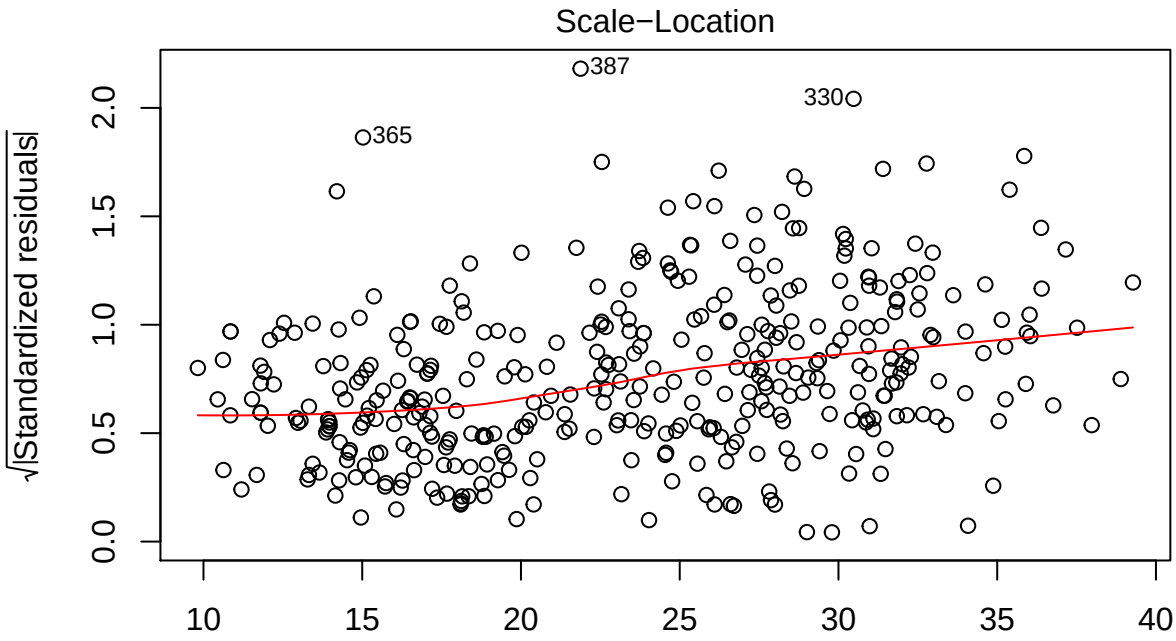
```r
df.lm3 <- lm (mpg ~ displacement + acceleration + origin + displacement:year +
                acceleration:year + acceleration:origin, data = dfSubsetWithNoNa)
summary(df.lm3)
```

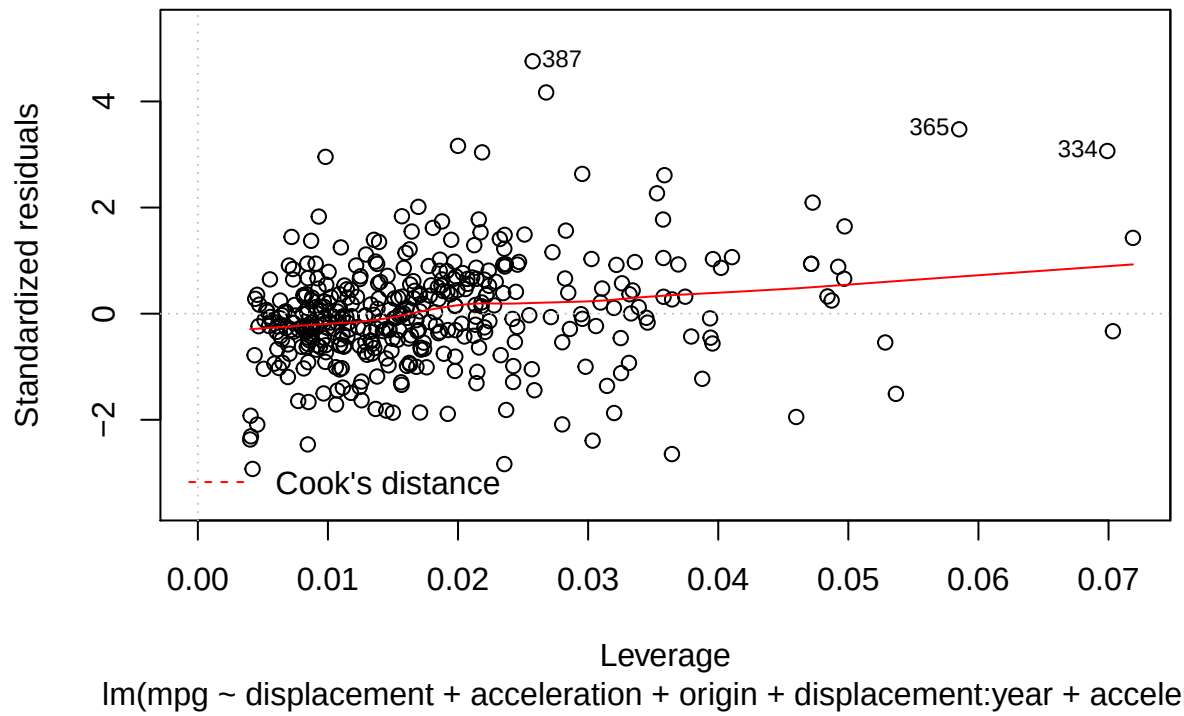
```
##
## Call:
## lm(formula = mpg ~ displacement + acceleration + origin + displacement:year +
##     acceleration:year + acceleration:origin, data = dfSubsetWithNoNa)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.0285  -1.7600  -0.1815   1.7881  16.1226
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          5.262e+01  3.325e+00  15.825  < 2e-16 ***
## displacement         7.006e-02  3.345e-02   2.095 0.036859 *
## acceleration        -6.220e+00  4.173e-01 -14.905  < 2e-16 ***
## origin              -1.109e+01  1.736e+00  -6.388 4.85e-10 ***
## displacement:year   -1.683e-03  4.489e-04  -3.748 0.000205 ***
## acceleration:year    6.554e-02  5.463e-03  11.997  < 2e-16 ***
## acceleration:origin  7.361e-01  1.048e-01   7.026 9.69e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.434 on 385 degrees of freedom
## Multiple R-squared:  0.8094, Adjusted R-squared:  0.8064
## F-statistic: 272.5 on 6 and 385 DF,  p-value: < 2.2e-16
```

```r
# plot diagnostic graphs
plot(df.lm3)
```

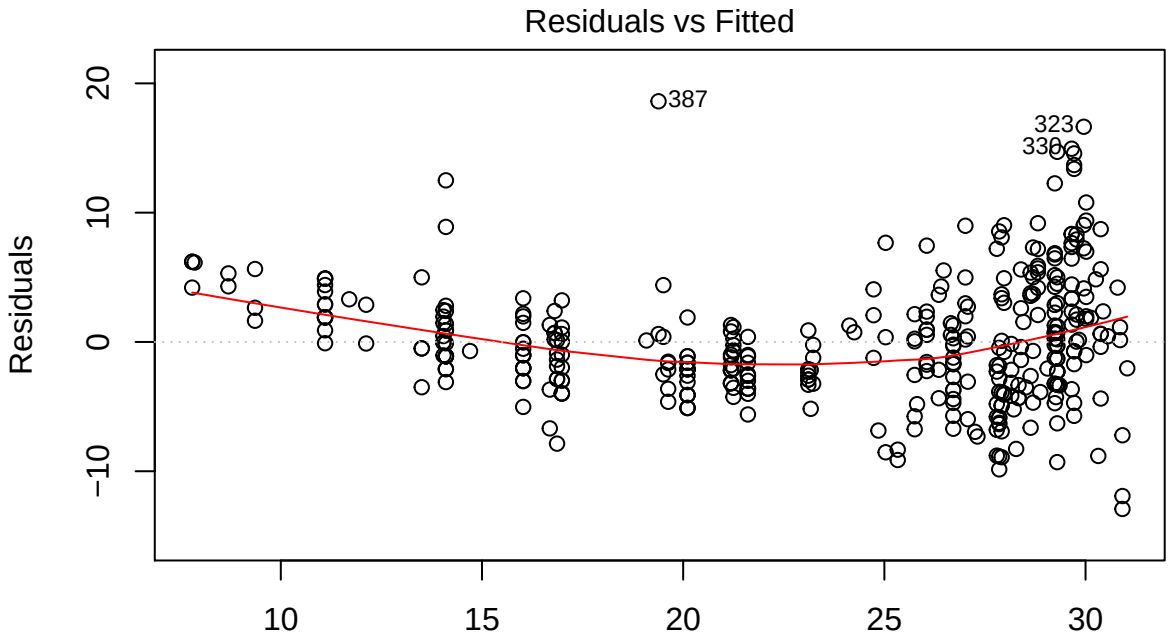

Residuals vs Fitted
387
330
365
Residuals
15
10
5
0
−5
−10
10
15
20
25
30
35
40
Fitted values
lm(mpg ~ displacement + acceleration + origin + displacement:year + acceler ...

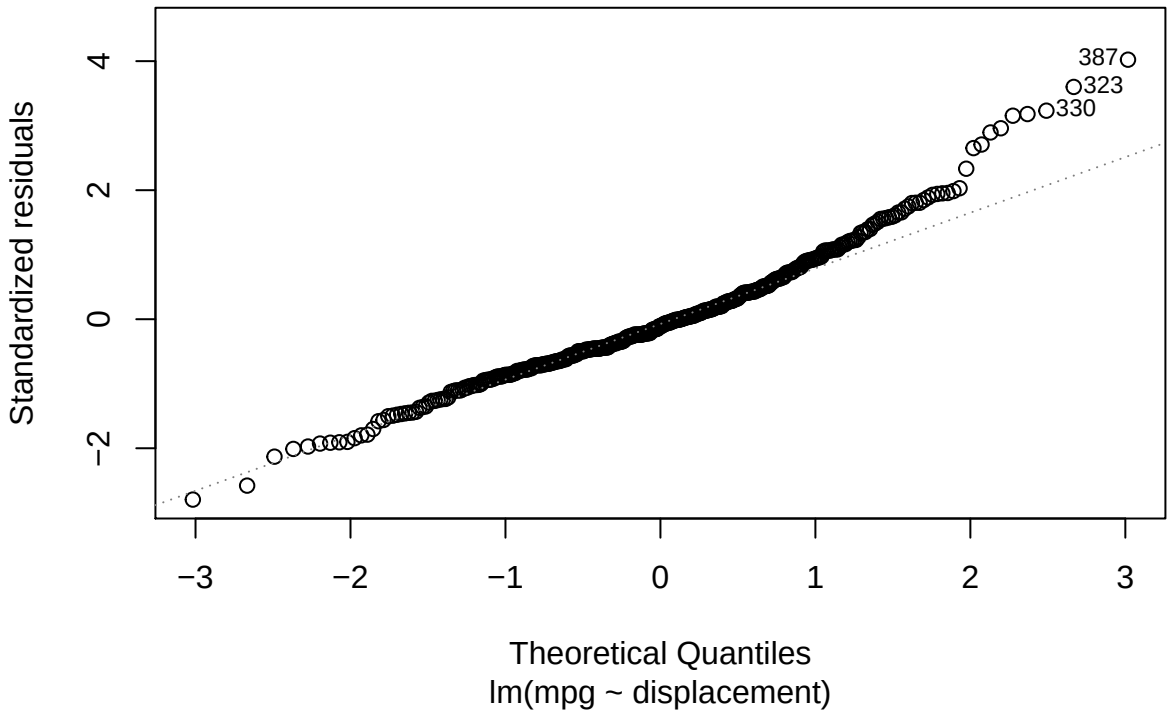

Normal Q−Q
387
330
365
Standardized residuals
4
2
0
−2
−3
−2
−1
0
1
2
3
Theoretical Quantiles
lm(mpg ~ displacement + acceleration + origin + displacement:year + acceler ...

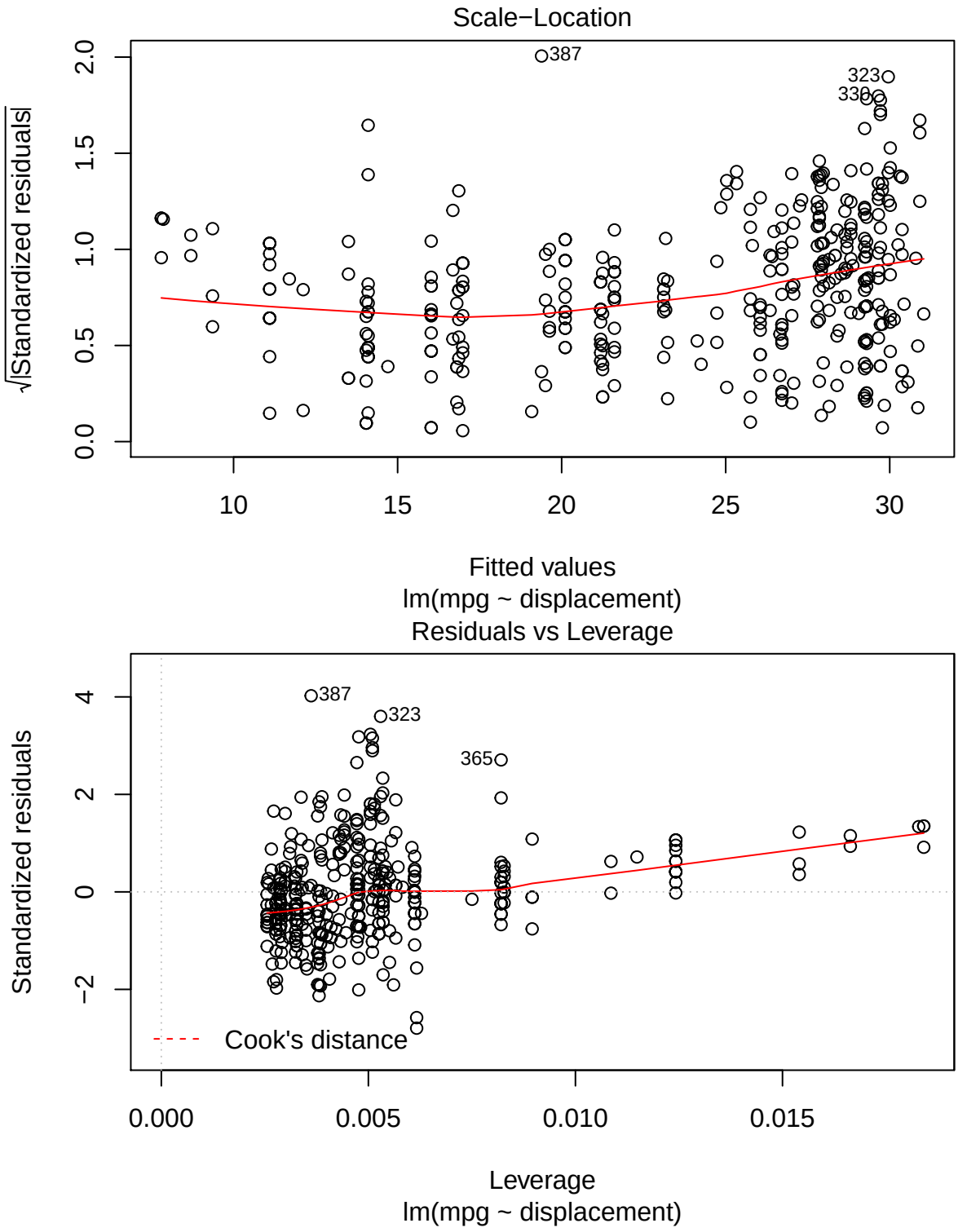





```r
# first regress mpg over displacement and plot fitted vs residual to
# see if there is any bend shape
mpg_displacement.lm <- lm(mpg ~ displacement, data = dfSubsetWithNoNa)
plot(mpg_displacement.lm)
```

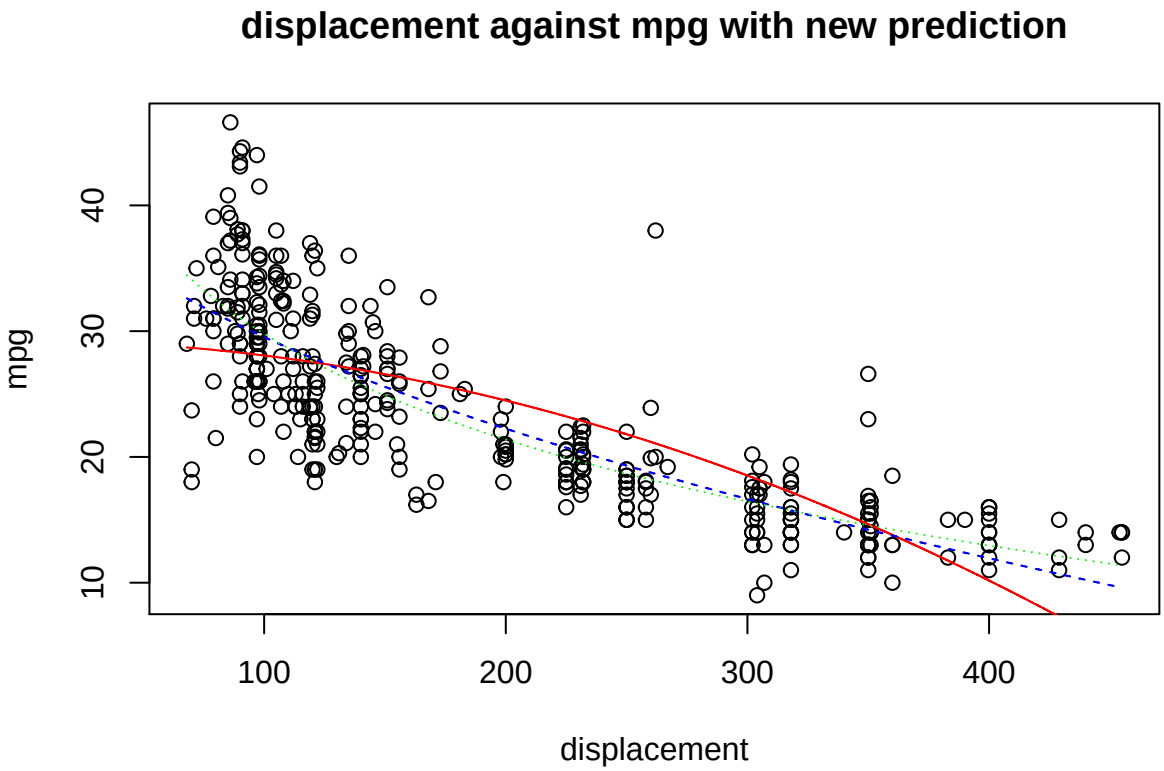

Residuals vs Fitted
Residuals
Fitted values
lm(mpg ~ displacement)
387
323
330
20
10
0
−10
10
15
20
25
30


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ displacement)
387
323
330
4
2
0
−2
−3
−2
−1
0
1
2
3




```
# clearly we can see a curve shape and heteroscedasticity, thus let's use
# displacement^2 to see any improvement

mpg_displacementSquared.lm <- lm(mpg ~ I(displacement^2), data = dfSubsetWithNoNa)
mpg_displacementSqrt.lm <- lm(mpg ~ I(displacement^0.5), data = dfSubsetWithNoNa)
mpg_displacementLog.lm <- lm(mpg ~ log1p(displacement), data = dfSubsetWithNoNa)
```

```r
displacementMin <- min(dfSubsetWithNoNa[["displacement"]])
displacementMax <- max(dfSubsetWithNoNa[["displacement"]])
displacementvalues  <- seq(displacementMin, displacementMax, 0.01)

newDisplacementDf <- data.frame(displacement=c(displacementvalues))

# Now predict new mpg for new displacement values for displacement squared
new_mpg_for_squared <- predict(mpg_displacementSquared.lm, newDisplacementDf)
new_mpg_for_sqrt <- predict(mpg_displacementSqrt.lm, newDisplacementDf)
new_mpg_for_log <- predict(mpg_displacementLog.lm, newDisplacementDf)

plot( dfSubsetWithNoNa[["displacement"]], dfSubsetWithNoNa[["mpg"]],
      xlab = "displacement", ylab="mpg",
      main = "displacement against mpg with new prediction")+
  lines(displacementvalues, new_mpg_for_squared, col="red") +
  lines(displacementvalues, new_mpg_for_sqrt, col="blue",lty=2) +
  lines(displacementvalues, new_mpg_for_log, col="green",lty=3)
```




```
## integer(0)
```

```r
# change displacement -> displacement^2 improve f statistics and Rsquered
df.lm4 <- lm (mpg ~ I(displacement^2) + acceleration + origin +
                displacement:year + acceleration:year + acceleration:origin,
              data = dfSubsetWithNoNa)
summary(df.lm4)
```

```
##
## Call:
## lm(formula = mpg ~ I(displacement^2) + acceleration + origin +
##     displacement:year + acceleration:year + acceleration:origin,
```

```
##     data = dfSubsetWithNoNa)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5543 -1.6210 -0.1621  1.5175 16.5202
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          5.223e+01  3.074e+00  16.990  < 2e-16 ***
## I(displacement^2)    1.158e-04  1.869e-05   6.193 1.52e-09 ***
## acceleration        -5.745e+00  3.032e-01 -18.947  < 2e-16 ***
## origin              -8.648e+00  1.695e+00  -5.101 5.33e-07 ***
## displacement:year   -1.456e-03  1.198e-04 -12.159  < 2e-16 ***
## acceleration:year    6.422e-02  3.299e-03  19.466  < 2e-16 ***
## acceleration:origin  5.514e-01  1.042e-01   5.292 2.03e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.293 on 385 degrees of freedom
## Multiple R-squared:  0.8247, Adjusted R-squared:  0.8219
## F-statistic: 301.8 on 6 and 385 DF,  p-value: < 2.2e-16
```

```
#change acceleration -> log(acceleration) reduce Rsquared and
# increase RSE thus it does not help
df.lm5 <- lm (mpg ~ I(displacement^2) + log1p(acceleration) + origin +
                displacement:year + acceleration:year + acceleration:origin,
              data = dfSubsetWithNoNa)
summary(df.lm5)
```

```
##
## Call:
## lm(formula = mpg ~ I(displacement^2) + log1p(acceleration) +
##     origin + displacement:year + acceleration:year + acceleration:origin,
##     data = dfSubsetWithNoNa)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.0359  -1.7654   0.1449   1.7414  17.2743
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          1.747e+02  1.039e+01  16.816  < 2e-16 ***
## I(displacement^2)    4.619e-05  2.052e-05   2.252   0.0249 *
## log1p(acceleration) -7.136e+01  4.540e+00 -15.718  < 2e-16 ***
## origin              -3.016e+00  1.724e+00  -1.749   0.0811 .
## displacement:year   -1.070e-03  1.294e-04  -8.268 2.24e-15 ***
## year:acceleration    5.118e-02  3.188e-03  16.053  < 2e-16 ***
## origin:acceleration  2.218e-01  1.066e-01   2.081   0.0381 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.573 on 385 degrees of freedom
## Multiple R-squared:  0.7936, Adjusted R-squared:  0.7904
## F-statistic: 246.8 on 6 and 385 DF,  p-value: < 2.2e-16
```

```
# change origin -> log(origin) does not change the result
df.lm6 <- lm (mpg ~ I(displacement^2) + acceleration + log1p(origin) +
                displacement:year + acceleration:year + acceleration:origin,
              data = dfSubsetWithNoNa)
```

```
carseats.df = read.csv("/Users/shahrdadshadab/env/my-R-project/ISLR/Data/Carseats.csv",
                       header=T,  na.strings = "?", stringsAsFactors = F)

# summary(carseats.df)

# Now find a subset of records that have at least one NA
carseats.dfWithNa <- newDf[rowSums(is.na(carseats.df)) > 0,]
sprintf(" Number of observations containing NA is %d ",nrow(carseats.dfWithNa))
```

```
## [1] " Number of observations containing NA is 0 "
```

```
# Check a particular column that has NA and include the record
# dfSubsetWithNaInOneCol <- newDf[is.na(newDf$horsepower), ]
# head(dfSubsetWithNaInOneCol)

# now that there is no Na, we can safely convert US and Urban columns into factor

carseats.df[["ShelveLoc_factor"]] <-
  factor(carseats.df[["ShelveLoc"]], levels = c("Good", "Medium", "Bad"))
carseats.df[["Urban_factor"]] <- factor(carseats.df[["Urban"]],
                                        levels = c("Yes", "No"))
carseats.df[["US_factor"]] <- factor(carseats.df[["US"]],
                                     levels = c("Yes", "No"))

sprintf(" -------------- contrats carseats.df[['ShelveLoc_factor']] ----------")
```

```
## [1] " -------------- contrats carseats.df[['ShelveLoc_factor']] ----------"
```

```
contrasts(carseats.df[["ShelveLoc_factor"]])
```

```
##        Medium Bad
## Good        0   0
## Medium      1   0
## Bad         0   1
```

```
table(carseats.df[["ShelveLoc_factor"]])
```

```
##
##   Good Medium    Bad
##     85    219     96
```

```
sprintf(" -------------- contrats carseats.df[['Urban_factor']] --------------")
```

```
## [1] " -------------- contrats carseats.df[['Urban_factor']] --------------"
```

```
contrasts(carseats.df[["Urban_factor"]])
```

```
##     No
## Yes  0
## No   1
```

```
table(carseats.df[["Urban_factor"]])
```

```
##
```

```
## Yes  No
## 282 118
```

```
sprintf(" -------------- contrats carseats.df[[US_factor]] --------------------")
```

```
## [1] " -------------- contrats carseats.df[[US_factor]] --------------------"
```

```
contrasts(carseats.df[["US_factor"]])
```

```
##     No
## Yes  0
## No   1
```

```
table(carseats.df[["US_factor"]])
```

```
##
## Yes  No
## 258 142
```

```
carseats.lm <- lm(Sales ~ Price + Urban_factor + US_factor, data = carseats.df)
summary(carseats.lm)
```

```
##
## Call:
## lm(formula = Sales ~ Price + Urban_factor + US_factor, data = carseats.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.9206 -1.6220 -0.0564  1.5786  7.0581
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    14.222125   0.639123  22.253  < 2e-16 ***
## Price          -0.054459   0.005242 -10.389  < 2e-16 ***
## Urban_factorNo  0.021916   0.271650   0.081    0.936
## US_factorNo    -1.200573   0.259042  -4.635 4.86e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.472 on 396 degrees of freedom
## Multiple R-squared:  0.2393, Adjusted R-squared:  0.2335
## F-statistic: 41.52 on 3 and 396 DF,  p-value: < 2.2e-16
```

```
sprintf("base is urbon = No and also Us = No" )
```

```
## [1] "base is urbon = No and also Us = No"
```

```
sprintf(" In urbon area sales increases by 2%% ")
```

```
## [1] " In urbon area sales increases by 2% "
```

```
sprintf(" outside US sales decreases by 12%% or
        equivalently in US increases by 12%%")
```

```
## [1] " outside US sales decreases by 12% or \n        equivalently in US increases by 12%"
```

```
sprintf(" model that only uses the predictors for which
        there is evidence of association")
```

```
## [1] " model that only uses the predictors for which \n        there is evidence of association"
```

```
carseats.lm.smaller <- lm(Sales ~ Price + US_factor, data = carseats.df)
summary1 <- summary(carseats.lm.smaller)
sprintf("F-statisitcs improved , RSE slightly improved but
        Rsquared did not improve that much")
```

```
## [1] "F-statisitcs improved , RSE slightly improved but \n        Rsquared did not improve that much"
```

```
coeffsMatrix <- coef(summary1)
coeffsMatrix
```

```
##                Estimate  Std. Error    t value     Pr(>|t|)
## (Intercept) 14.23043570 0.629978186  22.588775 3.000871e-73
## Price       -0.05447763 0.005230126 -10.416123 1.272157e-22
## US_factorNo -1.19964294 0.258461026  -4.641485 4.707187e-06
```

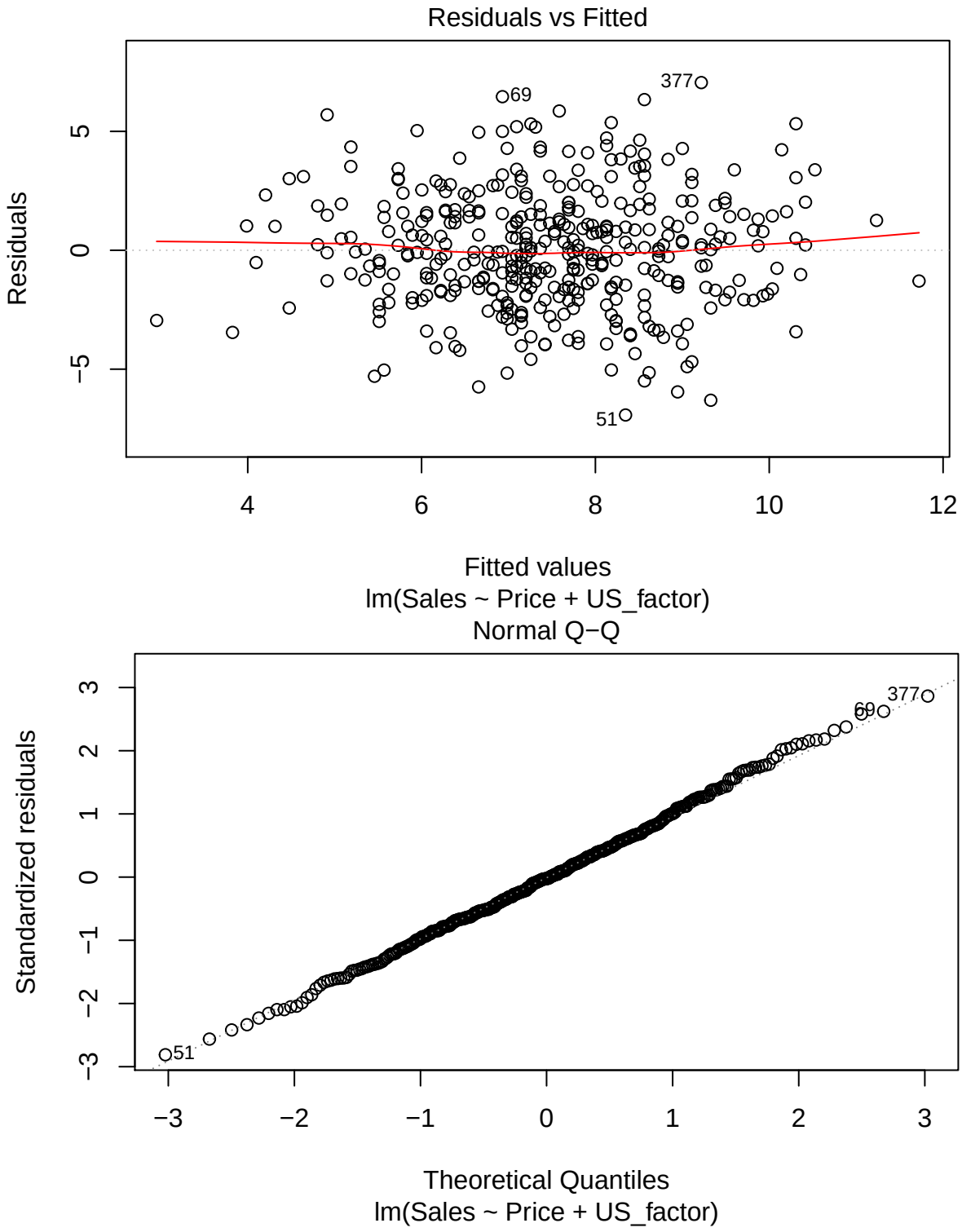
```
originalConfInt <- confint(carseats.lm.smaller)
myConfInt <- cbind(coeffsMatrix[,1]-coeffsMatrix[,2]*2,
                   coeffsMatrix[,1]+coeffsMatrix[,2]*2)
originalConfInt
```

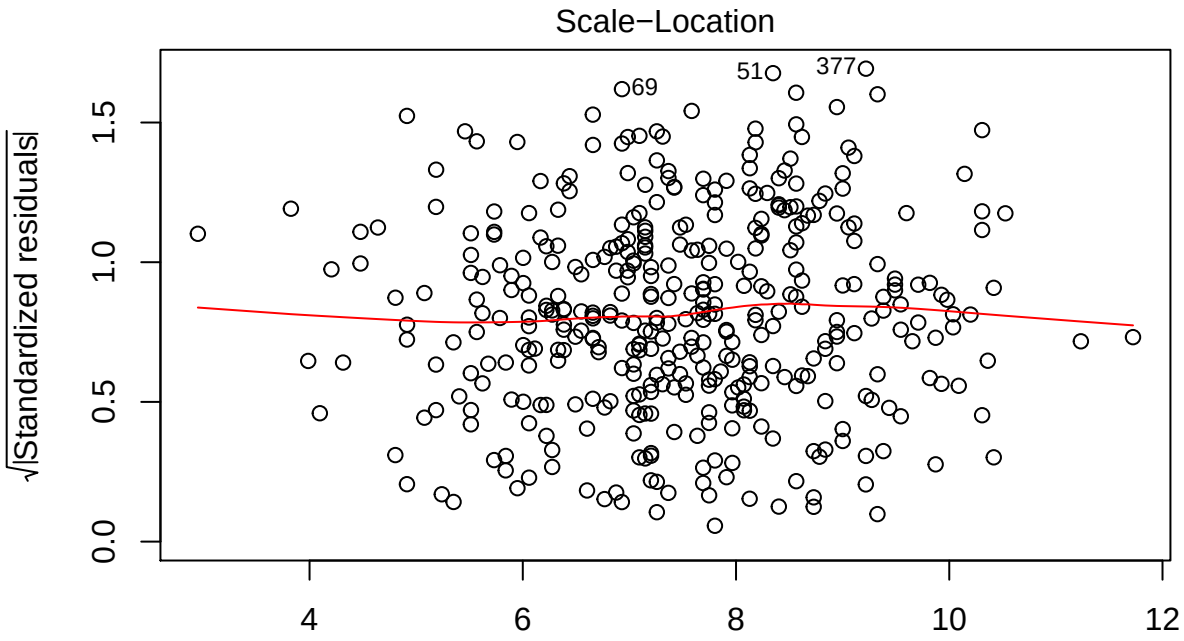
```
##                   2.5 %      97.5 %
## (Intercept) 12.99192540 15.46894599
## Price       -0.06475984 -0.04419543
## US_factorNo -1.70776632 -0.69151957
```

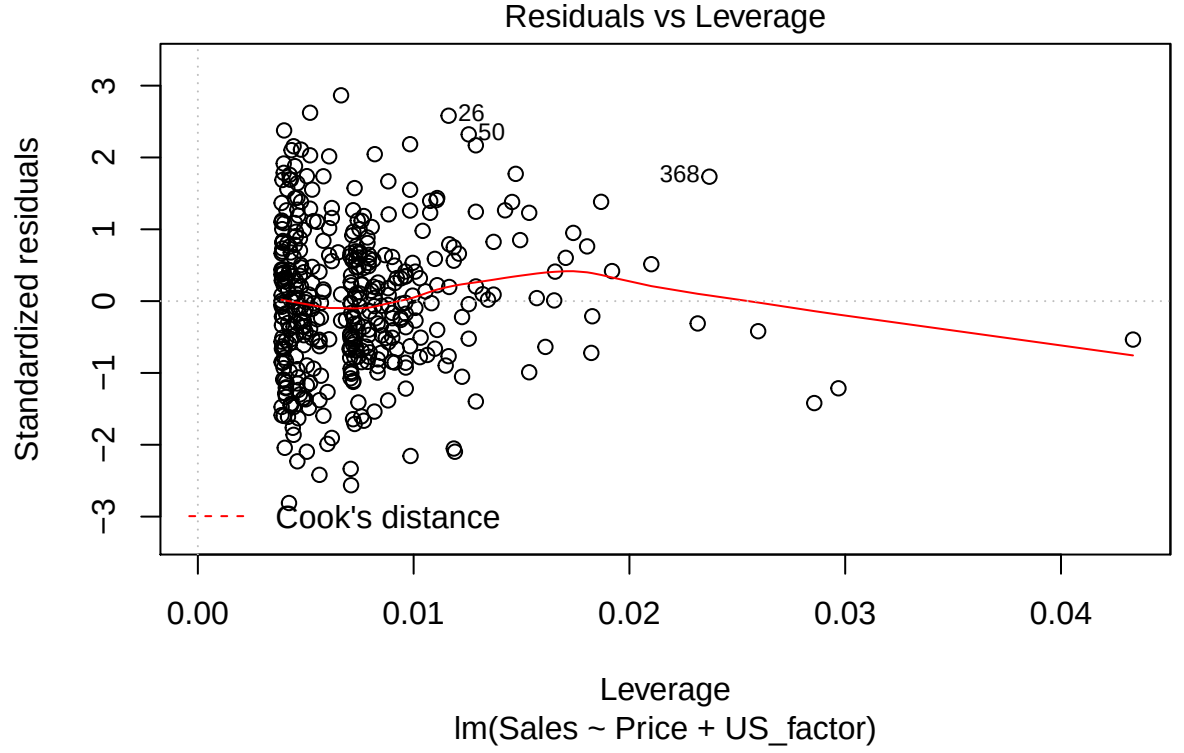
```
myConfInt
```

```
##                    [,1]        [,2]
## (Intercept) 12.97047933 15.49039207
## Price       -0.06493788 -0.04401738
## US_factorNo -1.71656500 -0.68272089
```

```
plot(carseats.lm.smaller)
```


Residuals vs Fitted
Residuals
Fitted values
lm(Sales ~ Price + US_factor)
69
377
51
Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(Sales ~ Price + US_factor)






```
sprintf("Outliers are observations 69, 51, 377")
```

```
## [1] "Outliers are observations 69, 51, 377"
```

```
sprintf("High Leverage points are 26, 50 and 368")
```

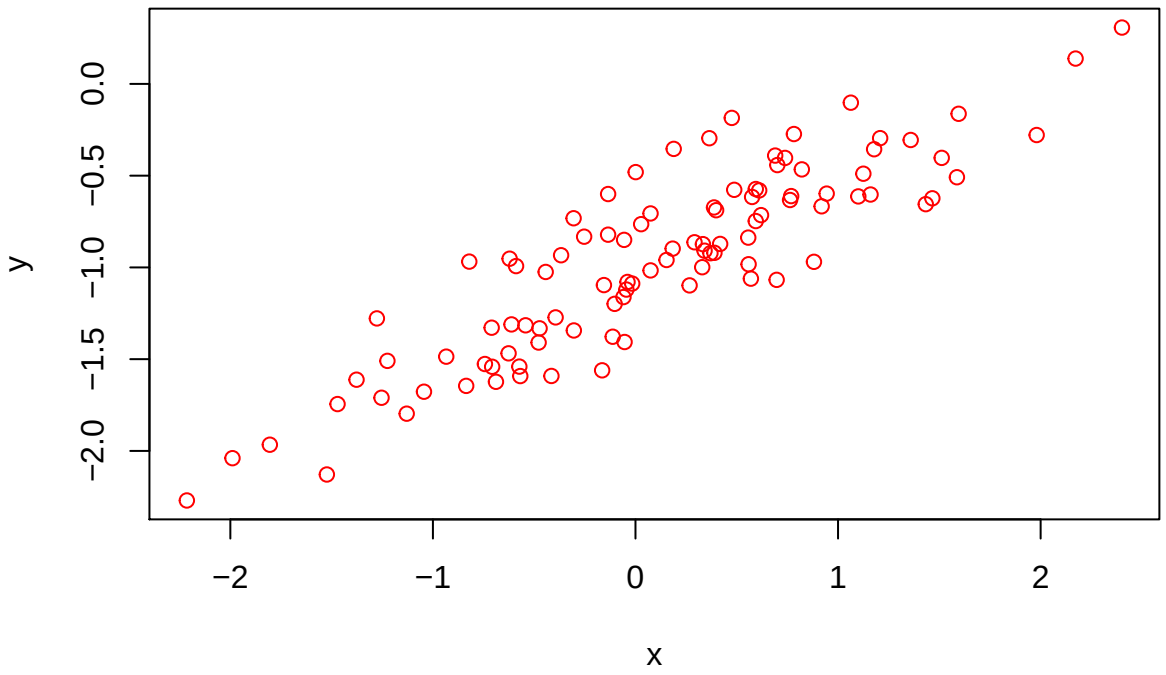
```
## [1] "High Leverage points are 26, 50 and 368"
```

```r
sprintf("There is no high leverage point which is also outlier")
```

```
## [1] "There is no high leverage point which is also outlier"
```

```r
# "outliers"
# plot(predict(carseats.lm.smaller),rstudent(carseats.lm.smaller),
#      col=ifelse(rstudent(carseats.lm.smaller)>3,"red","black"))
#
# "High leverage"
# n <- nrow(carseats.df)
# p <- ncol(carseats.df)
# plot(hatvalues(carseats.lm.smaller),
#      col=ifelse(hatvalues(carseats.lm.smaller) > (p+1)/n, "red", "blue"))+
#   text(hatvalues(carseats.lm.smaller),
#        labels=ifelse(hatvalues(carseats.lm.smaller) > (p+1)/n,
#                      names(which(hatvalues(carseats.lm.smaller) > (p+1)/n)), ""))
```

```r
# if (!require("pacman")) install.packages("pacman")
# p_load(datasets, ggplot2, ggthemes, dplyr, RColorBrewer, grid)
# data(airquality)

set.seed(1)
x <- rnorm(100)
eps <- rnorm(100, 0, 0.25)
y <- -1 + 0.5*x + eps

plot(x, y, col="red")
```




```r
sprintf("fit a least square model to model with noise variance 0.25")
```

```
## [1] "fit a least square model to model with noise variance 0.25"
```

```r
# first create a datafrme from x and y
data.df <- data.frame(x1 = x, y1 = y)
```

```r
data.lm <- lm(y1 ~ x1, data = data.df)
summary.lm <- summary(data.lm)
summary.lm
```

```
##
## Call:
## lm(formula = y1 ~ x1, data = data.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.46921 -0.15344 -0.03487  0.13485  0.58654
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.00942    0.02425  -41.63   <2e-16 ***
## x1           0.49973    0.02693   18.56   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2407 on 98 degrees of freedom
## Multiple R-squared:  0.7784, Adjusted R-squared:  0.7762
## F-statistic: 344.3 on 1 and 98 DF,  p-value: < 2.2e-16
```

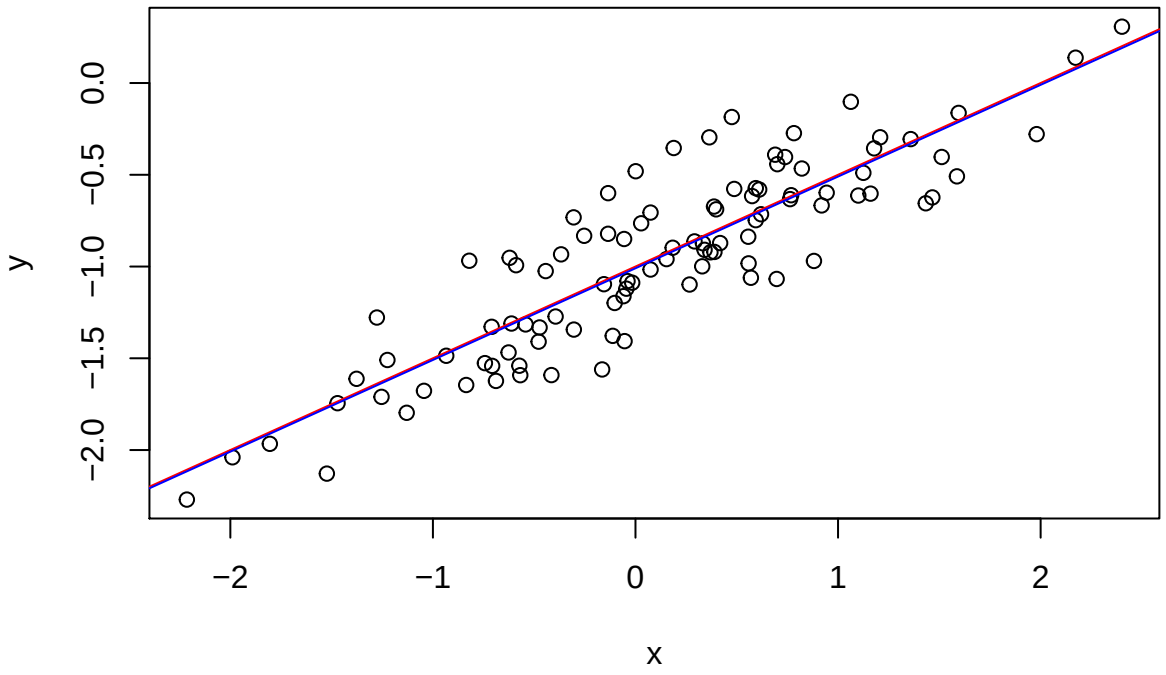
```r
coef(summary.lm)
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) -1.0094232 0.02424682 -41.63115 4.106694e-64
## x1           0.4997349 0.02693176  18.55560 7.723851e-34
```

```r
confint(data.lm)
```

```
##                  2.5 %     97.5 %
## (Intercept) -1.0575402 -0.9613061
## x1           0.4462897  0.5531801
```

```r
# draw least square line and population regression line
plot(x, y, col="black") +
  abline(coef = c(-1, 0.5), col="red")+
  abline(coef = coef(data.lm), col="blue")
```




```
## integer(0)
```

```
# redo the above with less noise in the data
sprintf("fit a least square model to model with noise variance 0.05")
```

```
## [1] "fit a least square model to model with noise variance 0.05"
```

```
eps1 <- rnorm(100, 0, 0.05)
yy <- -1 + 0.5 * x + eps1

plot(x, yy, col="black")
# fit a least square model
# first create a datafrme from x and y
data.df2 <- data.frame(x2 = x, y2 = yy)
data.lm2 <- lm(y2 ~ x2, data = data.df2)
summary.lm2 <- summary(data.lm2)
summary.lm2
```

```
##
## Call:
## lm(formula = y2 ~ x2, data = data.df2)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.145706 -0.024115 -0.002266  0.032462  0.132079
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.998632   0.005235 -190.75   <2e-16 ***
## x2           0.501058   0.005815   86.17   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05197 on 98 degrees of freedom
```

```
## Multiple R-squared:  0.987,  Adjusted R-squared:  0.9868
## F-statistic:  7425 on 1 and 98 DF,  p-value: < 2.2e-16
```

```
coef(summary.lm2)
```

```
##               Estimate  Std. Error    t value      Pr(>|t|)
## (Intercept) -0.9986316 0.005235197 -190.75339 8.557575e-128
## x2           0.5010583 0.005814908   86.16788  3.429228e-94
```

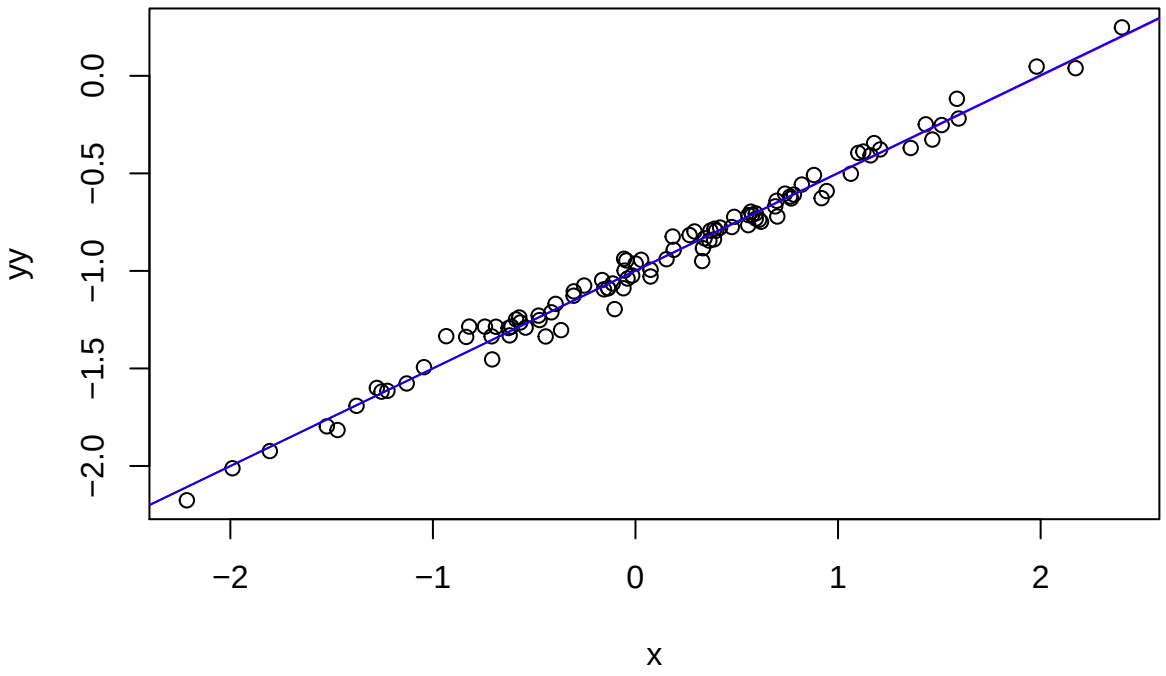
```
confint(data.lm2)
```

```
##                  2.5 %     97.5 %
## (Intercept) -1.0090206 -0.9882425
## x2           0.4895188  0.5125978
```

```
sprintf("RSE significantly reduced and F2 increased when we
        decreased error also coefficients gets much closer to real values")
```

```
## [1] "RSE significantly reduced and F2 increased when we \n        decreased error also coefficients g
```

```
plot(x, yy, col="black") +
  abline(coef = c(-1, 0.5), col="red")+
  abline(coef = coef(data.lm2), col="blue")
```




```
## integer(0)
```

```
# redo the above with more noise in the data
sprintf("fit a least square model to model with noise variance 0.8")
```

```
## [1] "fit a least square model to model with noise variance 0.8"
```

```
eps2 <- rnorm(100, 0, 0.8)
yy2 <- -1 + 0.5 * x + eps2

plot(x, yy2, col="black")
# fit a least square model
# first create a datafrme from x and y
```

```r
data.df3 <- data.frame(x3 = x, y3 = yy2)
data.lm3 <- lm(yy2 ~ x3, data = data.df3)
summary.lm3 <- summary(data.lm3)
summary.lm3
```

```
##
## Call:
## lm(formula = yy2 ~ x3, data = data.df3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.01301 -0.43620 -0.03021  0.53831  1.50310
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.95387    0.08023 -11.890  < 2e-16 ***
## x3           0.45545    0.08911   5.111 1.58e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7964 on 98 degrees of freedom
## Multiple R-squared:  0.2105, Adjusted R-squared:  0.2024
## F-statistic: 26.12 on 1 and 98 DF,  p-value: 1.584e-06
```

```r
coef(summary.lm3)
```

```
##               Estimate Std. Error    t value     Pr(>|t|)
## (Intercept) -0.9538677 0.08022518 -11.889880 1.028830e-20
## x3           0.4554512 0.08910879   5.111181 1.583903e-06
```

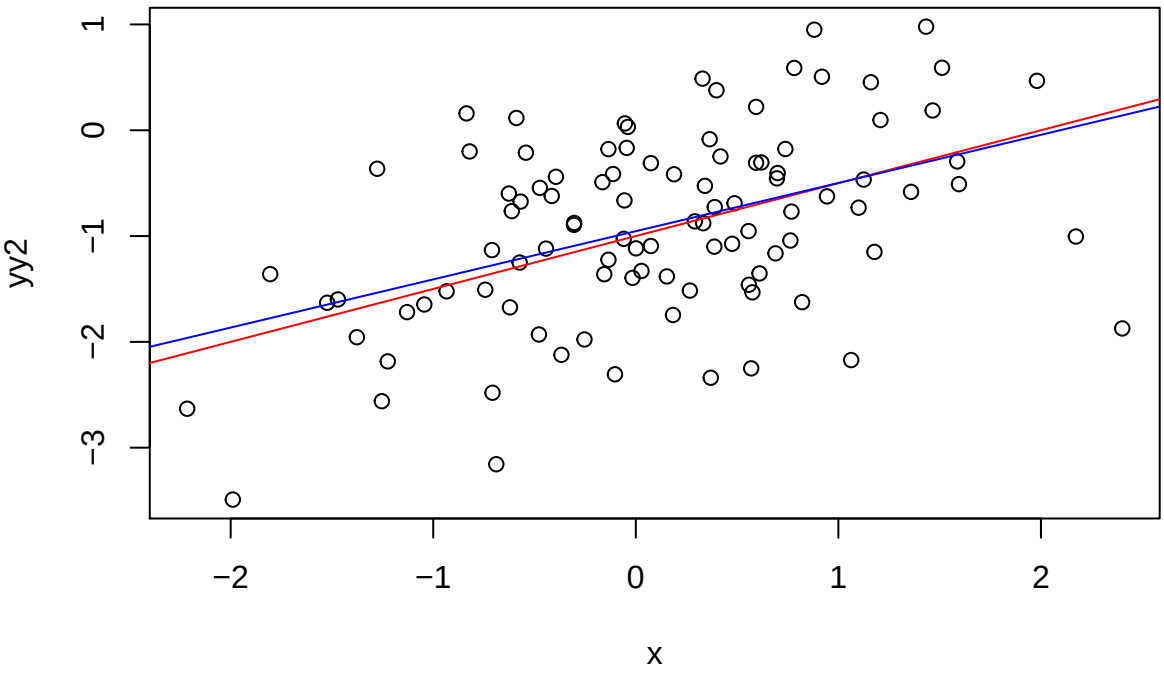
```r
confint(data.lm3)
```

```
##                  2.5 %     97.5 %
## (Intercept) -1.1130720 -0.7946635
## x3           0.2786177  0.6322846
```

```r
sprintf("RSE significantly increased and F2 decreased when we
        increased error also coefficients gets much closer to real values")
```

```
## [1] "RSE significantly increased and F2 decreased when we \n        increased error also coefficients
```

```r
plot(x, yy2, col="black") +
  abline(coef = c(-1, 0.5), col="red")+
  abline(coef = coef(data.lm3), col="blue")
```




```
## integer(0)
```

```
sprintf("Confedence interval for noisier data: ")
```

```
## [1] "Confedence interval for noisier data: "
```

```
confint(data.lm3)
```

```
##                   2.5 %     97.5 %
## (Intercept) -1.1130720 -0.7946635
## x3           0.2786177  0.6322846
```

```
sprintf("Confedence interval for less noisy data: ")
```

```
## [1] "Confedence interval for less noisy data: "
```

```
confint(data.lm2)
```

```
##                   2.5 %     97.5 %
## (Intercept) -1.0090206 -0.9882425
## x2           0.4895188  0.5125978
```

```
sprintf("For less noisy data has smaller confedence interval than noisier data")
```

```
## [1] "For less noisy data has smaller confedence interval than noisier data"
```

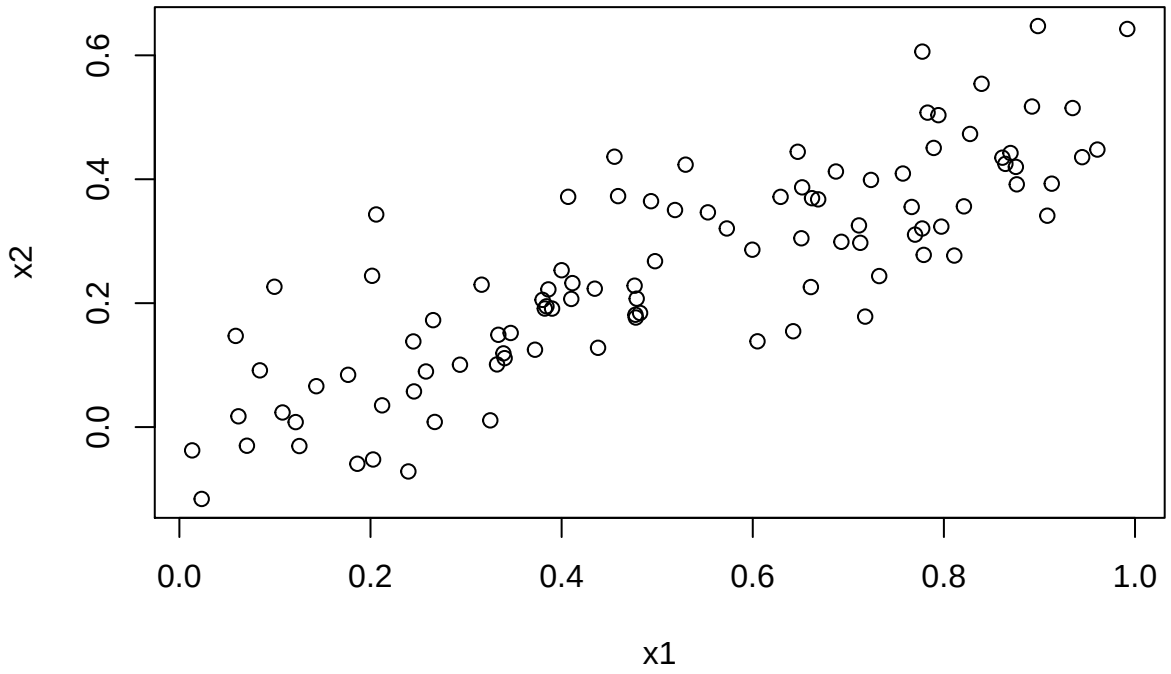
```
set.seed(1)
x1 <- runif(100)
x2 <- 0.5*x1 + rnorm(100)/10
y <- 2+2*x1+0.3*x2+rnorm(100)

sprintf("corrolation between x1 and x2:")
```

```
## [1] "corrolation between x1 and x2:"
```

```
cor(x1,x2)
```

```
## [1] 0.8351212
```

```r
plot(x1,x2)
```




```r
sprintf("Fit a least square model y ~ x1 + x2: ")
```

```
## [1] "Fit a least square model y ~ x1 + x2: "
```

```r
lm1 <- lm(yc ~ xc1 + xc2, data = data.frame(xc1 = x1, xc2 = x2, yc = y))
summary(lm1)
```

```
##
## Call:
## lm(formula = yc ~ xc1 + xc2, data = data.frame(xc1 = x1, xc2 = x2,
##     yc = y))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.8311 -0.7273 -0.0537  0.6338  2.3359
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.1305     0.2319   9.188 7.61e-15 ***
## xc1           1.4396     0.7212   1.996   0.0487 *
## xc2           1.0097     1.1337   0.891   0.3754
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.056 on 97 degrees of freedom
## Multiple R-squared:  0.2088, Adjusted R-squared:  0.1925
## F-statistic:  12.8 on 2 and 97 DF,  p-value: 1.164e-05
```

```r
sprintf("Fit a least square model y ~ x1: ")
```

```
## [1] "Fit a least square model y ~ x1: "
```

```r
lm2 <- lm(yc ~ xc1, data = data.frame(xc1 = x1, yc = y))
summary(lm2)
```

```
##
## Call:
## lm(formula = yc ~ xc1, data = data.frame(xc1 = x1, yc = y))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.89495 -0.66874 -0.07785  0.59221  2.45560
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.1124     0.2307   9.155 8.27e-15 ***
## xc1           1.9759     0.3963   4.986 2.66e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.055 on 98 degrees of freedom
## Multiple R-squared:  0.2024, Adjusted R-squared:  0.1942
## F-statistic: 24.86 on 1 and 98 DF,  p-value: 2.661e-06
```

```r
sprintf("Fit a least square model y ~ x2: ")
```

```
## [1] "Fit a least square model y ~ x2: "
```

```r
lm3 <- lm(yc ~ xc2, data = data.frame(xc2 = x2, yc = y))
summary(lm3)
```

```
##
## Call:
## lm(formula = yc ~ xc2, data = data.frame(xc2 = x2, yc = y))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.62687 -0.75156 -0.03598  0.72383  2.44890
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.3899     0.1949   12.26  < 2e-16 ***
## xc2           2.8996     0.6330    4.58 1.37e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.072 on 98 degrees of freedom
## Multiple R-squared:  0.1763, Adjusted R-squared:  0.1679
## F-statistic: 20.98 on 1 and 98 DF,  p-value: 1.366e-05
```

```r
sprintf("No contradiction: there is a correlation between x1 and x2,
        if one is there the other do not add much information
        (each one alone adds info)")
```

```
## [1] "No contradiction: there is a correlation between x1 and x2, \n        if one is there the other
```

```r
# add new information to existsing data
x1 <- c(x1, 0.1)
```

```r
x2 <- c(x2, 0.8)
y1 <- c(y,6)

sprintf("Fit a least square model y ~ x1 + x2 with new data: ")
```

```
## [1] "Fit a least square model y ~ x1 + x2 with new data: "
```

```r
lm1 <- lm(yc ~ xc1 + xc2, data = data.frame(xc1 = x1, xc2 = x2, yc = y1))
summary(lm1)
```

```
##
## Call:
## lm(formula = yc ~ xc1 + xc2, data = data.frame(xc1 = x1, xc2 = x2,
##     yc = y1))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.73348 -0.69318 -0.05263  0.66385  2.30619
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.2267     0.2314   9.624 7.91e-16 ***
## xc1           0.5394     0.5922   0.911  0.36458
## xc2           2.5146     0.8977   2.801  0.00614 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.075 on 98 degrees of freedom
## Multiple R-squared:  0.2188, Adjusted R-squared:  0.2029
## F-statistic: 13.72 on 2 and 98 DF,  p-value: 5.564e-06
```

```r
sprintf("Fit a least square model y ~ x1  with new data: ")
```

```
## [1] "Fit a least square model y ~ x1  with new data: "
```

```r
lm2 <- lm(yc ~ xc1, data = data.frame(xc1 = x1, yc = y1))
summary(lm2)
```

```
##
## Call:
## lm(formula = yc ~ xc1, data = data.frame(xc1 = x1, yc = y1))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.8897 -0.6556 -0.0909  0.5682  3.5665
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.2569     0.2390   9.445 1.78e-15 ***
## xc1           1.7657     0.4124   4.282 4.29e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.111 on 99 degrees of freedom
## Multiple R-squared:  0.1562, Adjusted R-squared:  0.1477
## F-statistic: 18.33 on 1 and 99 DF,  p-value: 4.295e-05
```

```r
sprintf("Fit a least square model y ~ x2  with new data: ")
```

```
## [1] "Fit a least square model y ~ x2  with new data: "
```

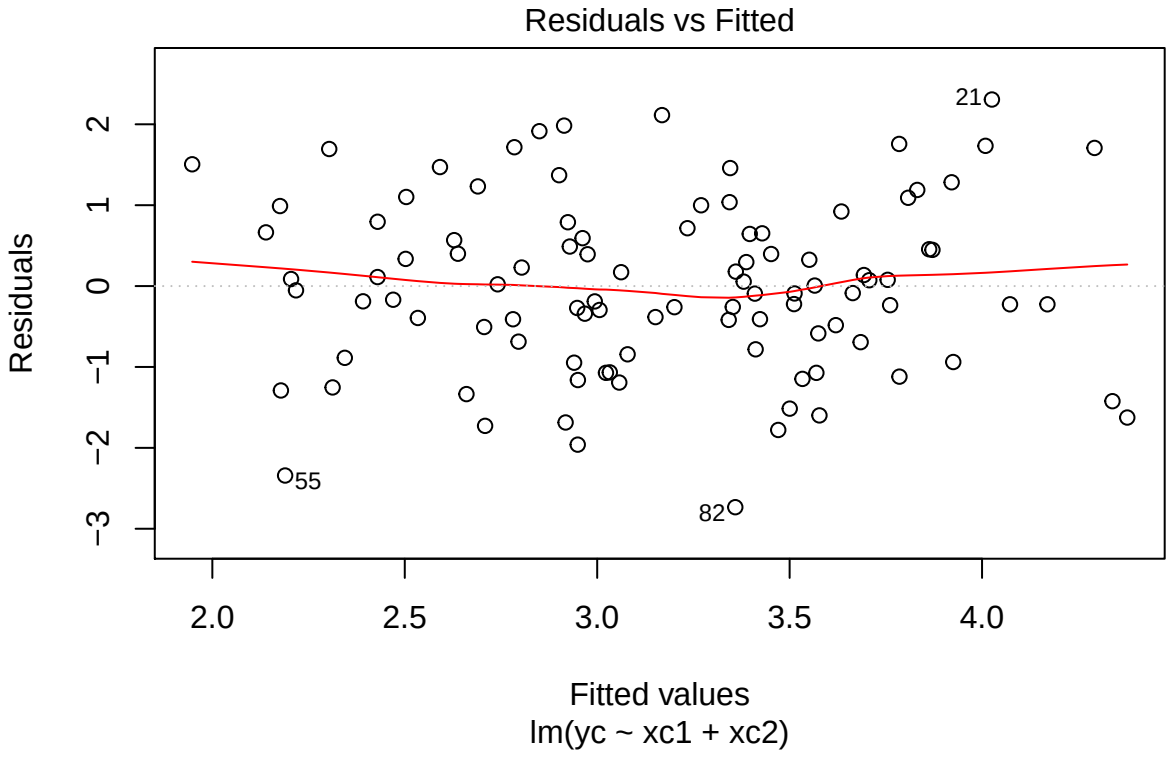
```r
lm3 <- lm(yc ~ xc2, data = data.frame(xc2 = x2, yc = y1))
summary(lm3)
```

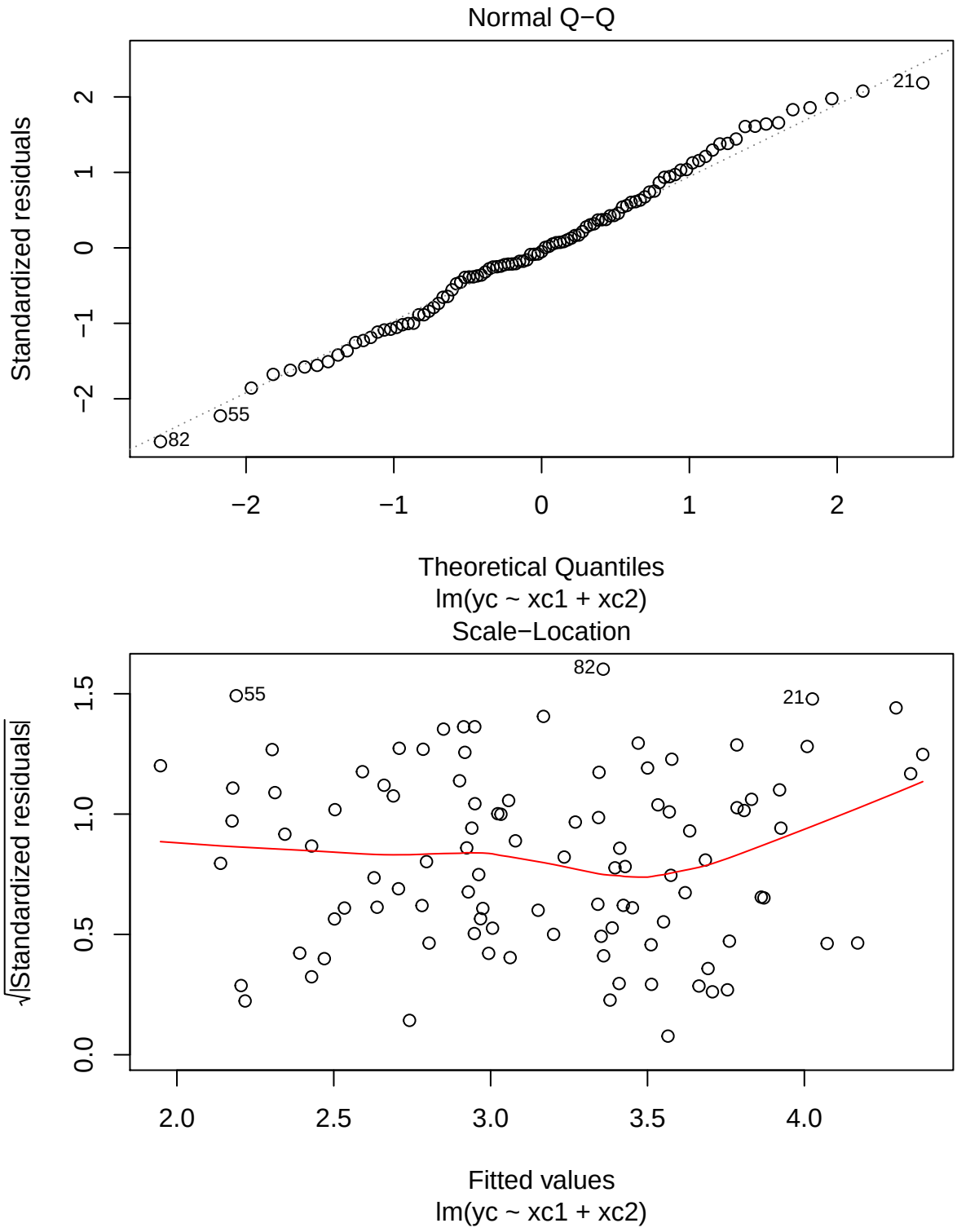
```
##
## Call:
## lm(formula = yc ~ xc2, data = data.frame(xc2 = x2, yc = y1))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.64729 -0.71021 -0.06899  0.72699  2.38074
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.3451     0.1912  12.264  < 2e-16 ***
## xc2           3.1190     0.6040   5.164 1.25e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.074 on 99 degrees of freedom
## Multiple R-squared:  0.2122, Adjusted R-squared:  0.2042
## F-statistic: 26.66 on 1 and 99 DF,  p-value: 1.253e-06
```

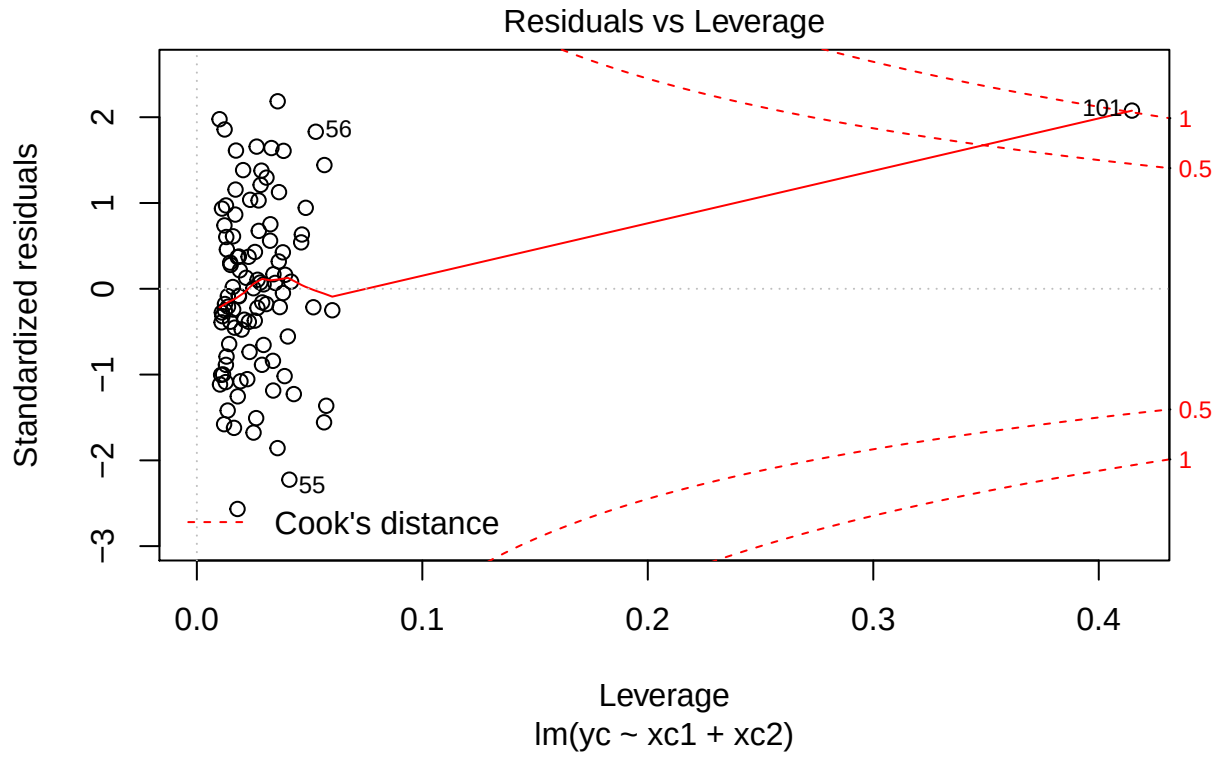
```r
"new data (observation 101) is only high leverage in y1 ~ x1 + x2 model"
```

```
## [1] "new data (observation 101) is only high leverage in y1 ~ x1 + x2 model"
```

```r
plot(lm1)
```



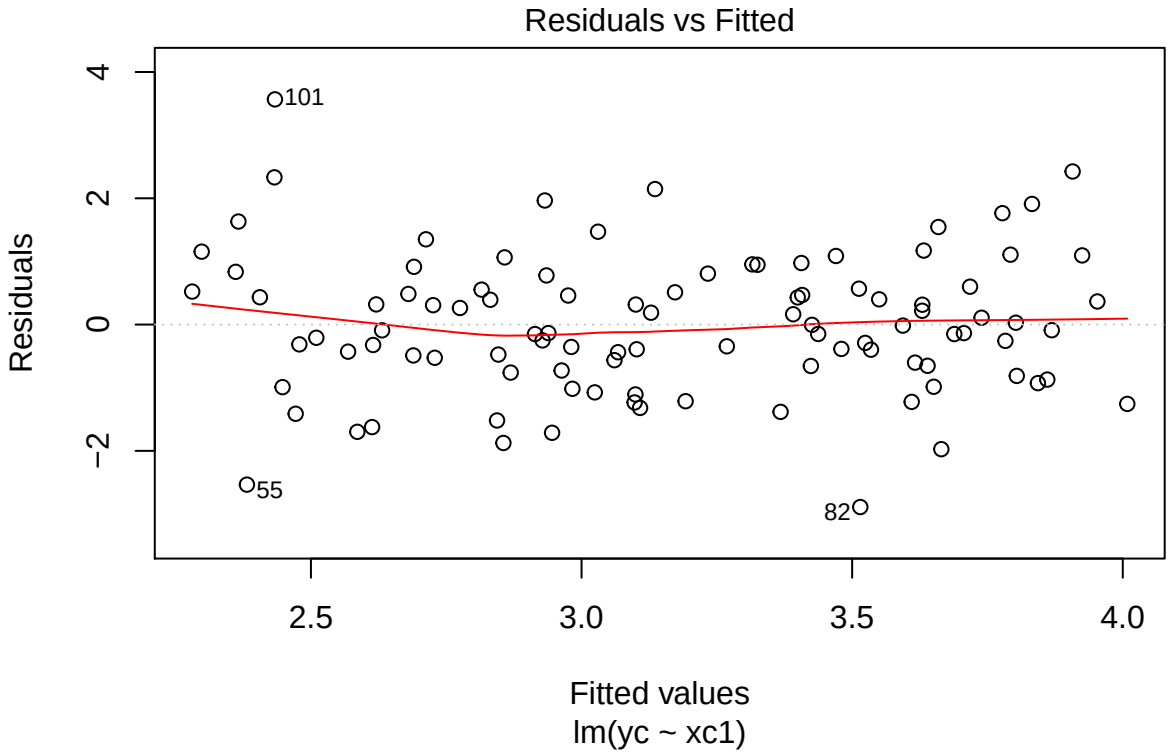

Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(yc ~ xc1 + xc2)
Scale−Location
√|Standardized residuals|
Fitted values
lm(yc ~ xc1 + xc2)
82
55
21

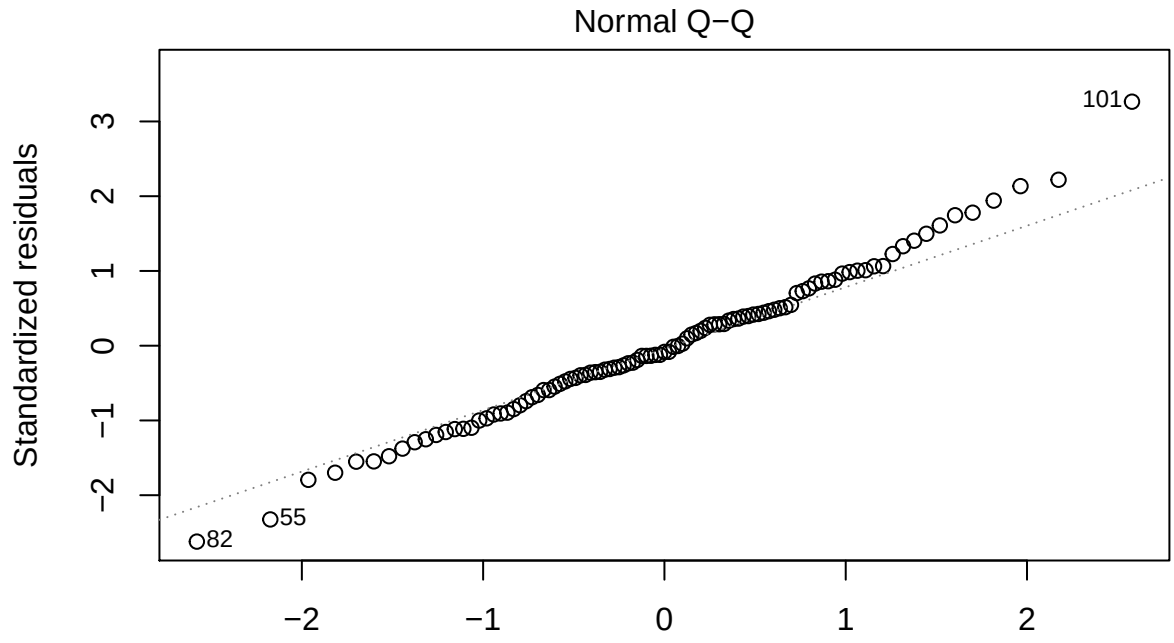


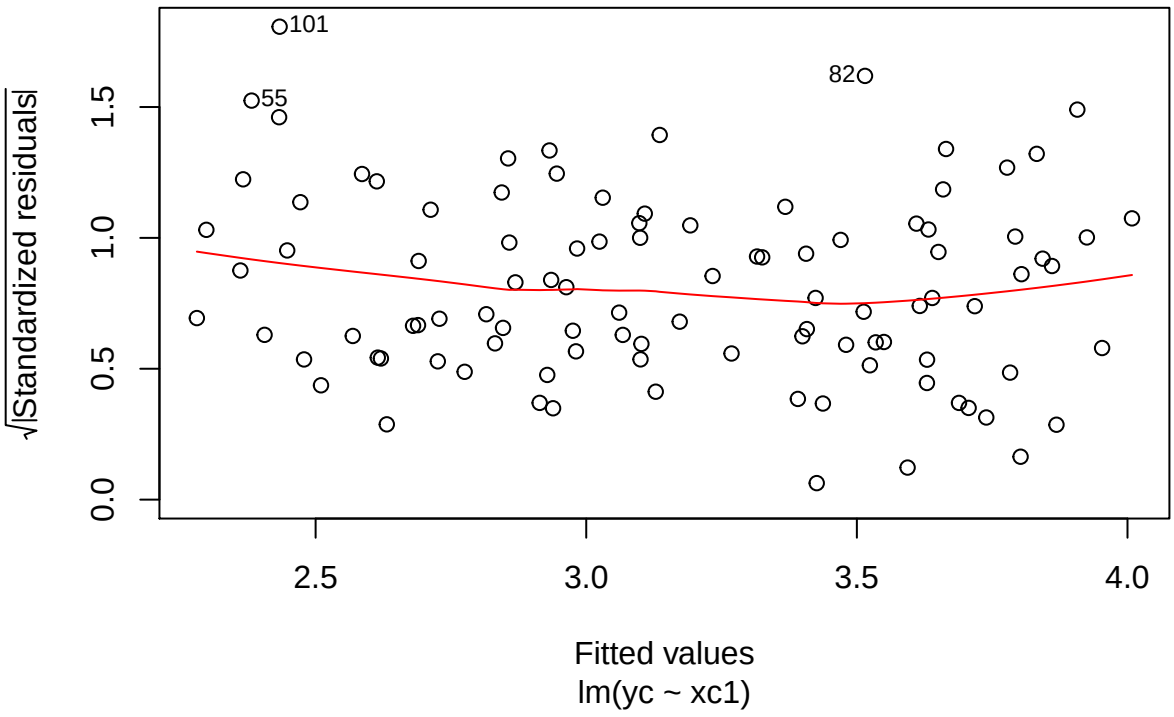

```
"new data (observation 101) is outlier and high leverage in y1 ~ x1 model"
```

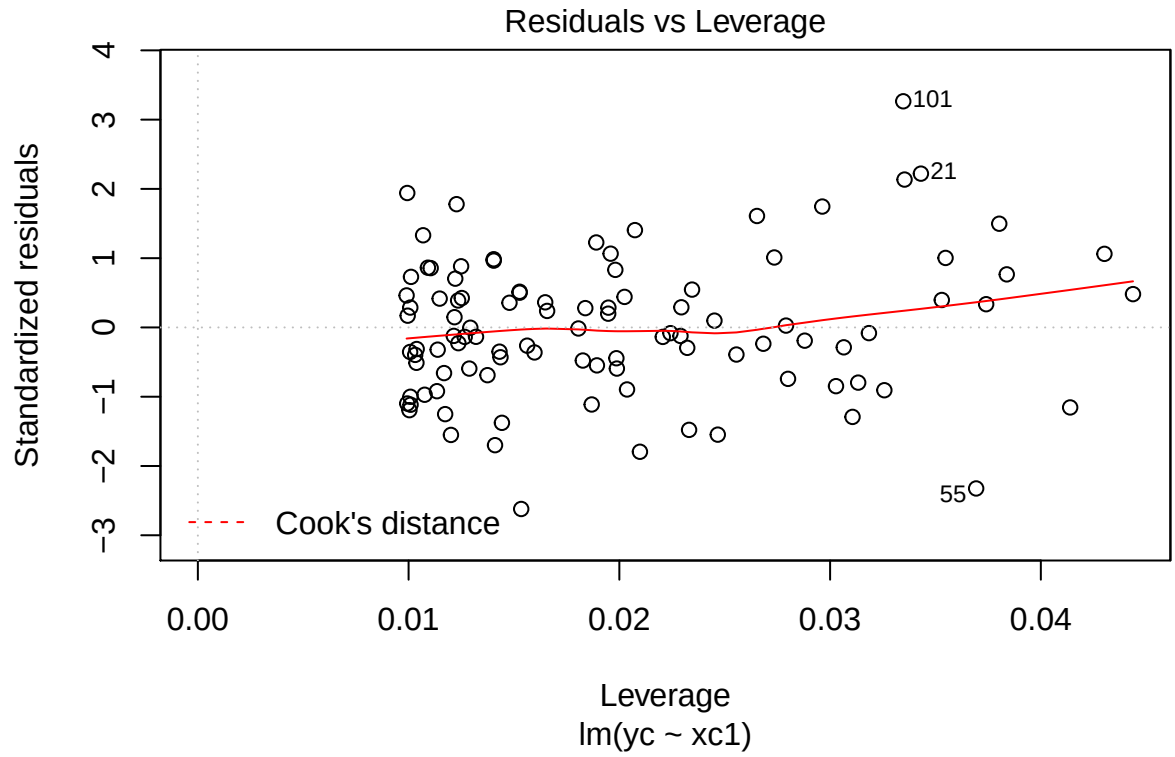
```
## [1] "new data (observation 101) is outlier and high leverage in y1 ~ x1 model"
```

```
plot(lm2)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(yc ~ xc1)
101
55
82
−2
−1
0
1
2
3

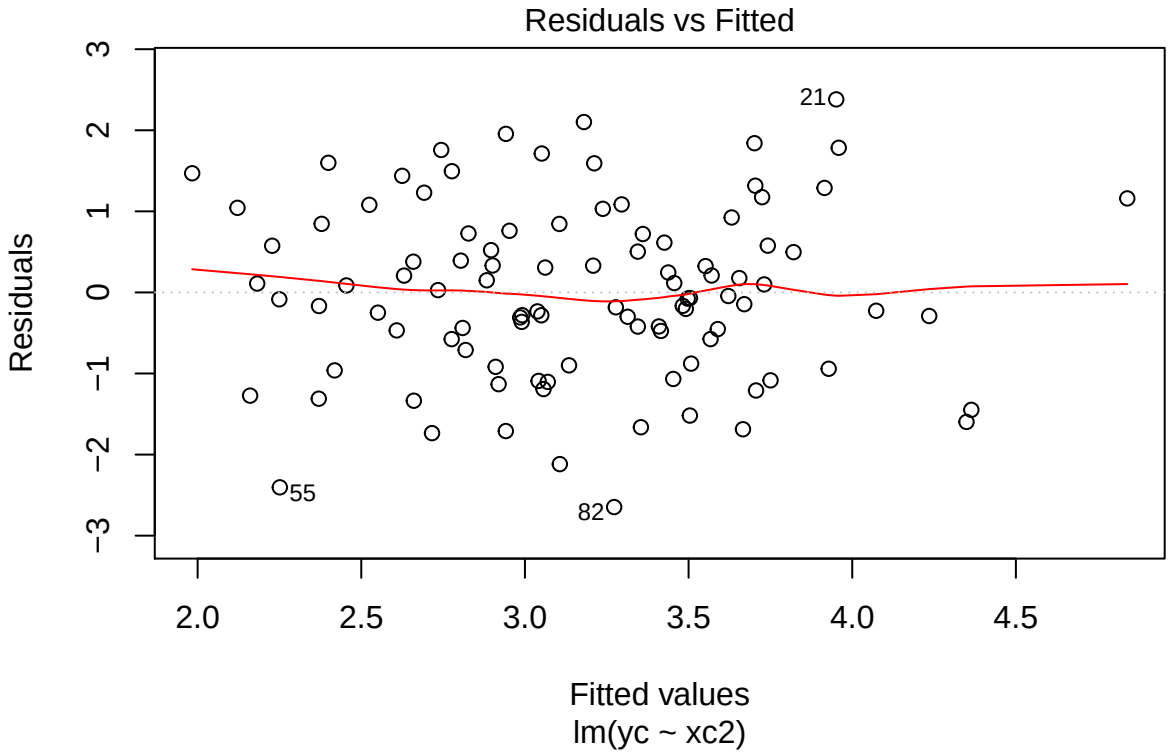

Scale−Location
√|Standardized residuals|
Fitted values
lm(yc ~ xc1)
101
55
82
0.0
0.5
1.0
1.5
2.5
3.0
3.5
4.0

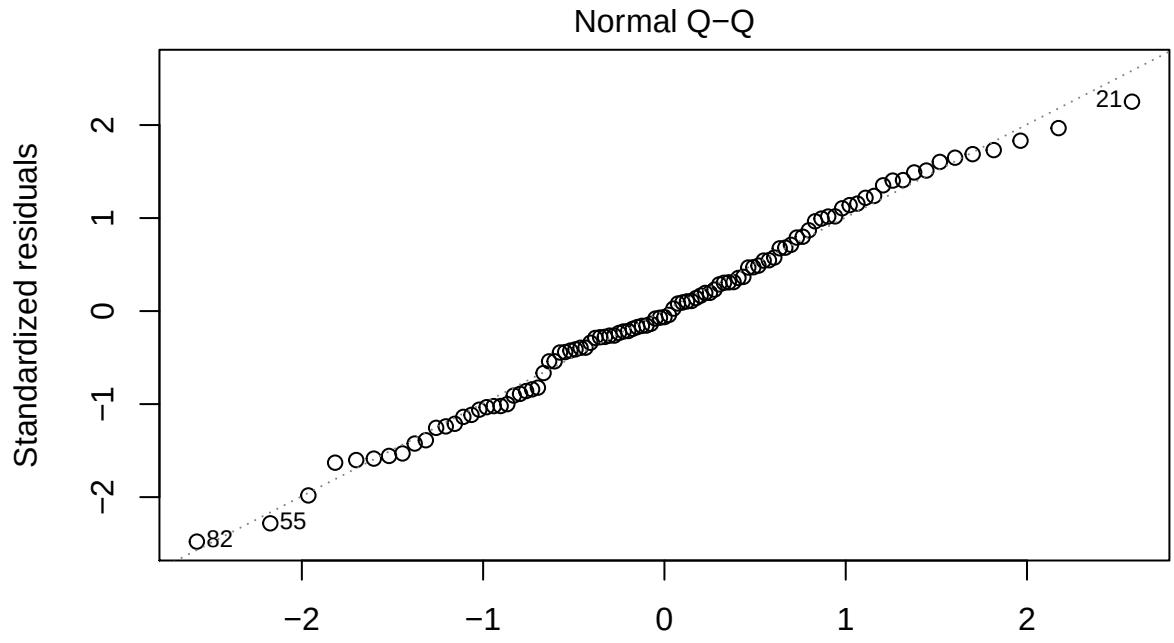


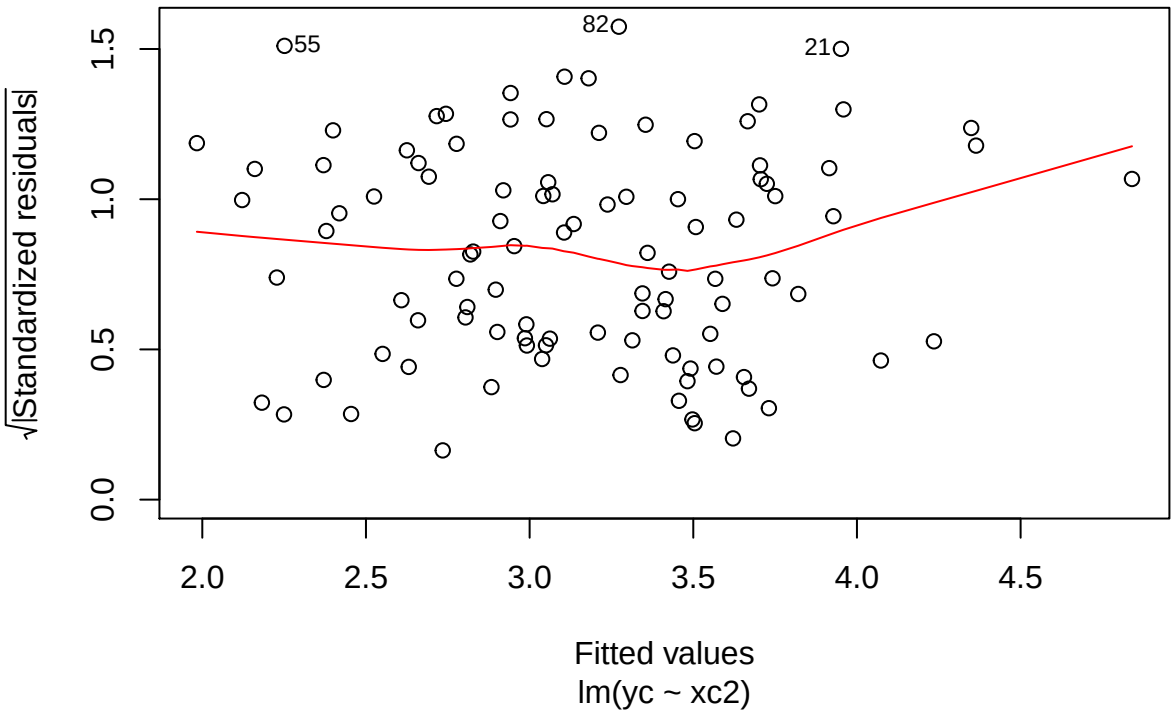

```
"new data (observation 101) is only high leverage in y1 ~ x2 model"
```

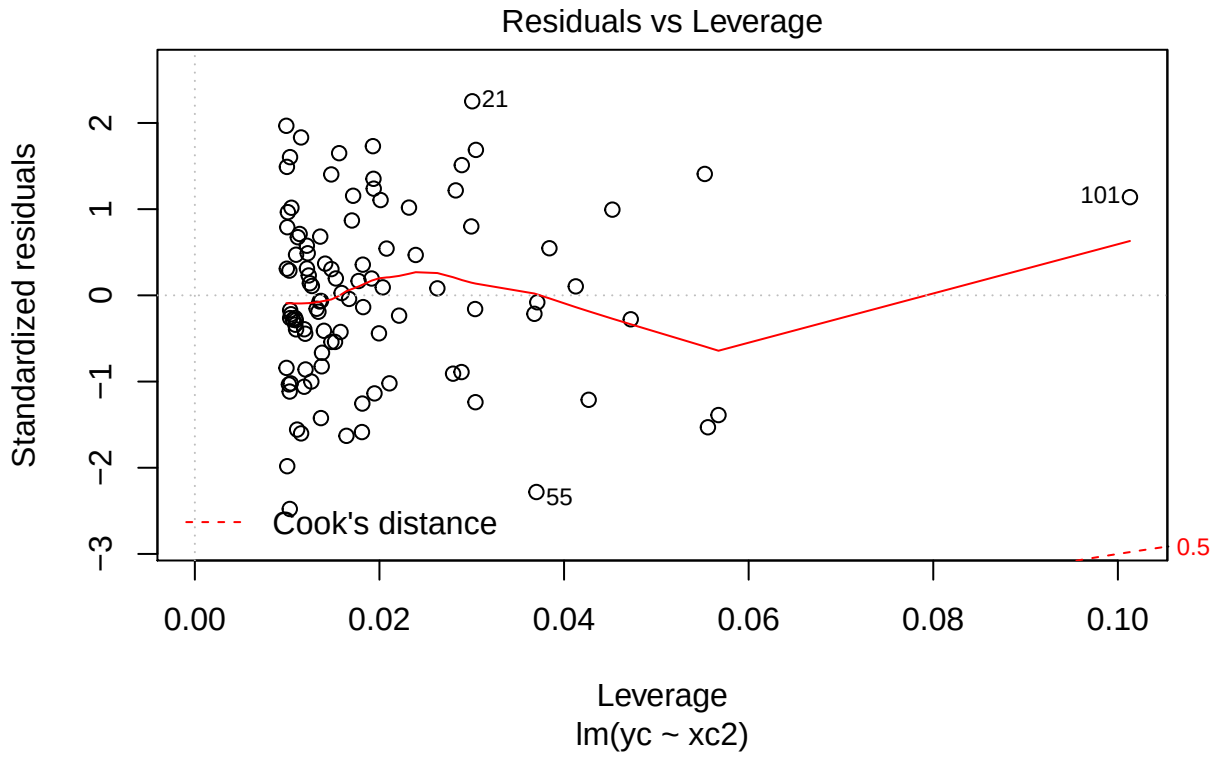
```
## [1] "new data (observation 101) is only high leverage in y1 ~ x2 model"
```

```
plot(lm3)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(yc ~ xc2)
82
55
21


Scale−Location
√|Standardized residuals|
Fitted values
lm(yc ~ xc2)
82
55
21




```
sprintf(" as before there is a correlation between x1 and x2,
        if one is there the other do not add much information
        (each one alone adds info)")
```

```
## [1] " as before there is a correlation between x1 and x2, \n        if one is there the other do not
```

```
boston.df = read.csv("/Users/shahrdadshadab/env/my-R-project/ISLR/Data/Boston.csv",
                     header=T,  na.strings = "?", stringsAsFactors = T)
# summary(boston.df)
str(boston.df)
```

```
## 'data.frame':    506 obs. of  14 variables:
##  $ crim   : num  0.00632 0.02731 0.02729 0.03237 0.06905 ...
##  $ zn     : num  18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...
##  $ indus  : num  2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...
##  $ chas   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ nox    : num  0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524 ...
##  $ rm     : num  6.58 6.42 7.18 7 7.15 ...
##  $ age    : num  65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...
##  $ dis    : num  4.09 4.97 4.97 6.06 6.06 ...
##  $ rad    : int  1 2 2 3 3 3 5 5 5 5 ...
##  $ tax    : int  296 242 242 222 222 222 311 311 311 311 ...
##  $ ptratio: num  15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...
##  $ black  : num  397 397 393 395 397 ...
##  $ lstat  : num  4.98 9.14 4.03 2.94 5.33 ...
##  $ medv   : num  24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...
```

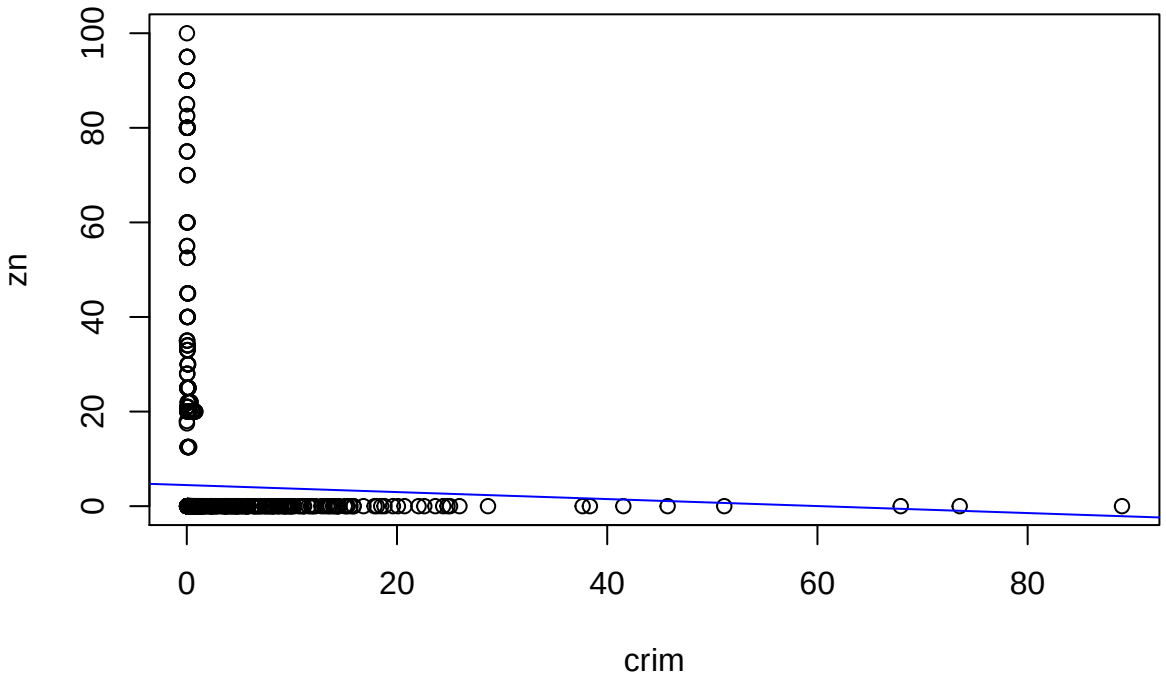
```
# count number of observations with NA
boston.df.WithNa <- boston.df[rowSums(is.na(boston.df)) > 0, ]
# str(attributes(is.na(boston.df)))
sprintf(" Number of observations containing NA is %d ",nrow(boston.df.WithNa))
```

```
## [1] " Number of observations containing NA is 0 "
```

```
# for each predictor fit a model
attach(boston.df)
lm1 <- lm(crim ~ zn, data = boston.df)
summary(lm1)
```

```
##
## Call:
## lm(formula = crim ~ zn, data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -4.429 -4.222 -2.620  1.250 84.523
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.45369    0.41722  10.675  < 2e-16 ***
## zn          -0.07393    0.01609  -4.594 5.51e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.435 on 504 degrees of freedom
## Multiple R-squared:  0.04019,    Adjusted R-squared:  0.03828
## F-statistic:  21.1 on 1 and 504 DF,  p-value: 5.506e-06
```

```
plot(crim, zn)+abline(coef = coef(lm1), col="blue")
```



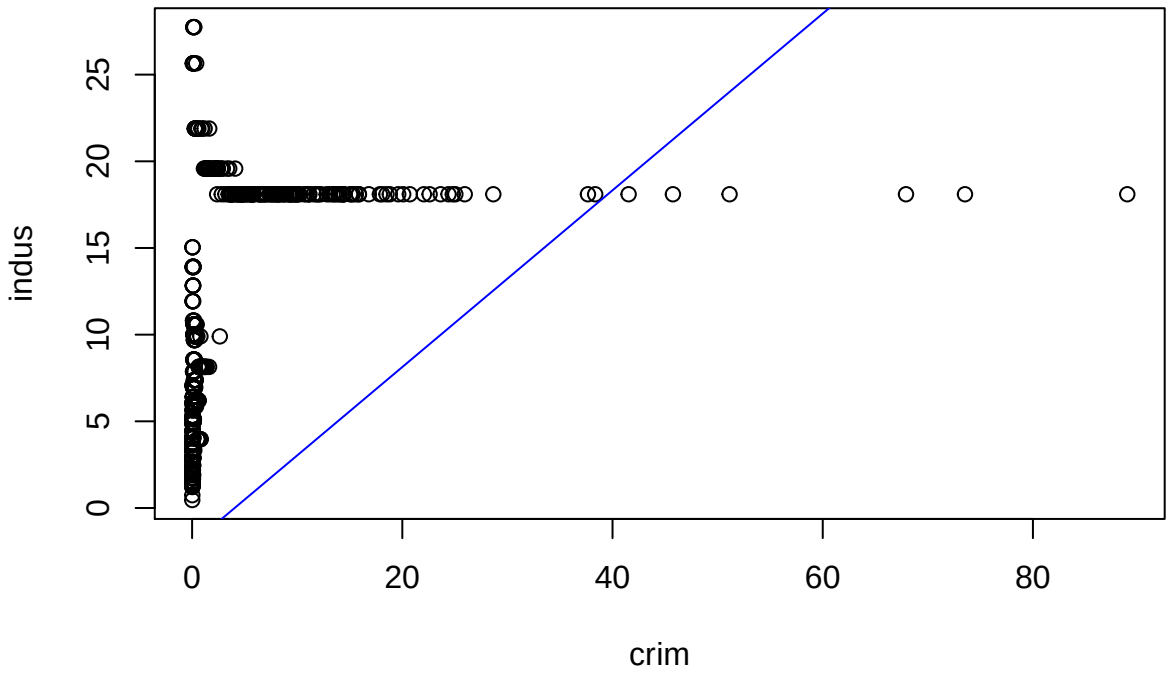

```
## integer(0)
```

```
lm2 <- lm(crim ~ indus, data = boston.df)
summary(lm2)
```

```
##
## Call:
```

```
## lm(formula = crim ~ indus, data = boston.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -11.972  -2.698  -0.736   0.712  81.813
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.06374    0.66723  -3.093  0.00209 **
## indus        0.50978    0.05102   9.991  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.866 on 504 degrees of freedom
## Multiple R-squared:  0.1653, Adjusted R-squared:  0.1637
## F-statistic: 99.82 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
plot(crim, indus)+abline(coef = coef(lm2), col="blue")
```



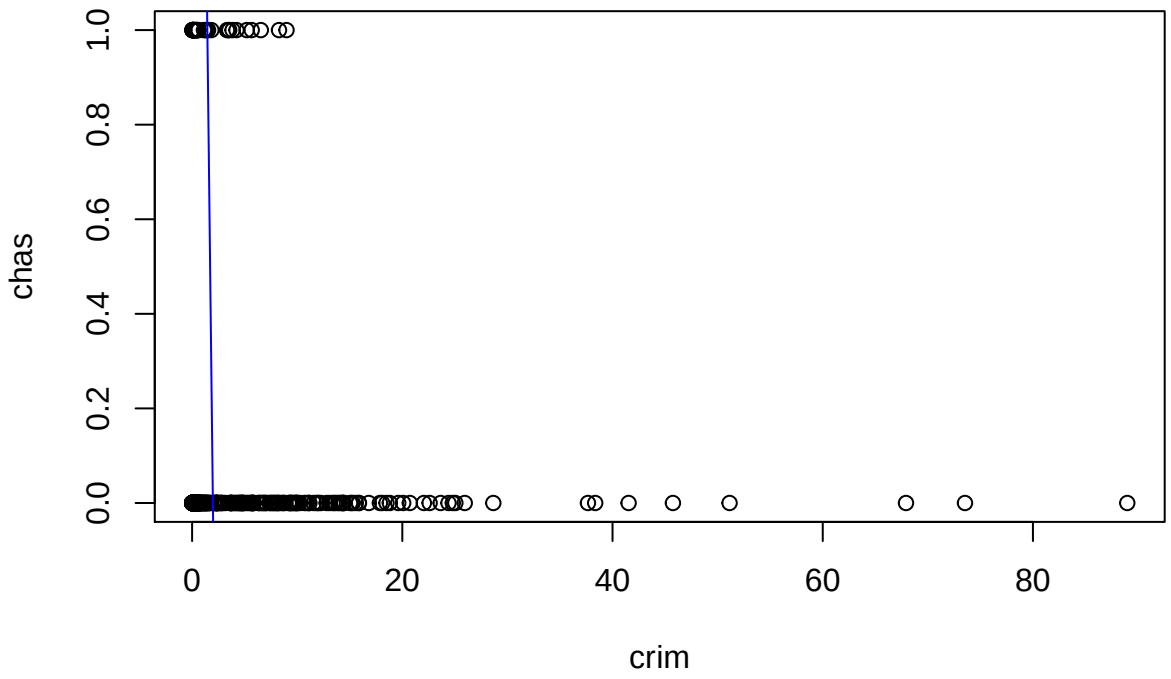

```
## integer(0)
```

```
lm3 <- lm(crim ~ chas, data = boston.df)
summary(lm3)
```

```
##
## Call:
## lm(formula = crim ~ chas, data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -3.738 -3.661 -3.435  0.018 85.232
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)   3.7444     0.3961   9.453   <2e-16 ***
## chas         -1.8928     1.5061  -1.257    0.209
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.597 on 504 degrees of freedom
## Multiple R-squared:  0.003124,   Adjusted R-squared:  0.001146
## F-statistic: 1.579 on 1 and 504 DF,  p-value: 0.2094
```

```
plot(crim, chas)+abline(coef = coef(lm3), col="blue")
```

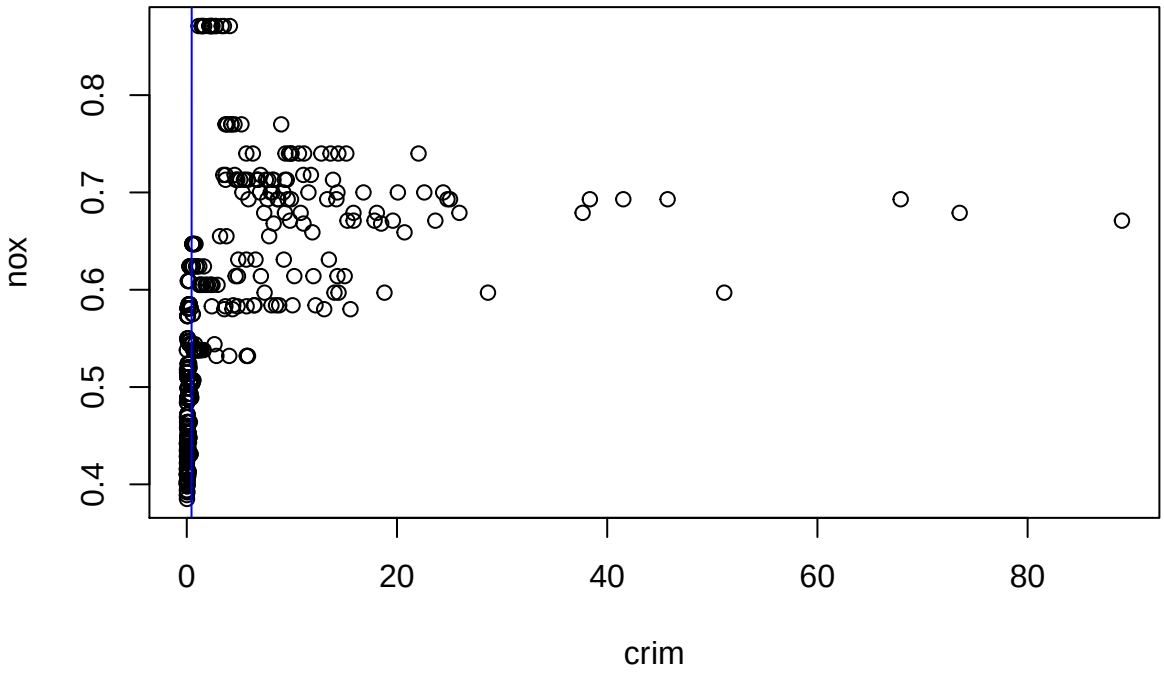



```
## integer(0)
```

```
lm4 <- lm(crim ~ nox, data = boston.df)
summary(lm4)
```

```
##
## Call:
## lm(formula = crim ~ nox, data = boston.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.371  -2.738  -0.974   0.559  81.728
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -13.720      1.699  -8.073 5.08e-15 ***
## nox           31.249      2.999  10.419  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.81 on 504 degrees of freedom
## Multiple R-squared:  0.1772, Adjusted R-squared:  0.1756
## F-statistic: 108.6 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
plot(crim, nox)+abline(coef = coef(lm4), col="blue")
```

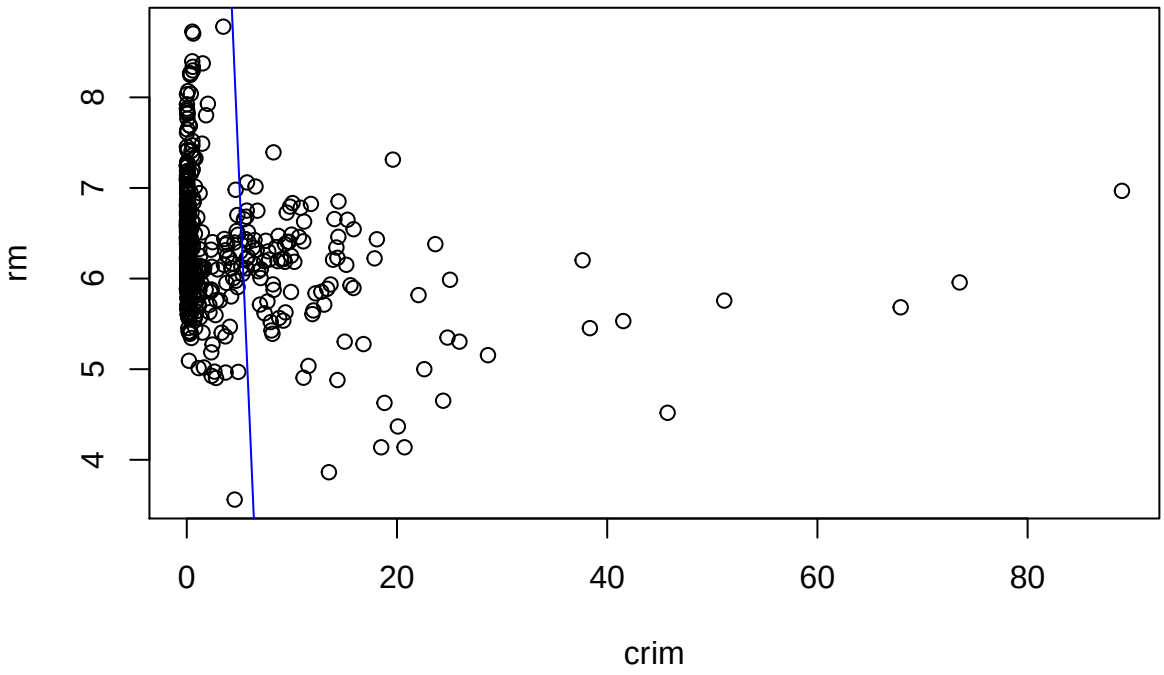



```
## integer(0)
```

```
lm5 <- lm(crim ~ rm, data = boston.df)
summary(lm5)
```

```
##
## Call:
## lm(formula = crim ~ rm, data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -6.604 -3.952 -2.654  0.989 87.197
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   20.482      3.365   6.088 2.27e-09 ***
## rm            -2.684      0.532  -5.045 6.35e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.401 on 504 degrees of freedom
## Multiple R-squared:  0.04807,    Adjusted R-squared:  0.04618
## F-statistic: 25.45 on 1 and 504 DF,  p-value: 6.347e-07
```

```
plot(crim, rm)+abline(coef = coef(lm5), col="blue")
```

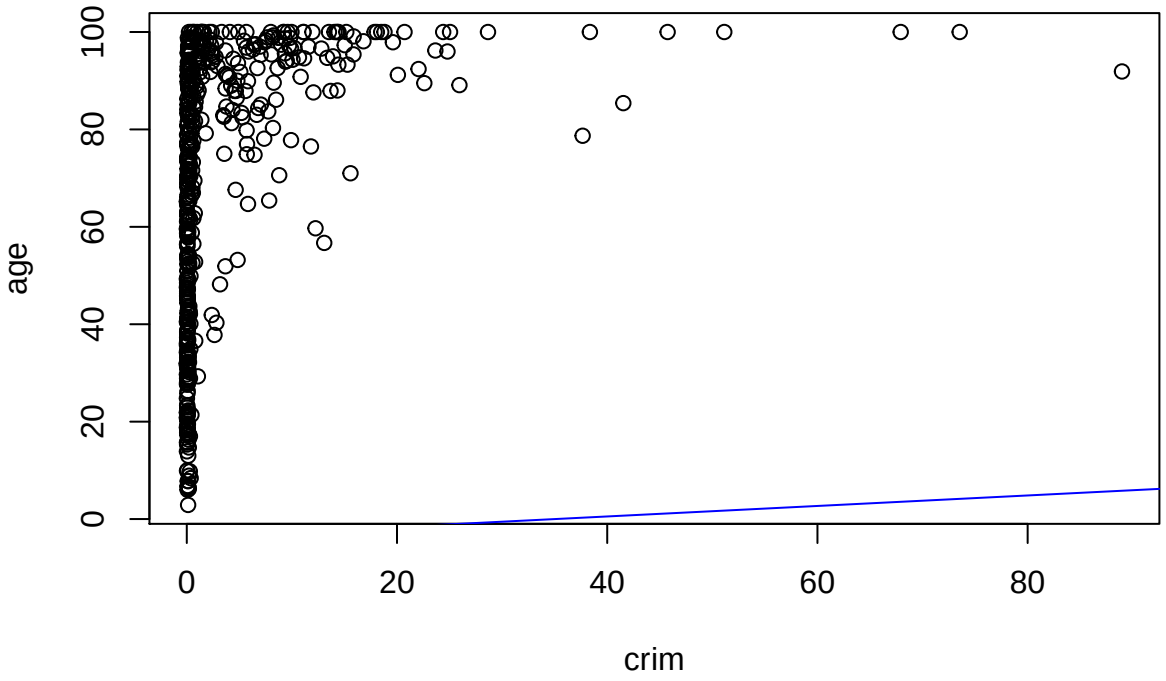



```
## integer(0)
```

```
lm6 <- lm(crim ~ age, data = boston.df)
summary(lm6)
```

```
##
## Call:
## lm(formula = crim ~ age, data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -6.789 -4.257 -1.230  1.527 82.849
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.77791    0.94398  -4.002 7.22e-05 ***
## age          0.10779    0.01274   8.463 2.85e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.057 on 504 degrees of freedom
## Multiple R-squared:  0.1244, Adjusted R-squared:  0.1227
## F-statistic: 71.62 on 1 and 504 DF,  p-value: 2.855e-16
```

```
plot(crim, age)+abline(coef = coef(lm6), col="blue")
```

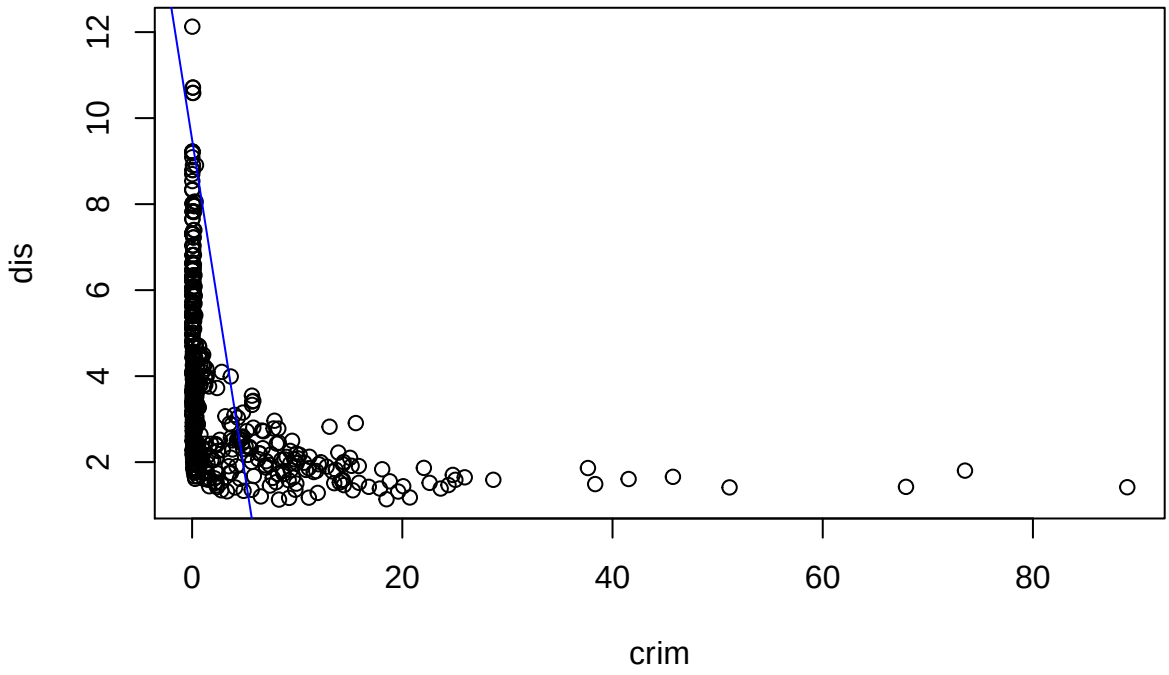



```
## integer(0)
```

```
lm7 <- lm(crim ~ dis, data = boston.df)
summary(lm7)
```

```
##
## Call:
## lm(formula = crim ~ dis, data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -6.708 -4.134 -1.527  1.516 81.674
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   9.4993     0.7304  13.006   <2e-16 ***
## dis          -1.5509     0.1683  -9.213   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.965 on 504 degrees of freedom
## Multiple R-squared:  0.1441, Adjusted R-squared:  0.1425
## F-statistic: 84.89 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
plot(crim, dis)+abline(coef = coef(lm7), col="blue")
```

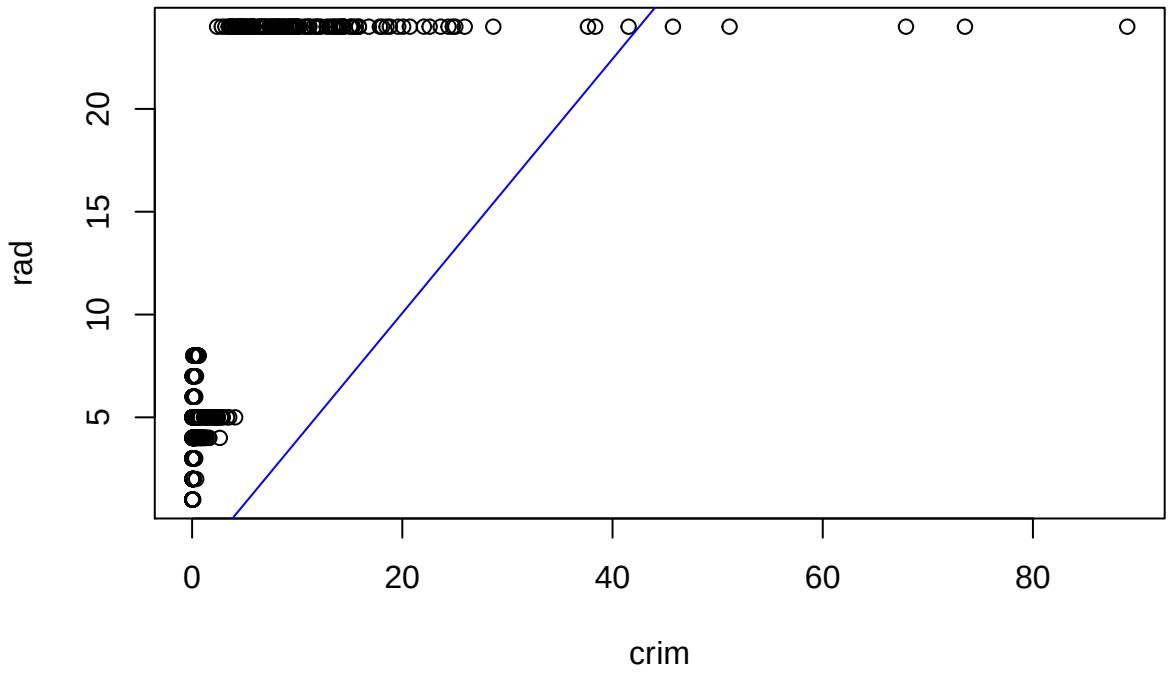



```
## integer(0)
```

```
lm8 <- lm(crim ~ rad, data = boston.df)
summary(lm8)
```

```
##
## Call:
## lm(formula = crim ~ rad, data = boston.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.164  -1.381  -0.141   0.660  76.433
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.28716    0.44348  -5.157 3.61e-07 ***
## rad          0.61791    0.03433  17.998  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.718 on 504 degrees of freedom
## Multiple R-squared:  0.3913, Adjusted R-squared:   0.39
## F-statistic: 323.9 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
plot(crim, rad)+abline(coef = coef(lm8), col="blue")
```

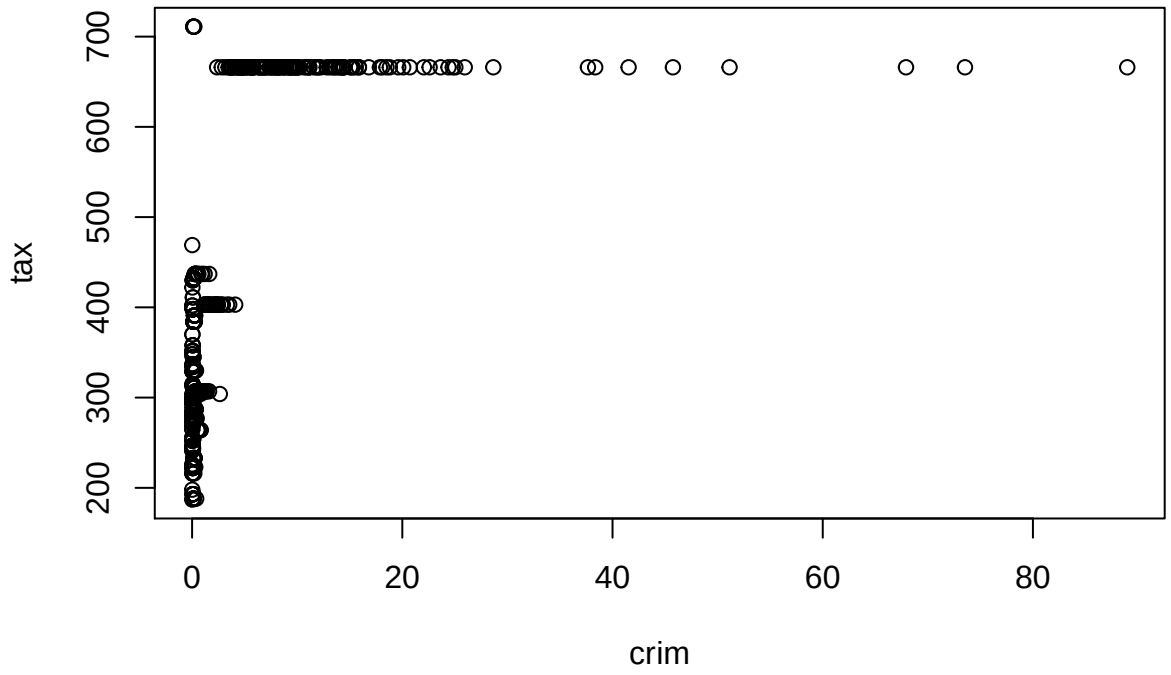



```
## integer(0)
```

```
lm9 <- lm(crim ~ tax, data = boston.df)
summary(lm9)
```

```
##
## Call:
## lm(formula = crim ~ tax, data = boston.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.513  -2.738  -0.194   1.065  77.696
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -8.528369   0.815809  -10.45   <2e-16 ***
## tax          0.029742   0.001847   16.10   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.997 on 504 degrees of freedom
## Multiple R-squared:  0.3396, Adjusted R-squared:  0.3383
## F-statistic: 259.2 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
plot(crim, tax)+abline(coef = coef(lm9), col="blue")
```

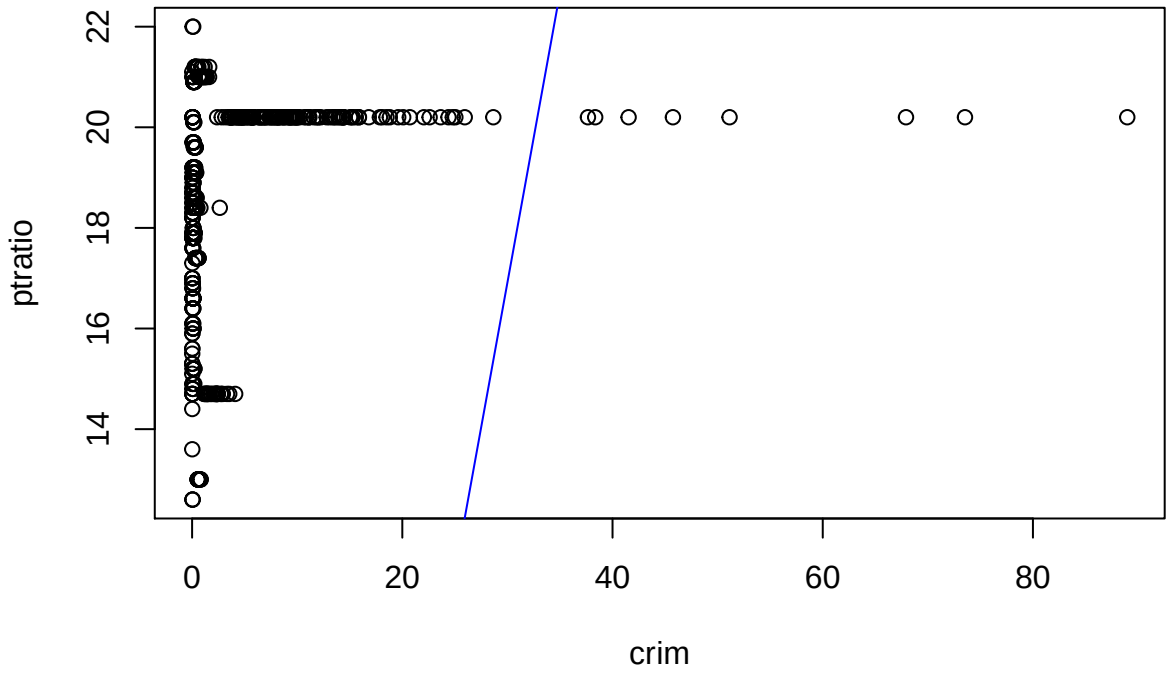



```
## integer(0)
```

```
lm10 <- lm(crim ~ ptratio, data = boston.df)
summary(lm10)
```

```
##
## Call:
## lm(formula = crim ~ ptratio, data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -7.654 -3.985 -1.912  1.825 83.353
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -17.6469     3.1473  -5.607 3.40e-08 ***
## ptratio       1.1520     0.1694   6.801 2.94e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.24 on 504 degrees of freedom
## Multiple R-squared:  0.08407,    Adjusted R-squared:  0.08225
## F-statistic: 46.26 on 1 and 504 DF,  p-value: 2.943e-11
```

```
plot(crim ,ptratio)+abline(coef = coef(lm10), col="blue")
```

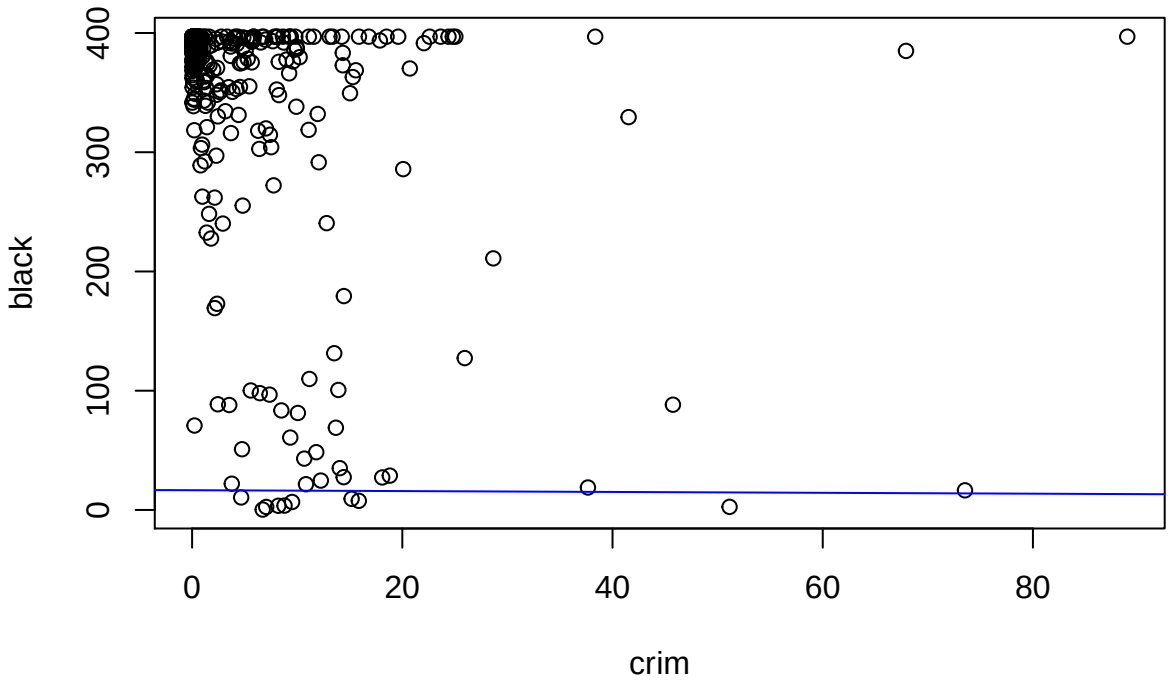



```
## integer(0)
```

```
lm11 <- lm(crim ~ black, data = boston.df)
summary(lm11)
```

```
##
## Call:
## lm(formula = crim ~ black, data = boston.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -13.756  -2.299  -2.095  -1.296  86.822
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 16.553529   1.425903  11.609   <2e-16 ***
## black       -0.036280   0.003873  -9.367   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.946 on 504 degrees of freedom
## Multiple R-squared:  0.1483, Adjusted R-squared:  0.1466
## F-statistic: 87.74 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
plot(crim, black)+abline(coef = coef(lm11), col="blue")
```

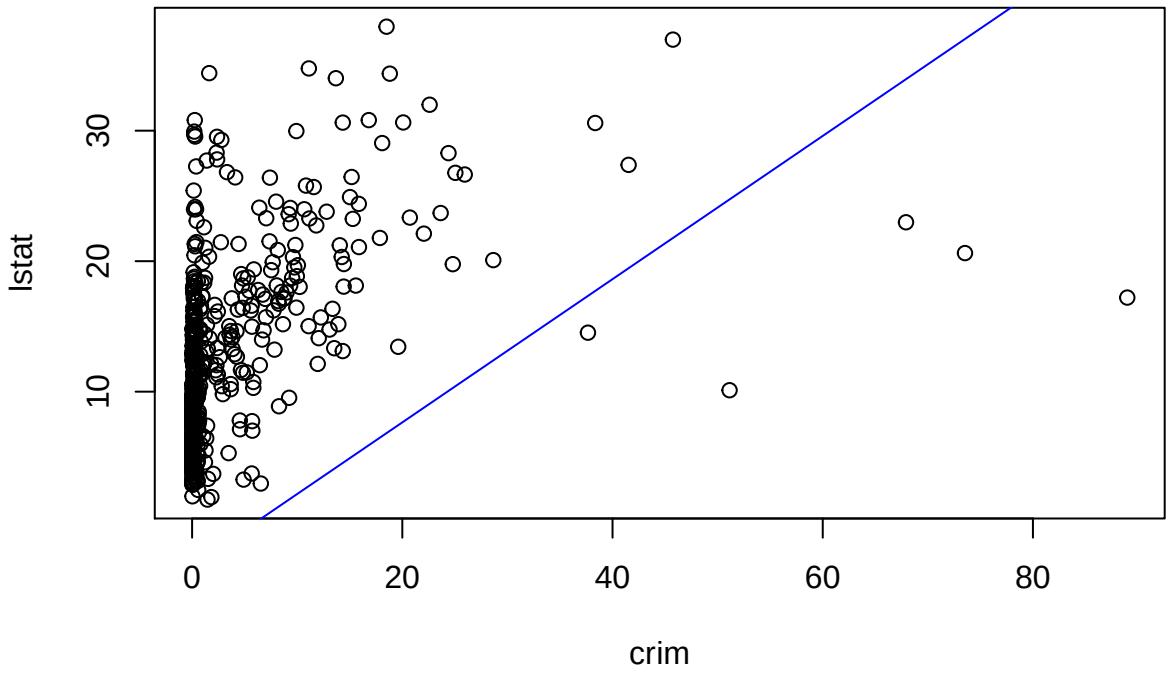



```
## integer(0)
```

```
lm12 <- lm(crim ~ lstat, data = boston.df)
summary(lm12)
```

```
##
## Call:
## lm(formula = crim ~ lstat, data = boston.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -13.925  -2.822  -0.664   1.079  82.862
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.33054    0.69376  -4.801 2.09e-06 ***
## lstat        0.54880    0.04776  11.491  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.664 on 504 degrees of freedom
## Multiple R-squared:  0.2076, Adjusted R-squared:  0.206
## F-statistic:   132 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
plot(crim, lstat)+abline(coef = coef(lm12), col="blue")
```

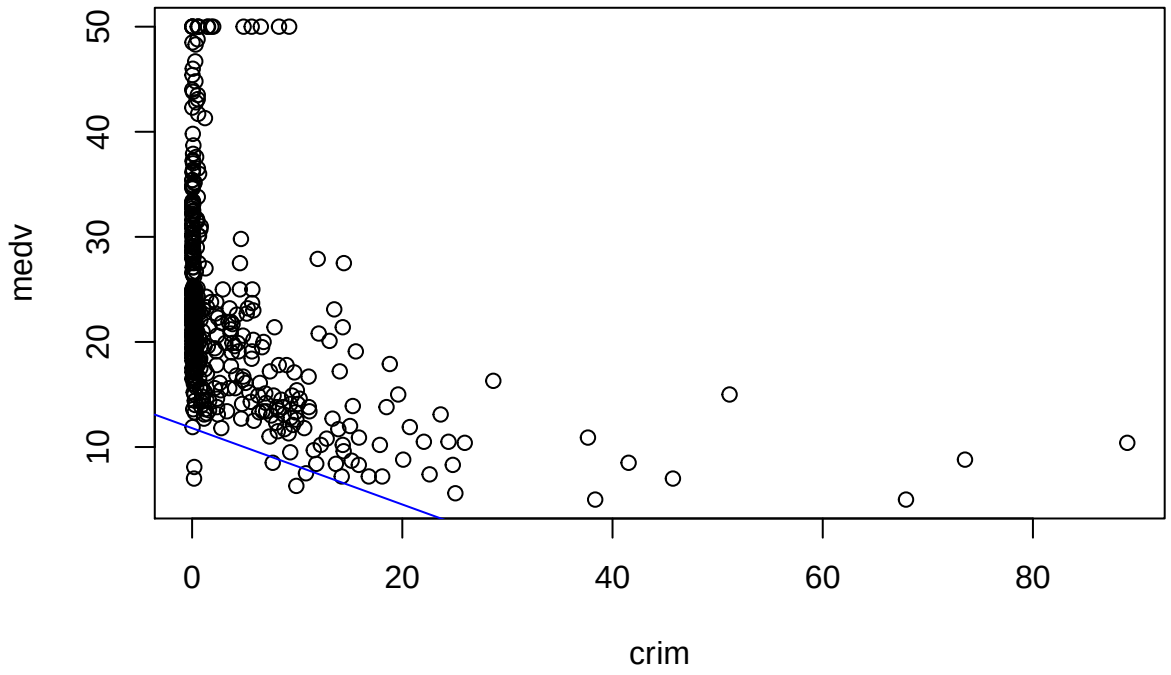



```
## integer(0)
```

```
lm13 <- lm(crim ~ medv, data = boston.df)
summary(lm13)
```

```
##
## Call:
## lm(formula = crim ~ medv, data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.071 -4.022 -2.343  1.298 80.957
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11.79654    0.93419   12.63   <2e-16 ***
## medv        -0.36316    0.03839   -9.46   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.934 on 504 degrees of freedom
## Multiple R-squared:  0.1508, Adjusted R-squared:  0.1491
## F-statistic: 89.49 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
plot(crim, medv)+abline(coef = coef(lm13), col="blue")
```




```
## integer(0)
```

```
sprintf("a ) There is no strong relationship between chas and crime")
```

```
## [1] "a ) There is no strong relationship between chas and crime"
```

```
boston.lm <- lm(crim ~ ., data = boston.df)
summary(boston.lm)
```

```
##
## Call:
## lm(formula = crim ~ ., data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.924 -2.120 -0.353  1.019 75.051
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  17.033228   7.234903   2.354 0.018949 *
## zn            0.044855   0.018734   2.394 0.017025 *
## indus        -0.063855   0.083407  -0.766 0.444294
## chas         -0.749134   1.180147  -0.635 0.525867
## nox         -10.313535   5.275536  -1.955 0.051152 .
## rm            0.430131   0.612830   0.702 0.483089
## age           0.001452   0.017925   0.081 0.935488
## dis          -0.987176   0.281817  -3.503 0.000502 ***
## rad           0.588209   0.088049   6.680 6.46e-11 ***
## tax          -0.003780   0.005156  -0.733 0.463793
## ptratio      -0.271081   0.186450  -1.454 0.146611
## black        -0.007538   0.003673  -2.052 0.040702 *
## lstat         0.126211   0.075725   1.667 0.096208 .
## medv         -0.198887   0.060516  -3.287 0.001087 **
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.439 on 492 degrees of freedom
## Multiple R-squared:  0.454,  Adjusted R-squared:  0.4396
## F-statistic: 31.47 on 13 and 492 DF,  p-value: < 2.2e-16
```

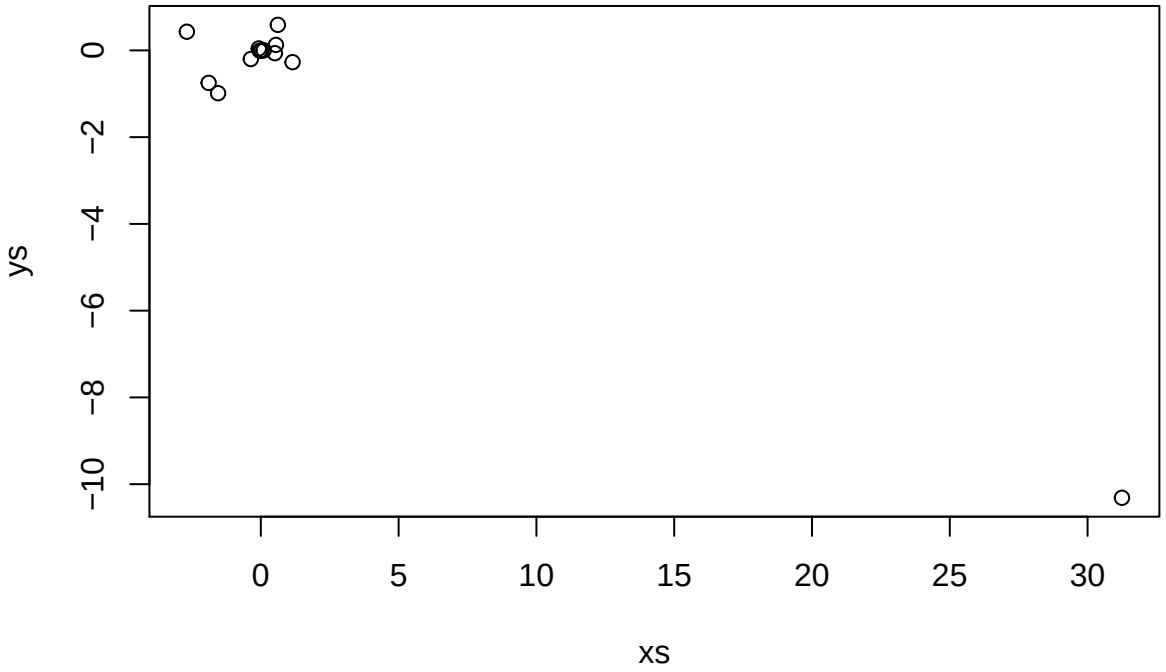
```
sprintf("b_) For all predictors except: indus, chas, nox, rm, age,
        tax, ptratio, lstat we can reject null hypothesis")
```

```
## [1] "b_) For all predictors except: indus, chas, nox, rm, age, \n        tax, ptratio, lstat we can r
```

```
xs <- c(coef(lm1)[2],coef(lm2)[2],coef(lm3)[2],coef(lm4)[2],coef(lm5)[2],
        coef(lm6)[2],coef(lm7)[2],coef(lm8)[2],
  coef(lm9)[2],coef(lm10)[2],coef(lm11)[2],coef(lm12)[2],coef(lm13)[2])

coefs <- coef(boston.lm)
ys <- coef(boston.lm)[-1] # drop intercept
length(ys)
```

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

```
plot(xs, ys)
```




```
sprintf(" Fit a model of the form b0 + b1*X + b2*X^2 + b3*X^3 ")
```

```
## [1] " Fit a model of the form b0 + b1*X + b2*X^2 + b3*X^3 "
```

```
lm1 <- lm(crim ~ zn + I(zn^2) + I(zn^3), data = boston.df)
summary(lm1)
```

```
##
## Call:
## lm(formula = crim ~ zn + I(zn^2) + I(zn^3), data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
```

```
## -4.821 -4.614 -1.294  0.473 84.130
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.846e+00  4.330e-01  11.192  < 2e-16 ***
## zn          -3.322e-01  1.098e-01  -3.025  0.00261 **
## I(zn^2)      6.483e-03  3.861e-03   1.679  0.09375 .
## I(zn^3)     -3.776e-05  3.139e-05  -1.203  0.22954
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.372 on 502 degrees of freedom
## Multiple R-squared:  0.05824,    Adjusted R-squared:  0.05261
## F-statistic: 10.35 on 3 and 502 DF,  p-value: 1.281e-06
```

```
lm2 <- lm(crim ~ indus + I(indus^2) + I(indus^3), data = boston.df)
summary(lm2)
```

```
##
## Call:
## lm(formula = crim ~ indus + I(indus^2) + I(indus^3), data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.278 -2.514  0.054  0.764 79.713
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.6625683  1.5739833   2.327   0.0204 *
## indus       -1.9652129  0.4819901  -4.077 5.30e-05 ***
## I(indus^2)   0.2519373  0.0393221   6.407 3.42e-10 ***
## I(indus^3)  -0.0069760  0.0009567  -7.292 1.20e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.423 on 502 degrees of freedom
## Multiple R-squared:  0.2597, Adjusted R-squared:  0.2552
## F-statistic: 58.69 on 3 and 502 DF,  p-value: < 2.2e-16
```

```
lm3 <- lm(crim ~ chas + I(chas^2) + I(chas^3), data = boston.df)
summary(lm3)
```

```
##
## Call:
## lm(formula = crim ~ chas + I(chas^2) + I(chas^3), data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -3.738 -3.661 -3.435  0.018 85.232
##
## Coefficients: (2 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.7444     0.3961   9.453   <2e-16 ***
## chas         -1.8928     1.5061  -1.257    0.209
## I(chas^2)         NA         NA      NA       NA
```

```
## I(chas^3)         NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.597 on 504 degrees of freedom
## Multiple R-squared:  0.003124,   Adjusted R-squared:  0.001146
## F-statistic: 1.579 on 1 and 504 DF,  p-value: 0.2094
```

```
lm4 <- lm(crim ~ nox + I(nox^2) + I(nox), data = boston.df)
summary(lm4)
```

```
##
## Call:
## lm(formula = crim ~ nox + I(nox^2) + I(nox), data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -7.512 -3.394 -1.467  1.444 80.946
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -41.325      7.572  -5.457  7.6e-08 ***
## nox          127.877     26.016   4.915  1.2e-06 ***
## I(nox^2)     -80.958     21.655  -3.738 0.000206 ***
## I(nox)            NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.711 on 503 degrees of freedom
## Multiple R-squared:  0.1995, Adjusted R-squared:  0.1963
## F-statistic: 62.66 on 2 and 503 DF,  p-value: < 2.2e-16
```

```
lm5 <- lm(crim ~ rm + I(rm^2) + I(rm^3), data = boston.df)
summary(lm5)
```

```
##
## Call:
## lm(formula = crim ~ rm + I(rm^2) + I(rm^3), data = boston.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.485  -3.468  -2.221  -0.015  87.219
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 112.6246    64.5172   1.746   0.0815 .
## rm          -39.1501    31.3115  -1.250   0.2118
## I(rm^2)       4.5509     5.0099   0.908   0.3641
## I(rm^3)      -0.1745     0.2637  -0.662   0.5086
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.33 on 502 degrees of freedom
## Multiple R-squared:  0.06779,    Adjusted R-squared:  0.06222
## F-statistic: 12.17 on 3 and 502 DF,  p-value: 1.067e-07
```

```r
lm6 <- lm(crim ~ age + I(age^2) + I(age^3), data = boston.df)
summary(lm6)
```

```
##
## Call:
## lm(formula = crim ~ age + I(age^2) + I(age^3), data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.762 -2.673 -0.516  0.019 82.842
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.549e+00  2.769e+00  -0.920  0.35780
## age          2.737e-01  1.864e-01   1.468  0.14266
## I(age^2)    -7.230e-03  3.637e-03  -1.988  0.04738 *
## I(age^3)     5.745e-05  2.109e-05   2.724  0.00668 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.84 on 502 degrees of freedom
## Multiple R-squared:  0.1742, Adjusted R-squared:  0.1693
## F-statistic: 35.31 on 3 and 502 DF,  p-value: < 2.2e-16
```

```r
lm7 <- lm(crim ~ dis + I(dis^2) + I(dis^3), data = boston.df)
summary(lm7)
```

```
##
## Call:
## lm(formula = crim ~ dis + I(dis^2) + I(dis^3), data = boston.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.757  -2.588   0.031   1.267  76.378
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  30.0476     2.4459  12.285  < 2e-16 ***
## dis         -15.5543     1.7360  -8.960  < 2e-16 ***
## I(dis^2)      2.4521     0.3464   7.078 4.94e-12 ***
## I(dis^3)     -0.1186     0.0204  -5.814 1.09e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.331 on 502 degrees of freedom
## Multiple R-squared:  0.2778, Adjusted R-squared:  0.2735
## F-statistic: 64.37 on 3 and 502 DF,  p-value: < 2.2e-16
```

```r
lm8 <- lm(crim ~ rad + I(rad^2) + I(rad^3), data = boston.df)
summary(lm8)
```

```
##
## Call:
## lm(formula = crim ~ rad + I(rad^2) + I(rad^3), data = boston.df)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.381  -0.412  -0.269   0.179  76.217
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.605545   2.050108  -0.295    0.768
## rad          0.512736   1.043597   0.491    0.623
## I(rad^2)    -0.075177   0.148543  -0.506    0.613
## I(rad^3)     0.003209   0.004564   0.703    0.482
##
## Residual standard error: 6.682 on 502 degrees of freedom
## Multiple R-squared:    0.4,  Adjusted R-squared:  0.3965
## F-statistic: 111.6 on 3 and 502 DF,  p-value: < 2.2e-16
```

```
lm9 <- lm(crim ~ tax + I(tax^2) + I(tax^3), data = boston.df)
summary(lm9)
```

```
##
## Call:
## lm(formula = crim ~ tax + I(tax^2) + I(tax^3), data = boston.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -13.273  -1.389   0.046   0.536  76.950
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.918e+01  1.180e+01   1.626    0.105
## tax         -1.533e-01  9.568e-02  -1.602    0.110
## I(tax^2)     3.608e-04  2.425e-04   1.488    0.137
## I(tax^3)    -2.204e-07  1.889e-07  -1.167    0.244
##
## Residual standard error: 6.854 on 502 degrees of freedom
## Multiple R-squared:  0.3689, Adjusted R-squared:  0.3651
## F-statistic:  97.8 on 3 and 502 DF,  p-value: < 2.2e-16
```

```
lm10 <- lm(crim ~ ptratio + I(ptratio^2) + I(ptratio^3), data = boston.df)
summary(lm10)
```

```
##
## Call:
## lm(formula = crim ~ ptratio + I(ptratio^2) + I(ptratio^3), data = boston.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -6.833 -4.146 -1.655  1.408 82.697
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  477.18405  156.79498   3.043  0.00246 **
## ptratio      -82.36054   27.64394  -2.979  0.00303 **
## I(ptratio^2)   4.63535    1.60832   2.882  0.00412 **
## I(ptratio^3)  -0.08476    0.03090  -2.743  0.00630 **
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.122 on 502 degrees of freedom
## Multiple R-squared:  0.1138, Adjusted R-squared:  0.1085
## F-statistic: 21.48 on 3 and 502 DF,  p-value: 4.171e-13
```

```
lm11 <- lm(crim ~ black + I(black^2) + I(black^3), data = boston.df)
summary(lm11)
```

```
##
## Call:
## lm(formula = crim ~ black + I(black^2) + I(black^3), data = boston.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -13.096  -2.343  -2.128  -1.439  86.790
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.826e+01  2.305e+00   7.924  1.5e-14 ***
## black       -8.356e-02  5.633e-02  -1.483    0.139
## I(black^2)   2.137e-04  2.984e-04   0.716    0.474
## I(black^3)  -2.652e-07  4.364e-07  -0.608    0.544
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.955 on 502 degrees of freedom
## Multiple R-squared:  0.1498, Adjusted R-squared:  0.1448
## F-statistic: 29.49 on 3 and 502 DF,  p-value: < 2.2e-16
```

```
lm12 <- lm(crim ~ lstat + I(lstat^2) + I(lstat^3), data = boston.df)
summary(lm12)
```

```
##
## Call:
## lm(formula = crim ~ lstat + I(lstat^2) + I(lstat^3), data = boston.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.234  -2.151  -0.486   0.066  83.353
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.2009656  2.0286452   0.592   0.5541
## lstat       -0.4490656  0.4648911  -0.966   0.3345
## I(lstat^2)   0.0557794  0.0301156   1.852   0.0646 .
## I(lstat^3)  -0.0008574  0.0005652  -1.517   0.1299
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.629 on 502 degrees of freedom
## Multiple R-squared:  0.2179, Adjusted R-squared:  0.2133
## F-statistic: 46.63 on 3 and 502 DF,  p-value: < 2.2e-16
```

```
lm13 <- lm(crim ~ medv + I(medv^2) + I(medv^3), data = boston.df)
summary(lm13)
```

```
##
## Call:
## lm(formula = crim ~ medv + I(medv^2) + I(medv^3), data = boston.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -24.427  -1.976  -0.437   0.439  73.655
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 53.1655381  3.3563105  15.840  < 2e-16 ***
## medv        -5.0948305  0.4338321 -11.744  < 2e-16 ***
## I(medv^2)    0.1554965  0.0171904   9.046  < 2e-16 ***
## I(medv^3)   -0.0014901  0.0002038  -7.312 1.05e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.569 on 502 degrees of freedom
## Multiple R-squared:  0.4202, Adjusted R-squared:  0.4167
## F-statistic: 121.3 on 3 and 502 DF,  p-value: < 2.2e-16
```

```
x <- cbind(x1 = 3, x2 = c(4:1, 2:5))
x
```

```
##      x1 x2
## [1,]  3  4
## [2,]  3  3
## [3,]  3  2
## [4,]  3  1
## [5,]  3  2
## [6,]  3  3
## [7,]  3  4
## [8,]  3  5
```

```
rowSums(x); colSums(x)
```

```
## [1] 7 6 5 4 5 6 7 8
```

```
## x1 x2
## 24 24
```

```
# choose rows of dataframe using an array od logicals
df <- data.frame(x = c(1,2,NA, 3, 4, NA), y=1:6)
df[c(F,T,T,F,F,T), ]
```

```
##    x y
## 2  2 2
## 3 NA 3
## 6 NA 6
```