Load Data and Set Sample

Statistical modelling

Josep Franquet

March 20, 2025

Contents

1 Presentation - Títol nivell 1

1.1 R Markdowns document - Títol nivell 2

This is an R Markdown document. We are showing some examples of GLMz. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com. Use * to provide emphasis such as *italics* and **bold**.

Create lists: Unordered * and + or ordered 1. 2.

1. Item 1
2. Item 2 + Item 2a + Item 2b

2 Header 1

2.1 Header 2

2.2 Data Description: 100,000 UK Used Car Data set

This data dictionary describes data (https://www.kaggle.com/adityadesai13/used-car-dataset-ford-and-mercedes) - A sample of 5000 trips has been randomly selected from Mercedes, BMW, Volkwagen and Audi manufacturers. So, firstly you have to combine used car from the 4 manufacturers into 1 dataframe.

The cars with engine size 0 are in fact electric cars, nevertheless Mercedes C class, and other given cars are not electric cars,so data imputation is requered.

- manufacturer Factor: Audi, BMW, Mercedes or Volkswagen
- model Car model
- year registration year
- price price in £

- transmission type of gearbox
- mileage distance used
- fuelType engine fuel
- tax road tax
- mpg Consumption in miles per gallon

- engineSize size in litres

3 Load Required Packages: to be increased over the course

```
# Load Required Packages: to be increased over the course
options(contrasts=c("contr.treatment","contr.treatment"))

requiredPackages <- c("effects","FactoMineR","car", "factoextra","RColorBrewer","ggplot2","dplyr","ggmap

#use this function to check if each package is on the local machine
#if a package is installed, it will be loaded
#if any are not, the missing package(s) will be installed and loaded
package.check <- lapply(requiredPackages, FUN = function(x) {
  if (!require(x, character.only = TRUE)) {
    install.packages(x, dependencies = TRUE)
    library(x, character.only = TRUE)
  }
})
```

```
## S'està carregant el paquet requerit: effects
```

```
## S'està carregant el paquet requerit: carData
```

```
## lattice theme set by effectsTheme()
## See ?effectsTheme for details.
```

```
## S'està carregant el paquet requerit: FactoMineR
```

```
## S'està carregant el paquet requerit: car
```

```
## S'està carregant el paquet requerit: factoextra
```

```
## S'està carregant el paquet requerit: ggplot2
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
## S'està carregant el paquet requerit: RColorBrewer
```

```
## S'està carregant el paquet requerit: dplyr
```

```
##
## S'està adjuntant el paquet: 'dplyr'
```

```
## L'objecte següent està emmascarat per 'package:car':
##
##     recode
```

```
## Els següents objectes estan emmascarats des de 'package:stats':
##
##     filter, lag
```

```
## Els següents objectes estan emmascarats des de 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
## S'està carregant el paquet requerit: ggmap
```

```
## i Google's Terms of Service: <https://mapsplatform.google.com>
##   Stadia Maps' Terms of Service: <https://stadiamaps.com/terms-of-service/>
##   OpenStreetMap's Tile Usage Policy: <https://operations.osmfoundation.org/policies/tiles/>
## i Please cite ggmap if you use it! Use `citation("ggmap")` for details.
## S'està carregant el paquet requerit: ggthemes
##
```

```
## S'està carregant el paquet requerit: knitr
```

```
#verify they are loaded
search()
```

```
##  [1] ".GlobalEnv"           "package:knitr"        "package:ggthemes"
##  [4] "package:ggmap"        "package:dplyr"        "package:RColorBrewer"
##  [7] "package:factoextra"   "package:ggplot2"      "package:car"
## [10] "package:FactoMineR"   "package:effects"      "package:carData"
## [13] "package:stats"        "package:graphics"     "package:grDevices"
## [16] "package:utils"        "package:datasets"     "package:methods"
## [19] "Autoloads"            "package:base"
```

4 Load data

```
load("/home2/users/alumnes/1289945/Baixades/MyOldCars-1000Clean.RData")
```

5 Linear Models: Using numerical explanatory variables

```
vars_con
```

```
## [1] "mileage" "tax"     "mpg"     "age"
```

```
ll<-which(df$age==0);ll
```

```
##  [1]   3  15  26  45  79  88  91  99 103 127 131 139 141 151 184 188 198 209 259
## [20] 261 273 281 282 298 309 334 347 350 443 469 471 476 478 483 492 510 520 529
## [39] 581 586 653 678 696 705 731 753 769 774 797 798 806 812 814 820 821 857 864
## [58] 869 886 895 909 912 944 956 962 995
```

```
df$age[ll]<-0.5

ll<-which(df$tax==0);ll
```

```
##  [1]  24  31  61  67 163 176 177 201 215 375 387 396 674 702 717 721 727 729 738
## [20] 752 775 776 778 795 826 879 891 990
```

```
df$tax[ll]<-0.5

#1st linear model with my numeric variables:
m1<-lm(price~mileage+tax+mpg+age,data=df)
summary(m1)
```

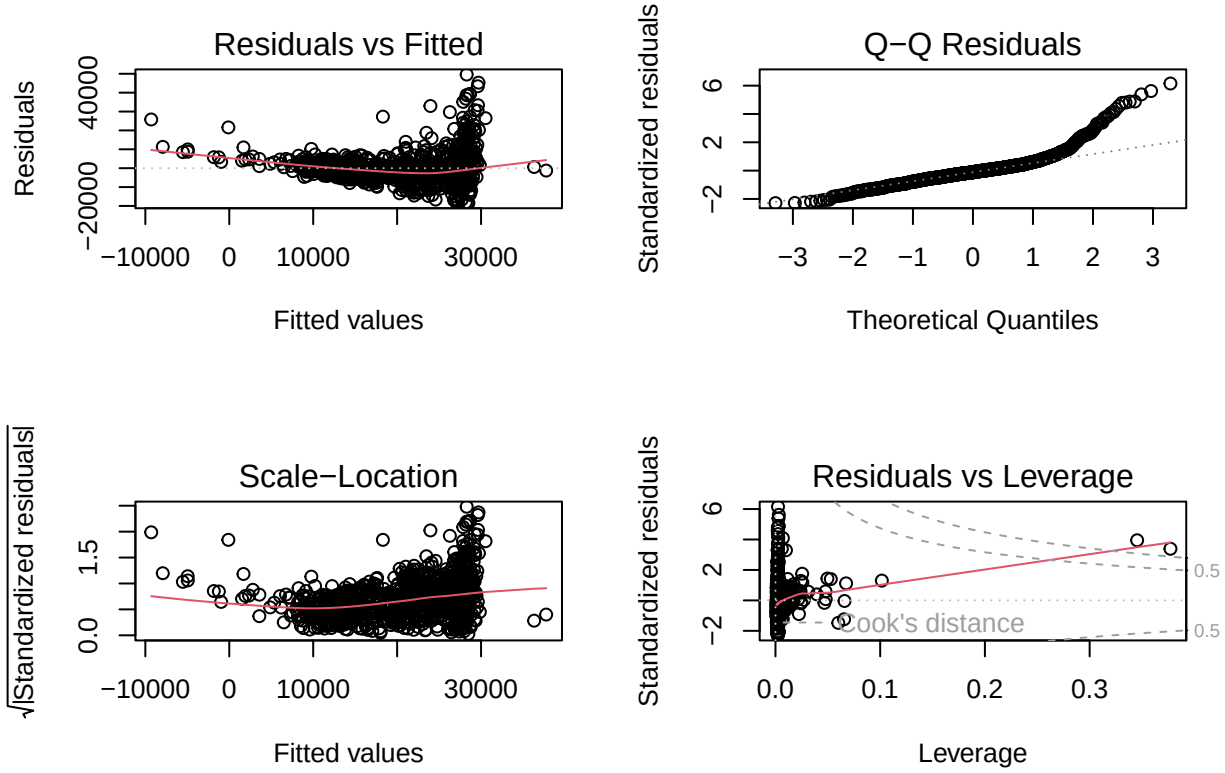
```
##
## Call:
## lm(formula = price ~ mileage + tax + mpg + age, data = df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -18493  -4349   -912   2591  49592
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.669e+04  1.010e+03  26.419  < 2e-16 ***
## mileage     -6.503e-02  1.919e-02  -3.388 0.000731 ***
## tax          3.978e+01  4.113e+00   9.672  < 2e-16 ***
## mpg         -5.420e+01  1.176e+01  -4.610 4.55e-06 ***
## age         -2.048e+03  1.913e+02 -10.703  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8074 on 995 degrees of freedom
## Multiple R-squared:  0.4325, Adjusted R-squared:  0.4302
```

```
## F-statistic: 189.6 on 4 and 995 DF,  p-value: < 2.2e-16
```

```
vif(m1) #Variance inflation factor: multicorrelation
```

```
##  mileage      tax      mpg      age
## 2.887119 1.174255 1.166440 2.802612
```

```
par(mfrow=c(2,2))
plot(m1,id.n=0)
```

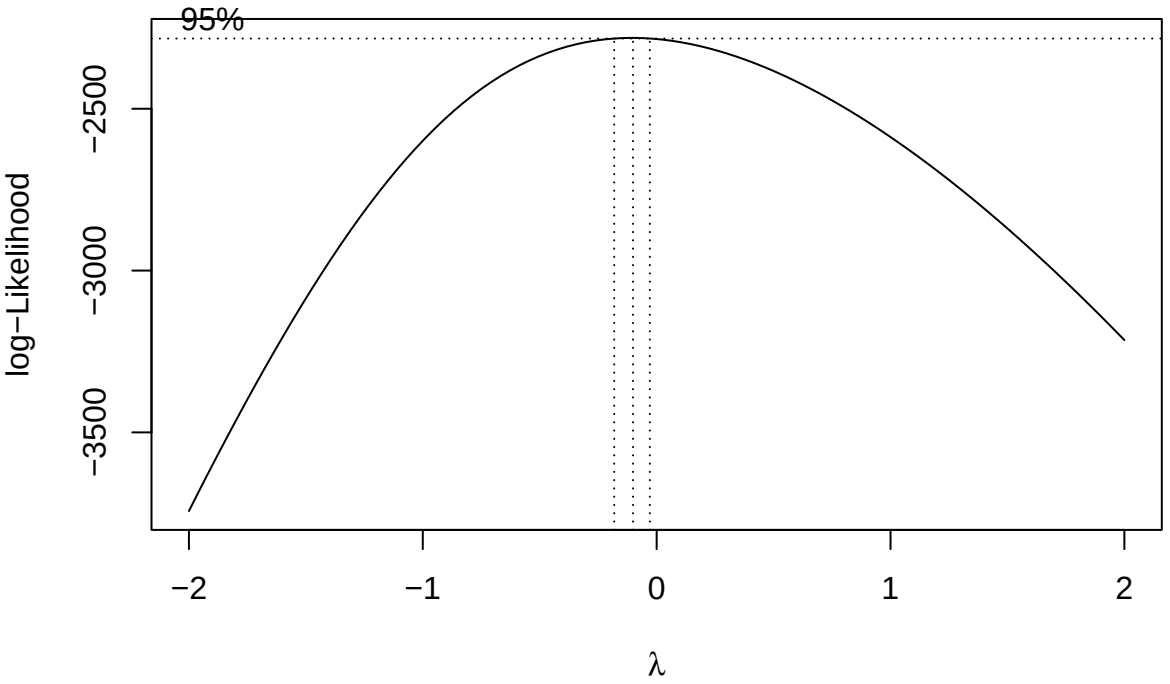



```
# Basic graphs for model validation
par(mfrow=c(1,1))

library(MASS)
```

```
##
## S'està adjuntant el paquet: 'MASS'
```

```
## L'objecte següent està emmascarat per 'package:dplyr':
##
##     select
```

```
# Target variable transformation?
boxcox(price~mileage+tax+mpg+age,data=df)
```




```
# Lambda=0 - log transformation is needed

# New model:
m2<-lm(log(price)~mileage+tax+mpg+age,data=df)
summary(m2)
```

```
##
## Call:
## lm(formula = log(price) ~ mileage + tax + mpg + age, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.05876 -0.17039  0.01573  0.16996  1.06270
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.007e+01  3.896e-02 258.531  < 2e-16 ***
## mileage     -2.520e-06  7.402e-07  -3.404 0.000691 ***
## tax          2.137e-03  1.586e-04  13.469  < 2e-16 ***
## mpg         -1.370e-03  4.534e-04  -3.022 0.002578 **
## age         -1.238e-01  7.378e-03 -16.779  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3114 on 995 degrees of freedom
## Multiple R-squared:  0.5906, Adjusted R-squared:  0.589
## F-statistic: 358.9 on 4 and 995 DF,  p-value: < 2.2e-16
```

```
vif(m2) #Not changed because explanatory variables have not changed
```

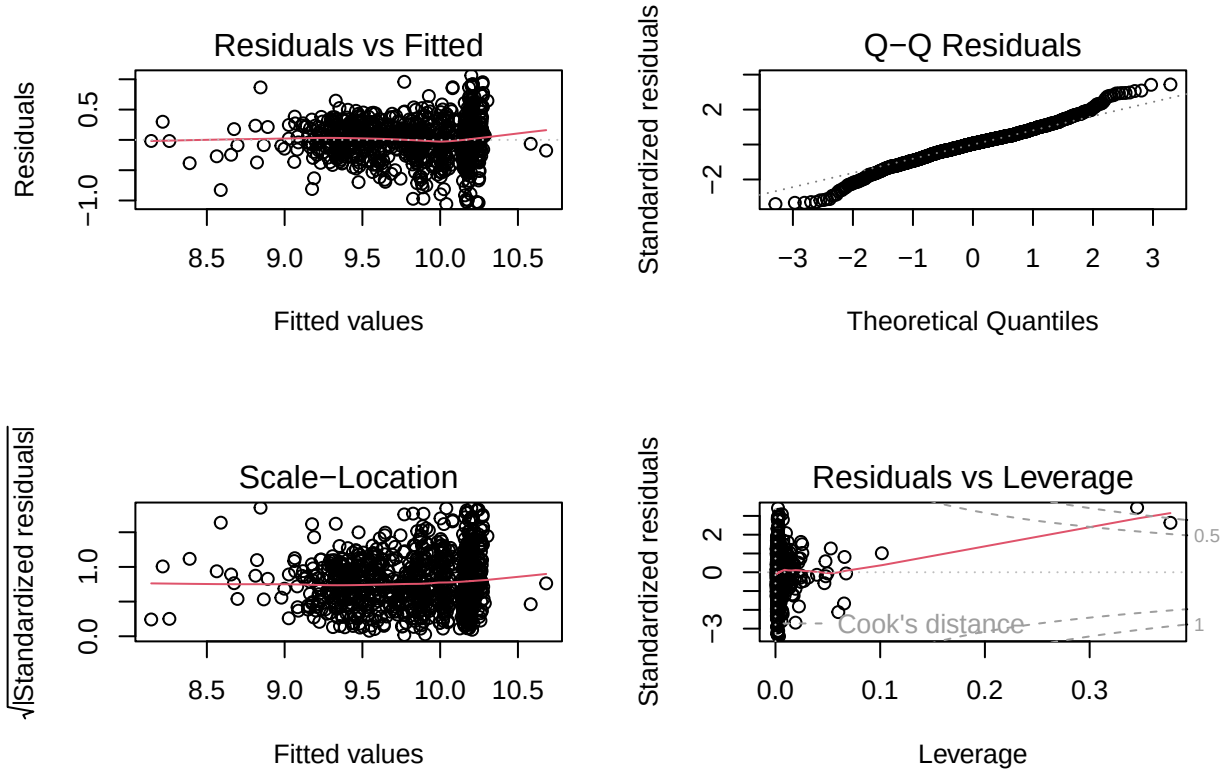
```
##  mileage      tax      mpg      age
## 2.887119 1.174255 1.166440 2.802612
```

```
# Transformations to my regresors?
boxTidwell(log(price)~mileage+tax+mpg+age,data=df[!df$mout=="YesMOut",])
```

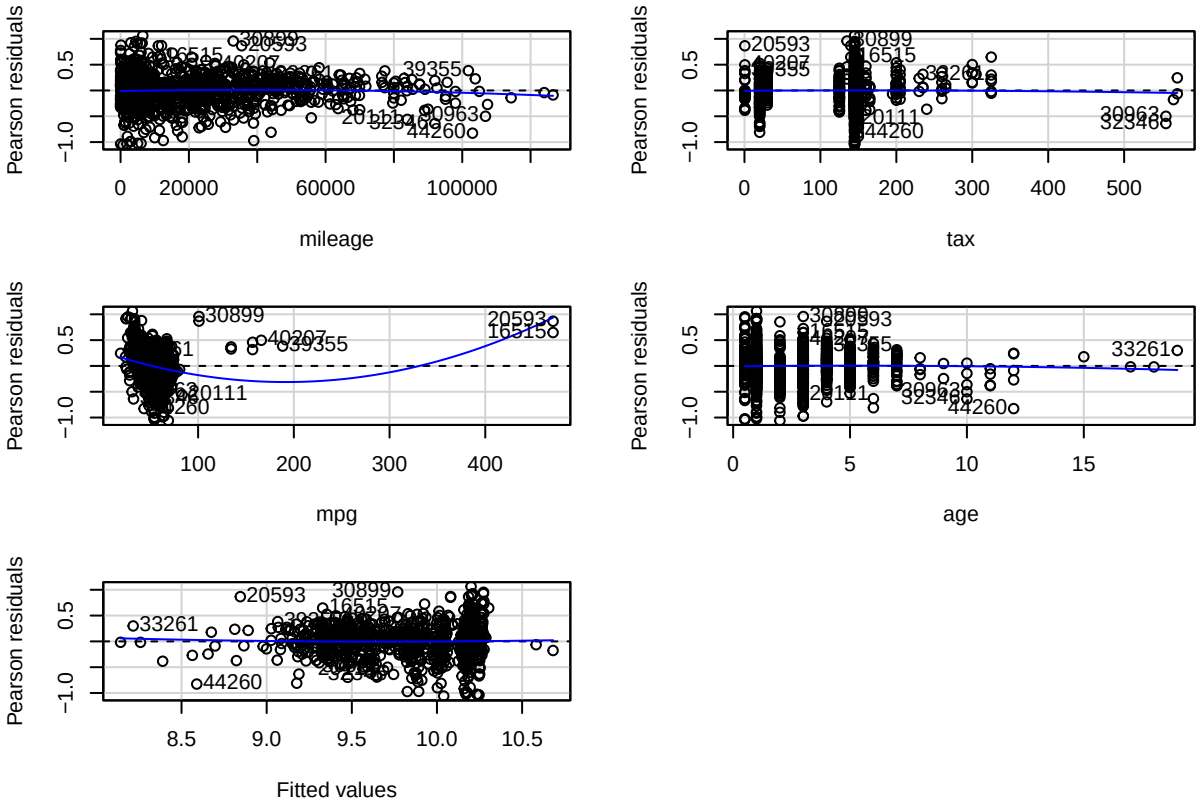
```
##         MLE of lambda Score Statistic (t)  Pr(>|t|)
## mileage       0.13269              1.0375 0.2997598
## tax           0.46073             -3.2720 0.0011062 **
## mpg          -2.25574              8.9625 < 2.2e-16 ***
## age           1.41541             -3.5963 0.0003393 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## iterations =  11
##
## Score test for null hypothesis that all lambdas = 1:
## F = 21.964, df = 4 and 955, Pr(>F) = < 2.2e-16
```

```
# Power transformations of the predictors in a linear model

par(mfrow=c(2,2))
plot(m2,id.n=0)
```

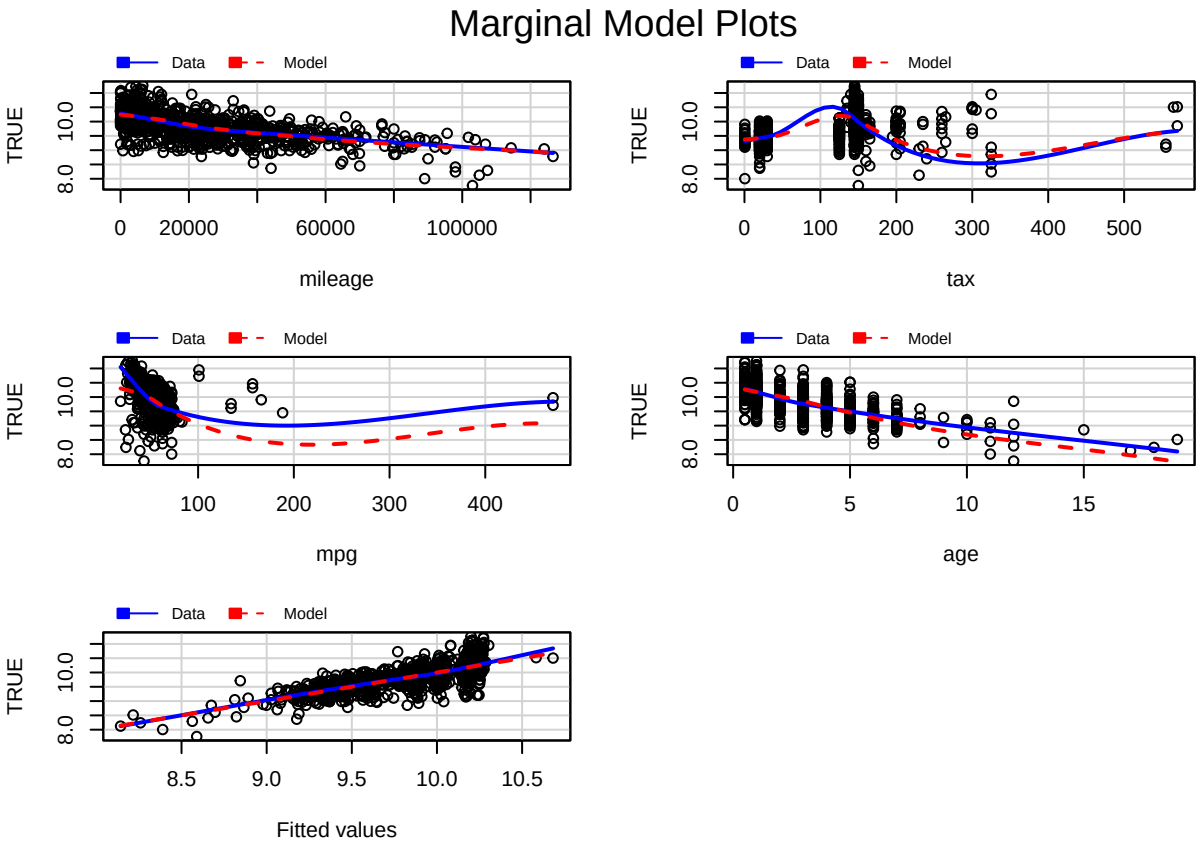



```
par(mfrow=c(1,1))

# Other tools to validate my linear model:
residualPlots(m2,id=list(method=cooks.distance(m2),n=10))
```

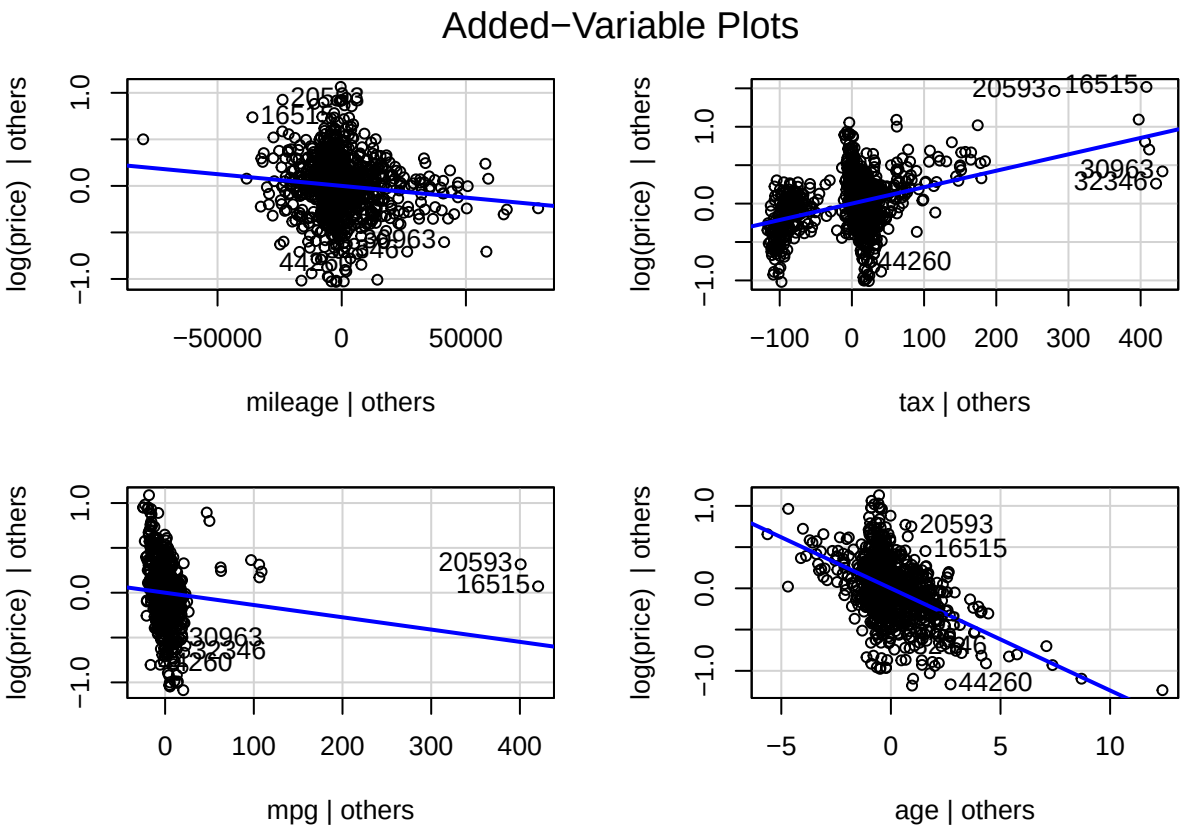


```
##             Test stat Pr(>|Test stat|)
## mileage       -1.1860           0.2359
## tax           -0.4671           0.6405
## mpg            8.6934           <2e-16 ***
## age           -0.6010           0.5480
## Tukey test     0.5282           0.5974
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

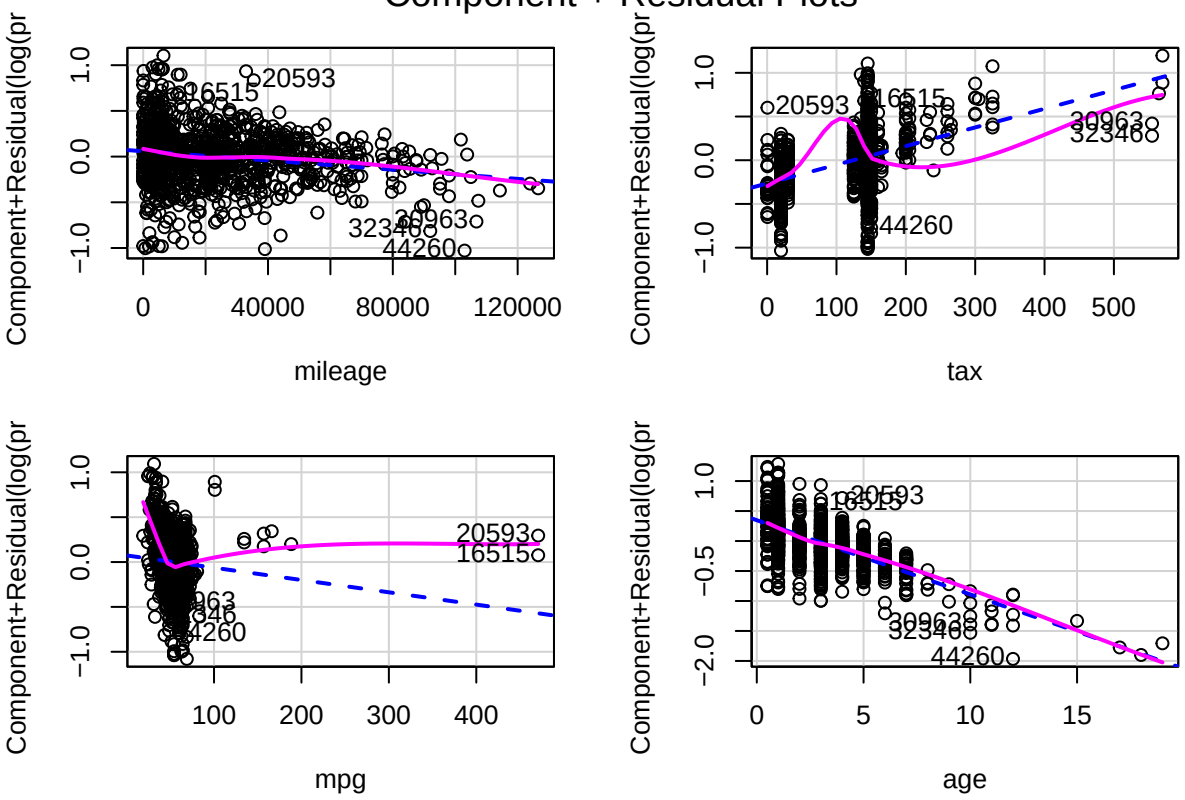
```
marginalModelPlots(m2)
```




```
avPlots(m2,id=list(method=cooks.distance(m2),n=5))
```




```
crPlots(m2,id=list(method=cooks.distance(m2),n=5))
```




```
# Objective: Check linearity of my data (model and fit data) and chech how my model fits to my data

# Suggested by BoxTidwell
m3<-lm(log(price)~log(mileage)+sqrt(tax)+poly(mpg,2)+age,data=df[!df$mout=="YesMOut",])
summary(m3)
```

```
##
## Call:
## lm(formula = log(price) ~ log(mileage) + sqrt(tax) + poly(mpg,
##     2) + age, data = df[!df$mout == "YesMOut", ])
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.93339 -0.17191  0.01775  0.18515  0.74411
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     9.970361   0.070171 142.086   <2e-16 ***
## log(mileage)   -0.011279   0.007013  -1.608    0.108
## sqrt(tax)       0.028511   0.003392   8.405   <2e-16 ***
## poly(mpg, 2)1 -4.017122   0.355400 -11.303   <2e-16 ***
## poly(mpg, 2)2  2.491839   0.277641   8.975   <2e-16 ***
## age           -0.112431   0.005775 -19.468   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2671 on 958 degrees of freedom
## Multiple R-squared:  0.6545, Adjusted R-squared:  0.6527
## F-statistic: 362.9 on 5 and 958 DF,  p-value: < 2.2e-16
```

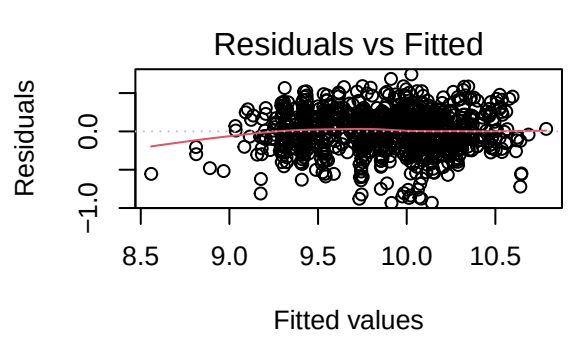
```
# Validation and effects consideration:
Anova(m3) #Net effect test
```

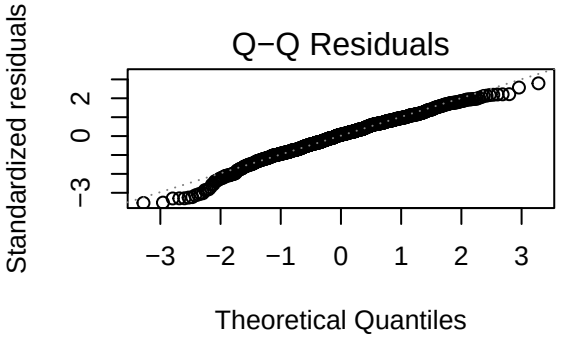
```
## Anova Table (Type II tests)
##
## Response: log(price)
##              Sum Sq  Df  F value Pr(>F)
## log(mileage)  0.184   1   2.5864 0.1081
## sqrt(tax)     5.040   1  70.6516 <2e-16 ***
## poly(mpg, 2) 17.777   2 124.6003 <2e-16 ***
## age          27.036   1 379.0041 <2e-16 ***
## Residuals    68.338 958
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

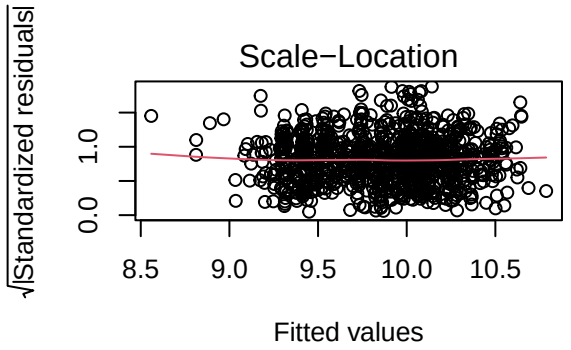
```
vif(m3)
```

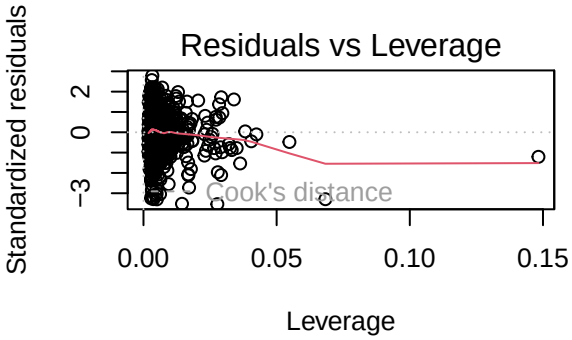
```
##                  GVIF Df GVIF^(1/(2*Df))
## log(mileage) 1.759300  1        1.326386
## sqrt(tax)    1.854292  1        1.361724
## poly(mpg, 2) 1.859921  2        1.167814
## age          1.750466  1        1.323052
```

```
par(mfrow=c(2,2))
plot(m3,id.n=0)
```

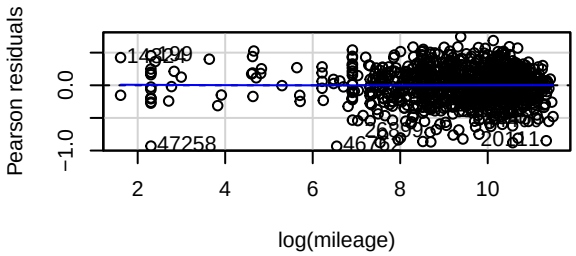


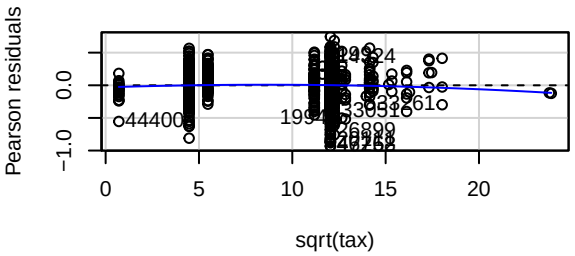


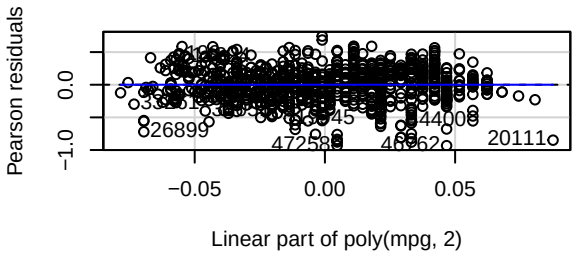


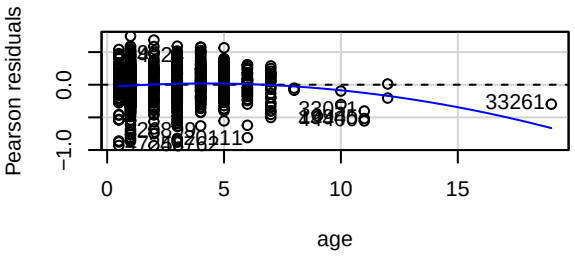


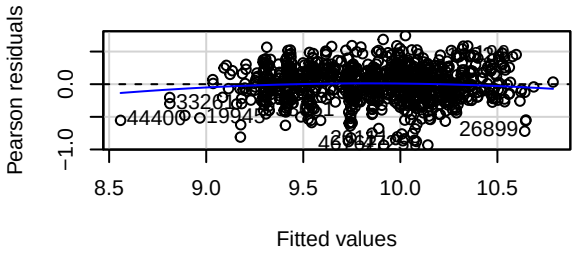

```
par(mfrow=c(1,1))
residualPlots(m3,id=list(method=cooks.distance(m3),n=10))
```







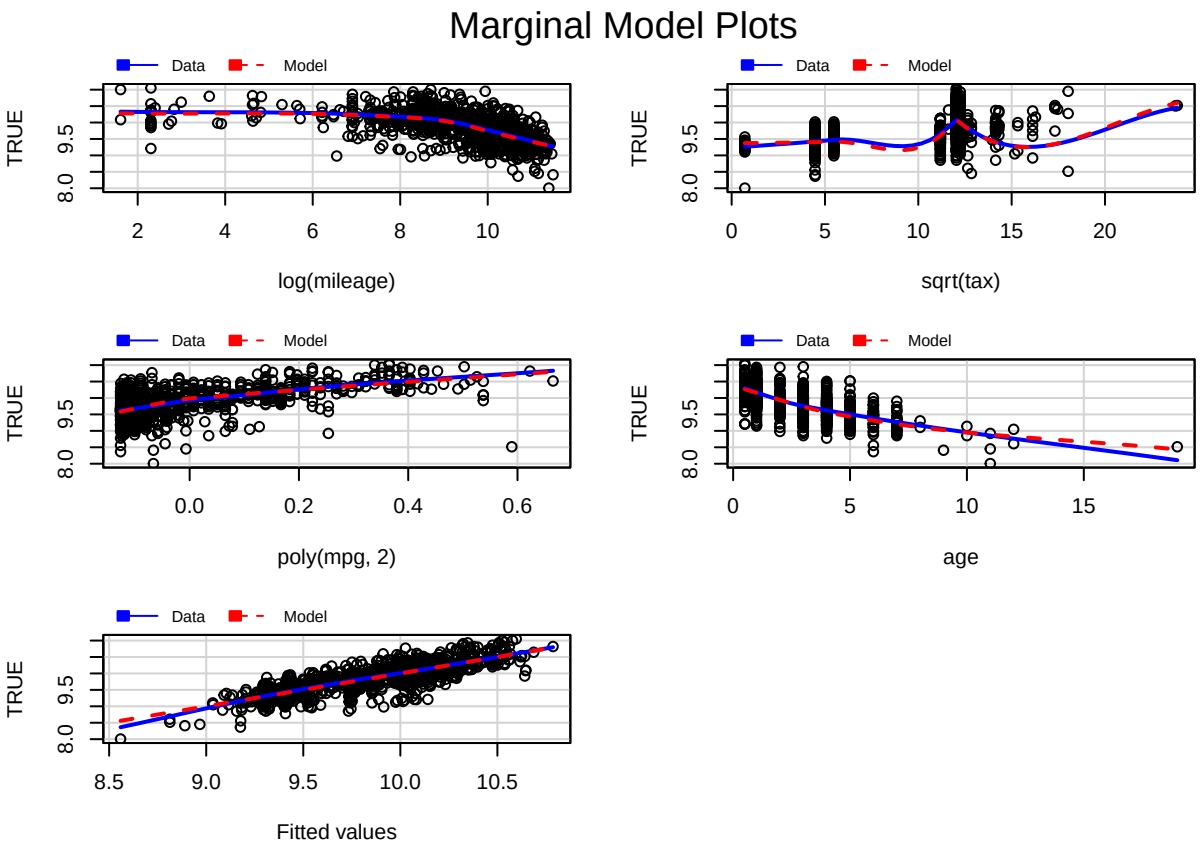




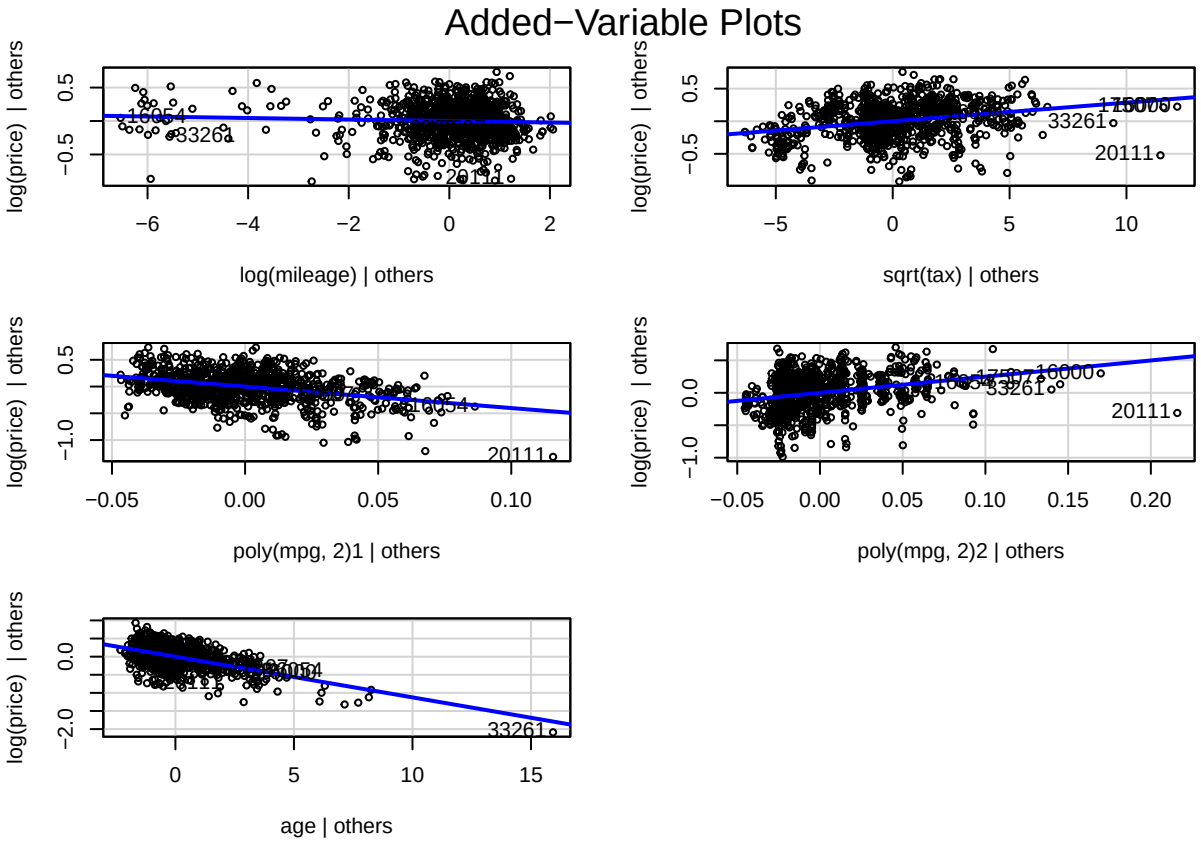

```
##              Test stat Pr(>|Test stat|)
## log(mileage)    0.2238         0.822952
## sqrt(tax)      -1.1074         0.268403
## poly(mpg, 2)
## age            -3.8364         0.000133 ***
## Tukey test     -2.6842         0.007271 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
marginalModelPlots(m3)
```

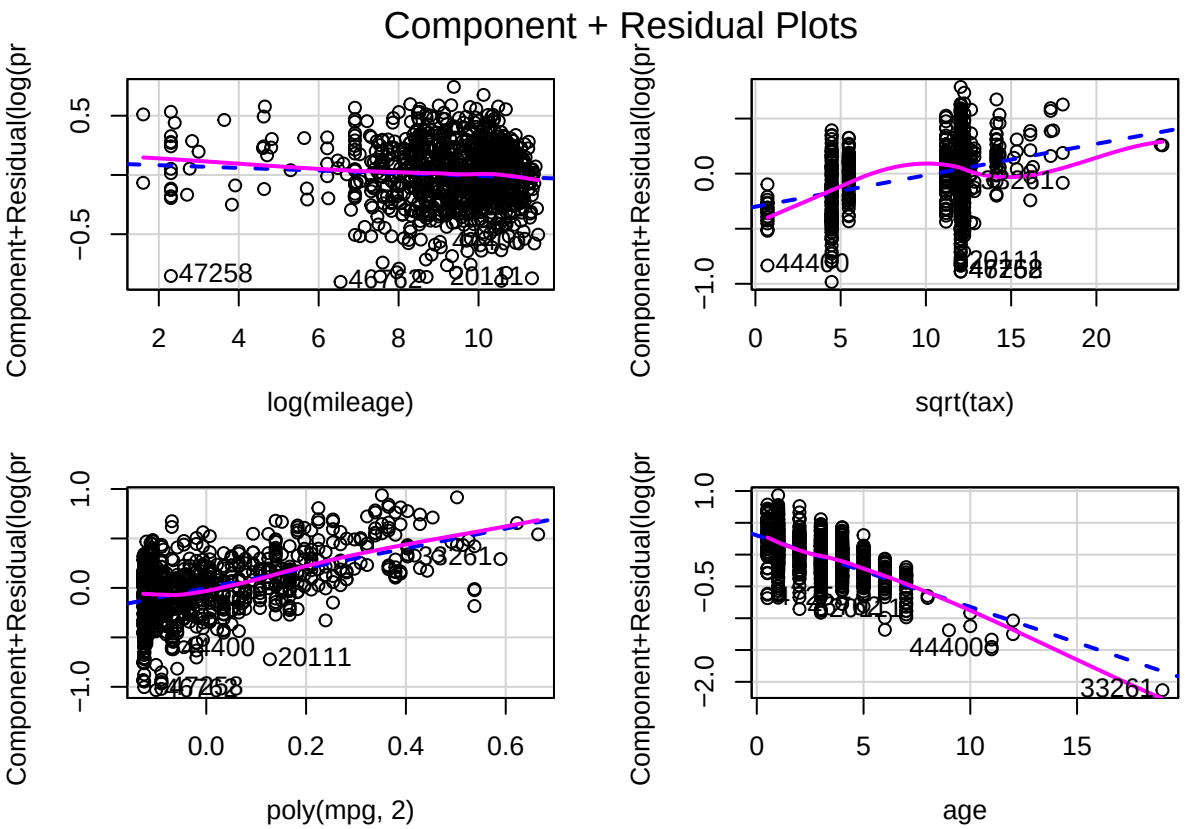
```
## Warning in mmps(...): Splines and/or polynomials replaced by a fitted linear
## combination
```




```
avPlots(m3,id=list(method=hatvalues(m3),n=5))
```



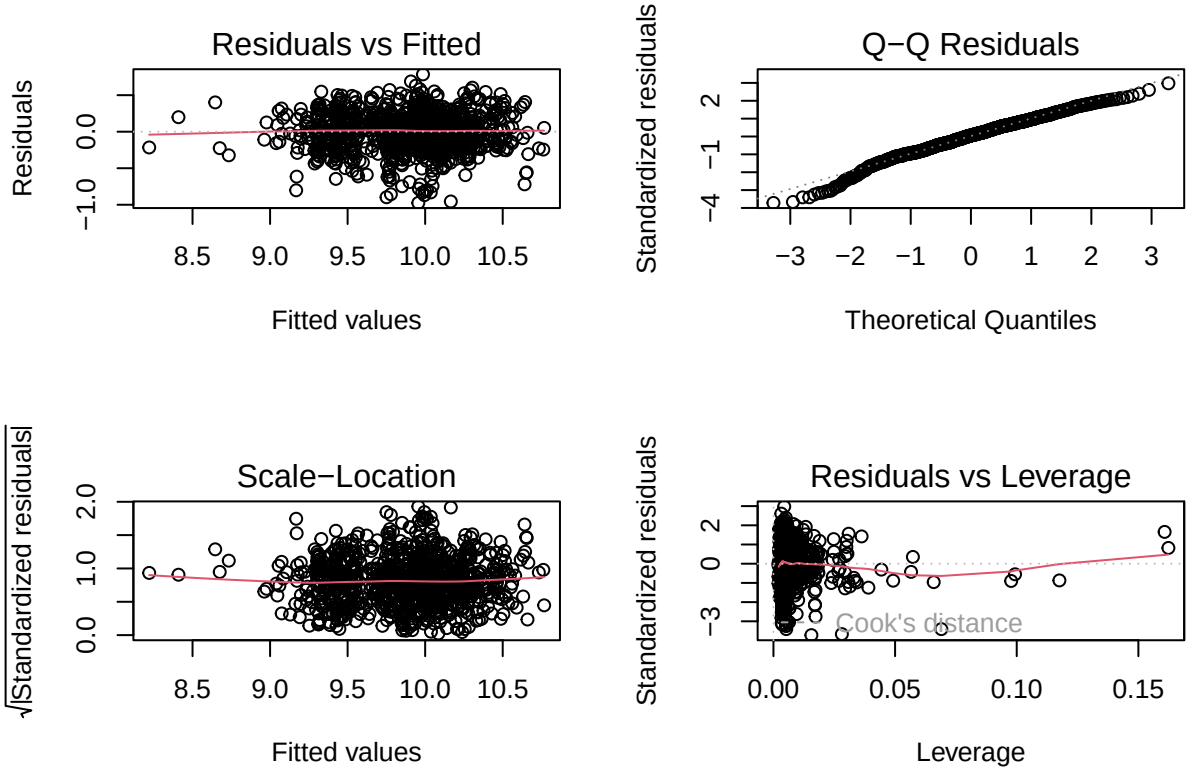

```
crPlots(m3,id=list(method=cooks.distance(m3),n=5))
```



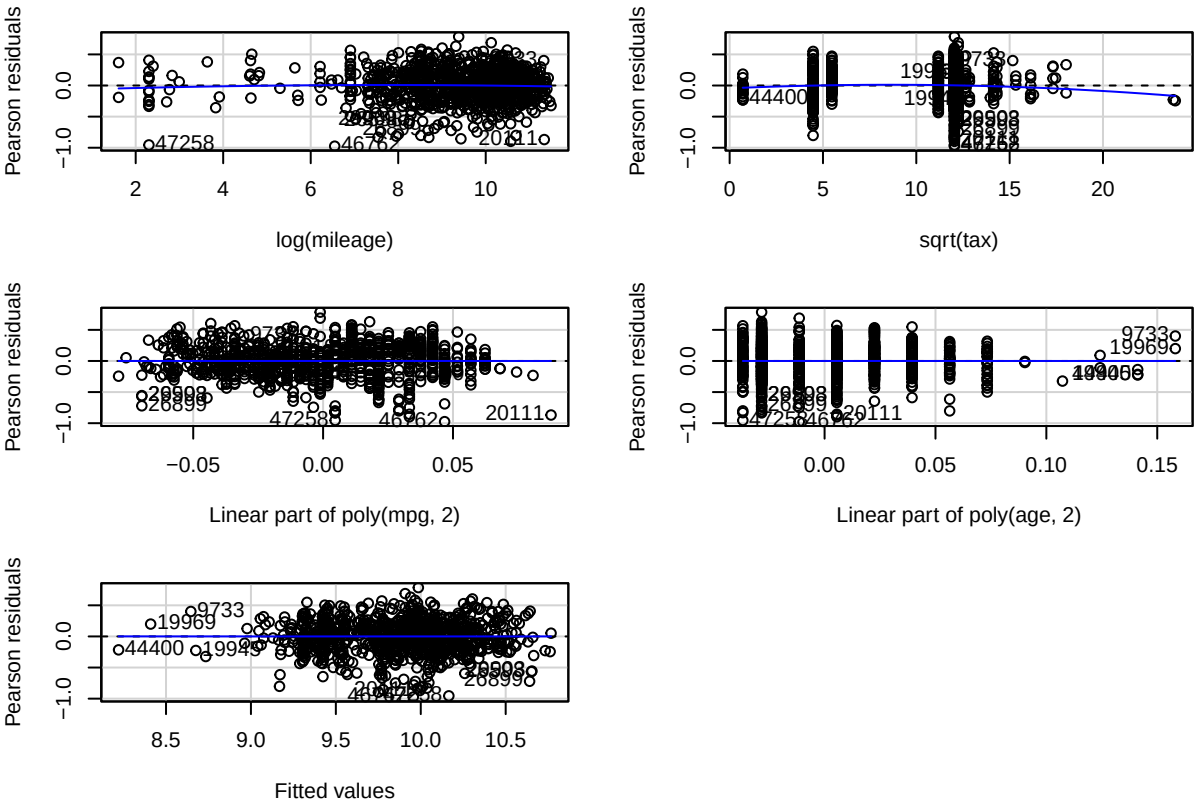

```
#Non linearity of age could be caused by observation 33261:
df2 <- df[!df$mout=="YesMOut",]
df2 <- df2[row.names(df2)!="33261",]


m4<-lm(log(price)~log(mileage)+sqrt(tax)+poly(mpg,2)+poly(age,2),data=df2)
summary(m4)
```

```
##
## Call:
## lm(formula = log(price) ~ log(mileage) + sqrt(tax) + poly(mpg,
##     2) + poly(age, 2), data = df2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.97502 -0.16879  0.01518  0.18209  0.78473
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     9.759964   0.077254 126.337  < 2e-16 ***
## log(mileage)   -0.024655   0.007500  -3.287  0.00105 **
## sqrt(tax)       0.031730   0.003435   9.238  < 2e-16 ***
## poly(mpg, 2)1 -4.201080   0.354069 -11.865  < 2e-16 ***
## poly(mpg, 2)2  2.598306   0.276577   9.395  < 2e-16 ***
## poly(age, 2)1 -5.905876   0.380619 -15.516  < 2e-16 ***
## poly(age, 2)2 -1.376830   0.300255  -4.586 5.13e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2643 on 956 degrees of freedom
## Multiple R-squared:  0.6593, Adjusted R-squared:  0.6571
## F-statistic: 308.3 on 6 and 956 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(m4,id.n=0)
```



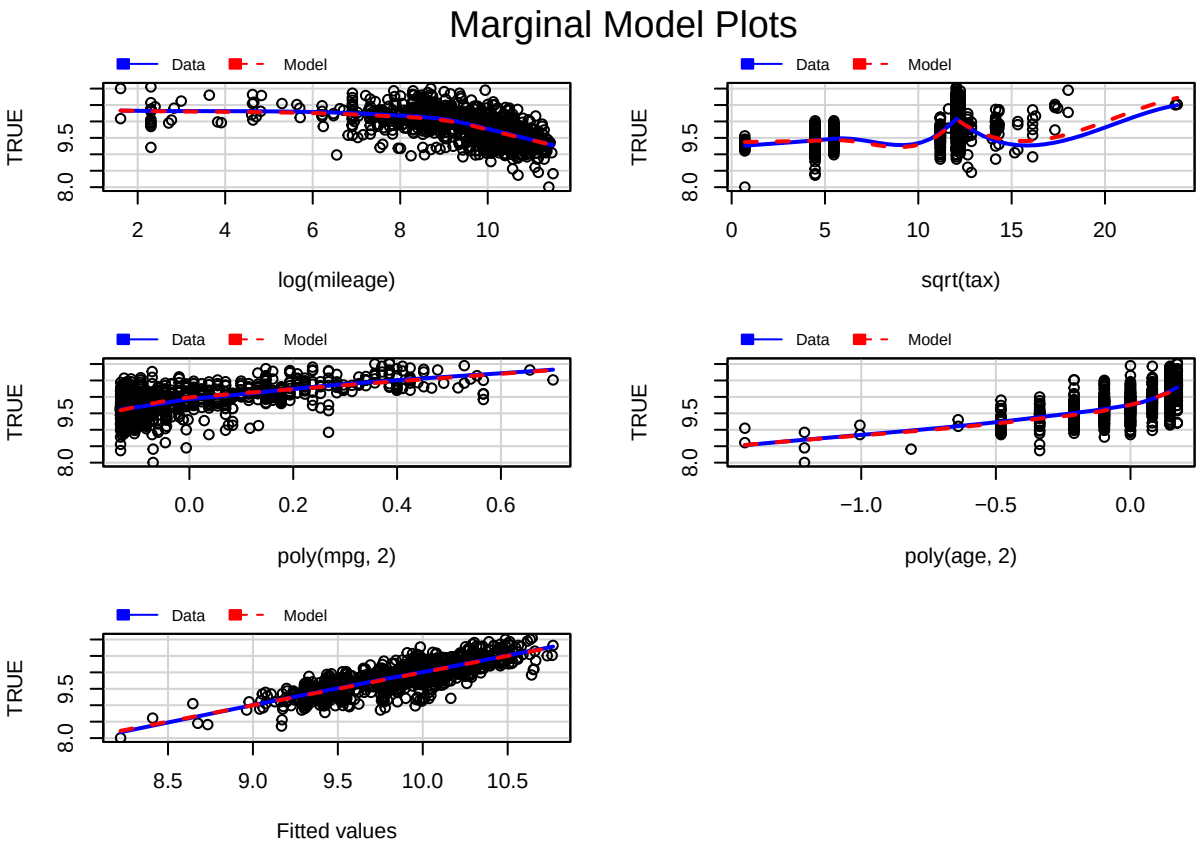

```
par(mfrow=c(1,1))
residualPlots(m4,id=list(method=cooks.distance(m4),n=10))
```



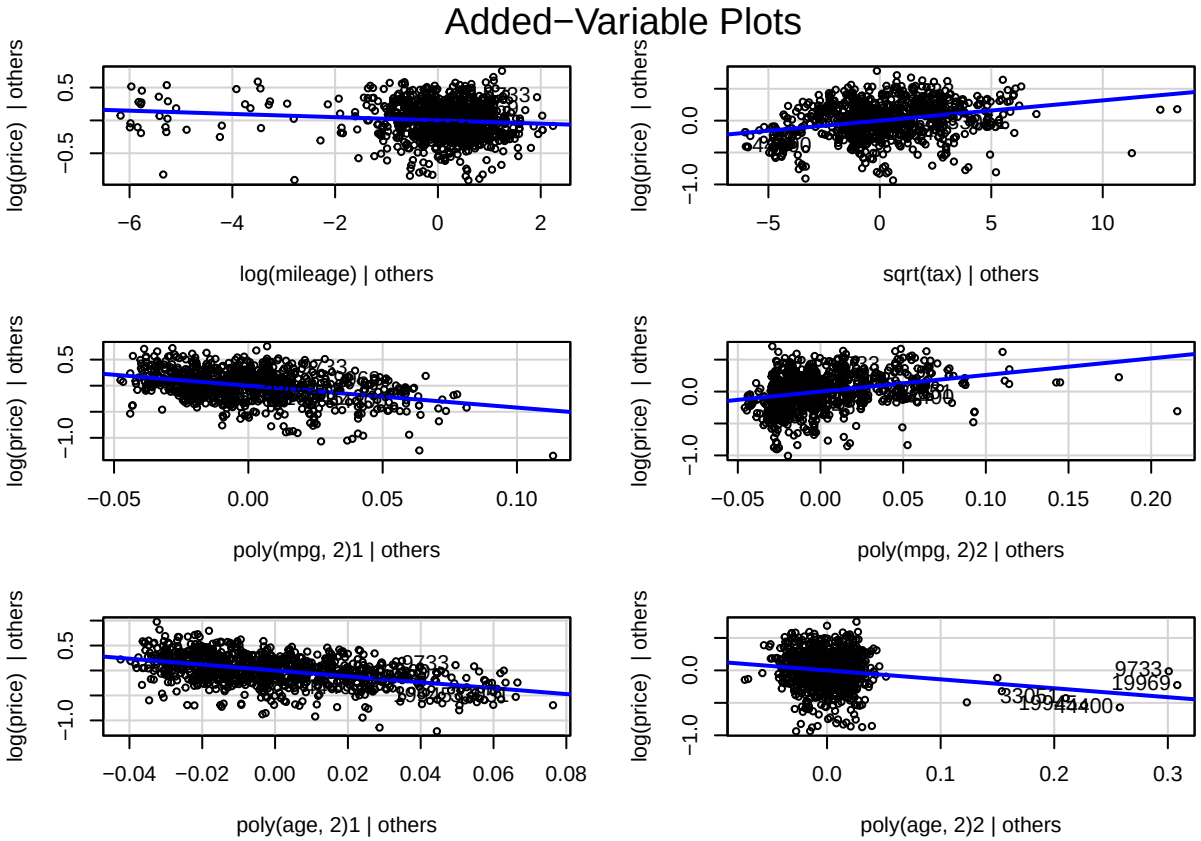

```
##              Test stat Pr(>|Test stat|)
## log(mileage)   -1.2083           0.2272
## sqrt(tax)      -1.5775           0.1150
## poly(mpg, 2)
## poly(age, 2)
## Tukey test      0.0156           0.9875
```

```
marginalModelPlots(m4)
```

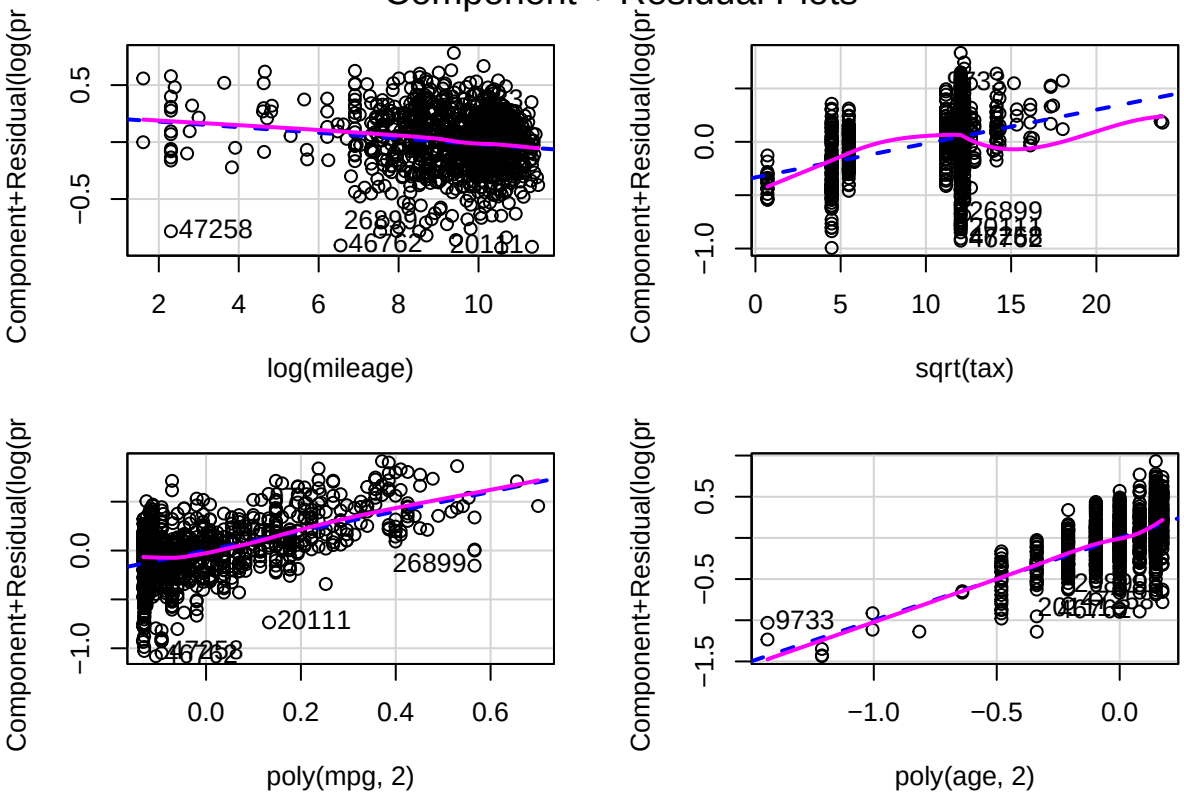
```
## Warning in mmps(...): Splines and/or polynomials replaced by a fitted linear
## combination
```




```
avPlots(m4,id=list(method=hatvalues(m4),n=5))
```




```
crPlots(m4,id=list(method=cooks.distance(m4),n=5))
```




```
m5<-lm(log(price)~log(mileage)+sqrt(tax)+poly(mpg,2)+poly(age,2),data=df2)
summary(m5)
```

```
##
## Call:
## lm(formula = log(price) ~ log(mileage) + sqrt(tax) + poly(mpg,
##     2) + poly(age, 2), data = df2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.97502 -0.16879  0.01518  0.18209  0.78473
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     9.759964   0.077254 126.337  < 2e-16 ***
## log(mileage)   -0.024655   0.007500  -3.287  0.00105 **
## sqrt(tax)       0.031730   0.003435   9.238  < 2e-16 ***
## poly(mpg, 2)1 -4.201080   0.354069 -11.865  < 2e-16 ***
## poly(mpg, 2)2  2.598306   0.276577   9.395  < 2e-16 ***
## poly(age, 2)1 -5.905876   0.380619 -15.516  < 2e-16 ***
## poly(age, 2)2 -1.376830   0.300255  -4.586 5.13e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2643 on 956 degrees of freedom
## Multiple R-squared:  0.6593, Adjusted R-squared:  0.6571
## F-statistic: 308.3 on 6 and 956 DF,  p-value: < 2.2e-16
```

```
library(lmtest)
```

```
## S'està carregant el paquet requerit: zoo
```

```
##
## S'està adjuntant el paquet: 'zoo'
```

```
## Els següents objectes estan emmascarats des de 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
bptest(m5) #Homoskedasticity
```

```
##
##  studentized Breusch-Pagan test
##
## data:  m5
## BP = 31.991, df = 6, p-value = 1.638e-05
```

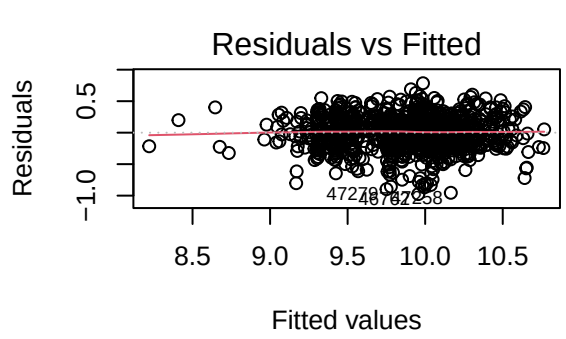
```r
anova(m4, m5) #Does the variable age squared have to be included in my model
```

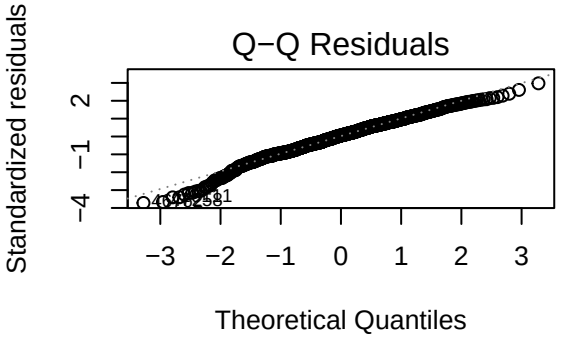
```
## Analysis of Variance Table
##
## Model 1: log(price) ~ log(mileage) + sqrt(tax) + poly(mpg, 2) + poly(age,
##     2)
## Model 2: log(price) ~ log(mileage) + sqrt(tax) + poly(mpg, 2) + poly(age,
##     2)
##   Res.Df    RSS Df Sum of Sq F Pr(>F)
## 1    956 66.766
## 2    956 66.766  0         0
```

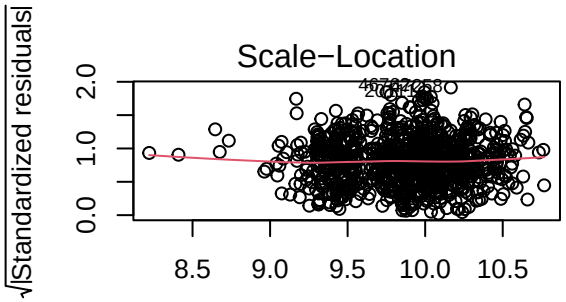
```r
summary(m5)
```

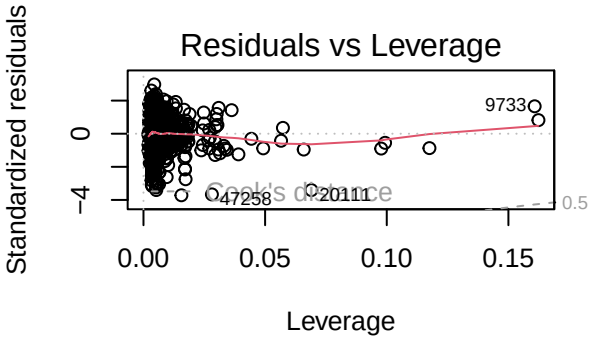
```
##
## Call:
## lm(formula = log(price) ~ log(mileage) + sqrt(tax) + poly(mpg,
##     2) + poly(age, 2), data = df2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.97502 -0.16879  0.01518  0.18209  0.78473
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    9.759964   0.077254 126.337  < 2e-16 ***
## log(mileage)  -0.024655   0.007500  -3.287  0.00105 **
## sqrt(tax)      0.031730   0.003435   9.238  < 2e-16 ***
## poly(mpg, 2)1 -4.201080   0.354069 -11.865  < 2e-16 ***
## poly(mpg, 2)2  2.598306   0.276577   9.395  < 2e-16 ***
## poly(age, 2)1 -5.905876   0.380619 -15.516  < 2e-16 ***
## poly(age, 2)2 -1.376830   0.300255  -4.586 5.13e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2643 on 956 degrees of freedom
## Multiple R-squared:  0.6593, Adjusted R-squared:  0.6571
## F-statistic: 308.3 on 6 and 956 DF,  p-value: < 2.2e-16
```

```r
par(mfrow=c(2,2))
plot(m5)
```










```
m4 <- step( m5, k=log(nrow(df)))
```

```
## Start:  AIC=-2521.76
## log(price) ~ log(mileage) + sqrt(tax) + poly(mpg, 2) + poly(age,
##     2)
##
##                Df Sum of Sq    RSS     AIC
## <none>                      66.766 -2521.8
## - log(mileage)  1    0.7547 67.520 -2517.8
## - sqrt(tax)     1    5.9599 72.726 -2446.3
## - poly(mpg, 2)  2   18.9194 85.685 -2295.3
## - poly(age, 2)  2   24.1129 90.879 -2238.7
```

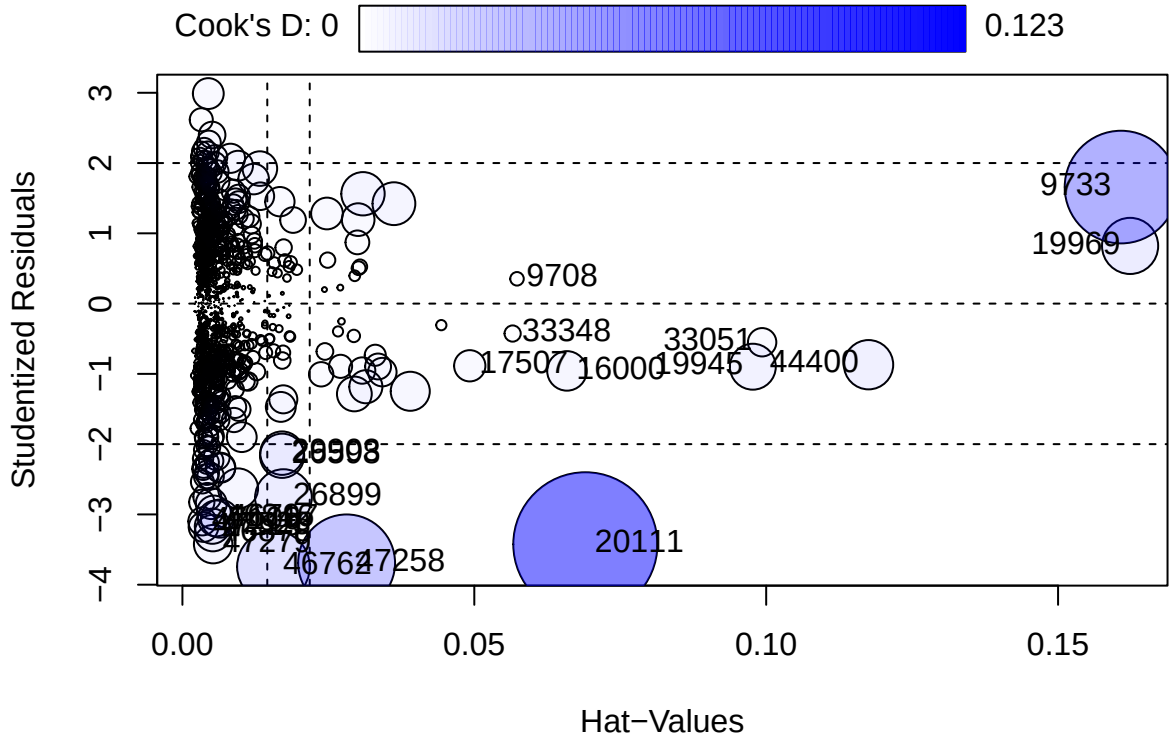
```
llres <- which(abs(rstudent(m4))>4);length(llres)
```

```
## [1] 0
```

```
df[llres,]
```

```
##  [1] model        year         price        transmission mileage
##  [6] fuelType     tax          mpg          engineSize   manufacturer
## [11] age          aux_price    aux_mileage  aux_mpg      aux_tax
## [16] aux_age      Audi         mout
## <0 files> (o «row.names» de longitud 0)
```

```
par(mfrow=c(1,1))
influencePlot(m5, id=list(n=10))
```




```
##          StudRes         Hat       CookD
## 9708   0.3529672 0.057314963 0.001083103
## 9733   1.6579262 0.160786129 0.075095616
## 16000 -0.9577222 0.065869454 0.009240493
## 17507 -0.8850064 0.049219957 0.005793680
## 19945 -0.8980776 0.097712134 0.012480183
## 19969  0.8206597 0.162340476 0.018652447
## 20111 -3.4254539 0.069022383 0.122896623
## 26598 -2.1654359 0.017072300 0.011590186
## 26899 -2.7604927 0.017306529 0.019040131
## 29903 -2.1279707 0.017059012 0.011185613
## 33051 -0.5501960 0.099307054 0.004771523
## 33348 -0.4259872 0.056564935 0.001555616
## 44400 -0.8696283 0.117536955 0.014393220
## 46762 -3.7438451 0.015596414 0.031297962
## 46797 -3.0594208 0.006425350 0.008572260
## 46846 -3.0565697 0.005742528 0.007641900
## 46878 -3.2654477 0.005216880 0.007908645
## 47106 -3.0992775 0.003411843 0.004655907
## 47219 -3.1799516 0.003614597 0.005191056
## 47258 -3.6900067 0.028126256 0.055560292
## 47279 -3.4229888 0.005262155 0.008756419
## 47523 -3.1673855 0.005140402 0.007335929
```

```
cooks.distance(m5) #What would be threshold for this metric?
```

```
##           64          198          199          298          396          413
## 8.494708e-05 3.712750e-04 1.112115e-02 8.503426e-05 1.460658e-04 1.362234e-04
##          447          546          557          618          626          732
## 5.091417e-05 1.483769e-03 1.349794e-04 9.043368e-04 4.392994e-04 3.689870e-06
##          801          925         1026         1142         1242         1267
## 1.101496e-03 2.901728e-04 2.824663e-05 2.823527e-06 1.806866e-04 1.237527e-03
##         1367         1514         1566         1568         1612         1625
## 9.365012e-05 2.158747e-07 1.069089e-04 5.982401e-05 6.142492e-04 1.856194e-06
##         1630         1651         1799         1889         1966         1981
## 6.257533e-04 5.218156e-04 1.499346e-05 7.442799e-04 1.866068e-05 1.924967e-04
##         2029         2112         2116         2136         2165         2201
## 3.968428e-03 2.508246e-05 1.053441e-03 3.231368e-05 4.963399e-05 6.809822e-06
##         2245         2273         2352         2453         2577         2602
## 4.021830e-07 1.656018e-03 1.069169e-04 4.874163e-05 1.674116e-04 6.941223e-05
##         2618         2624         2808         2867         2901         2918
```

```
## 1.794358e-05 9.210610e-04 1.905764e-03 2.149884e-04 6.212455e-06 2.645618e-04
##         3011         3032         3048         3103         3168         3237
## 9.224573e-04 1.071105e-04 1.181195e-04 2.040927e-04 3.981610e-04 1.904882e-05
##         3281         3328         3393         3417         3459         3467
## 2.324734e-04 5.431336e-04 4.196372e-05 1.020241e-03 5.437719e-05 7.161485e-03
##         3506         3513         3534         3565         3573         3631
## 6.054924e-05 3.790664e-05 2.153521e-05 1.188660e-04 2.192467e-04 3.298033e-04
##         3677         3687         3834         3884         3913         3936
## 6.885911e-05 1.196200e-03 4.786383e-05 1.103151e-04 5.007473e-04 1.533118e-04
##         3963         4215         4419         4514         4533         4578
## 1.189847e-04 1.151111e-04 9.095060e-05 1.640650e-03 1.026911e-03 9.853776e-04
##         4585         4598         4668         4850         4851         4861
## 6.522965e-05 1.767081e-04 1.490077e-03 2.334508e-04 2.958673e-05 5.998210e-04
##         4918         4934         4948         5011         5012         5032
## 2.858757e-04 9.368043e-06 5.273798e-03 1.647789e-03 6.461259e-05 1.381826e-03
##         5192         5209         5289         5307         5349         5390
## 5.617924e-04 2.278565e-06 3.131870e-03 2.288545e-05 4.709306e-04 4.483034e-05
##         5461         5468         5536         5644         5745         5777
## 6.783883e-04 2.863689e-03 4.535243e-04 1.469702e-04 1.988535e-04 1.361986e-06
##         5781         5873         5887         5945         5988         6029
## 6.331322e-05 5.656791e-05 1.629854e-03 1.066495e-05 3.295870e-04 7.707991e-05
##         6068         6073         6112         6113         6144         6151
## 2.743986e-04 3.985013e-05 3.216798e-03 1.475472e-04 7.104017e-05 4.908974e-05
##         6202         6206         6235         6287         6309         6361
## 6.986423e-05 1.450426e-05 4.783899e-06 3.469897e-04 3.243901e-06 1.323684e-05
##         6363         6391         6404         6434         6515         6517
## 5.299620e-06 4.340745e-04 4.259107e-06 2.230605e-05 5.906294e-04 3.862574e-05
##         6601         6617         6653         6797         6840         6919
## 6.950742e-04 6.017006e-04 1.437445e-03 4.421283e-04 1.621312e-04 5.840653e-04
##         6959         7039         7043         7059         7076         7184
## 1.066546e-07 7.675810e-06 9.152644e-04 1.218977e-03 1.724216e-06 4.958355e-04
##         7277         7282         7289         7370         7380         7491
## 2.201304e-03 1.107267e-04 5.957431e-03 7.641962e-05 1.618298e-05 2.046138e-04
##         7548         7550         7608         7634         7650         7664
## 3.201766e-05 1.138039e-04 8.819621e-04 1.623143e-04 1.324203e-04 1.043766e-05
##         7692         7708         7738         7897         8020         8066
## 2.647500e-04 7.661513e-04 1.744392e-05 1.196437e-04 2.118241e-05 8.003177e-05
##         8068         8083         8159         8166         8321         8341
## 3.503259e-04 2.582138e-04 8.099992e-04 1.000306e-03 3.533791e-07 5.309601e-04
##         8761         8773         8787         8788         8805         8838
## 2.217246e-04 9.114550e-05 1.151153e-04 2.283587e-04 3.198664e-04 3.793326e-04
##         8872         8934         8996         9053         9091         9117
## 3.267042e-04 9.403324e-05 7.588600e-04 3.354040e-04 6.001108e-04 3.099917e-06
##         9192         9197         9203         9252         9270         9305
## 4.160508e-05 3.706459e-04 1.096314e-04 4.446438e-08 4.605399e-07 5.778908e-04
##         9313         9316         9404         9405         9438         9469
## 1.102744e-03 1.237015e-03 2.227233e-05 9.343520e-04 7.644272e-05 1.128484e-03
##         9548         9557         9708         9733         9933        10071
## 4.758244e-04 1.273633e-03 1.083103e-03 7.509562e-02 1.415230e-03 1.896494e-03
##        10116        10141        10240        10336        10353        10410
## 1.841518e-04 1.693995e-04 5.148820e-06 6.666333e-04 1.582119e-06 1.410756e-04
##        10491        10498        10503        10534        10545        10547
## 3.385531e-04 1.836539e-05 1.735534e-09 6.730216e-06 6.568116e-05 5.095426e-03
##        10601        10617        10626        10639        10643        10646
## 4.622412e-04 1.006257e-03 4.046122e-08 5.015414e-06 8.408224e-05 5.706333e-04
##        10725        10732        10814        10848        10863        10909
## 1.307963e-03 5.187232e-04 1.079607e-05 4.047522e-07 6.958601e-04 2.206558e-04
##        11017        11065        11110        11169        11302        11356
## 1.451350e-04 2.067085e-04 1.902259e-03 2.003725e-03 7.746240e-04 1.018000e-07
##        11365        11474        11476        11519        11650        11702
## 1.613180e-04 7.071454e-03 3.218272e-04 5.625173e-04 8.512257e-04 3.332966e-04
##        11769        11799        11909        11942        11947        11954
## 9.872450e-06 1.099399e-04 5.773122e-04 6.015951e-04 2.323803e-05 2.957799e-04
##        11976        12118        12213        12245        12246        12263
```

```
## 4.686963e-05 4.023644e-04 3.586526e-05 5.074579e-04 1.677848e-05 1.634930e-04
##        12304        12407        12528        12582        12830        12862
## 3.659318e-06 1.078252e-04 4.054239e-04 2.586643e-04 3.389556e-03 2.110132e-06
##        12937        12982        13055        13081        13095        13156
## 3.302882e-05 8.361808e-04 1.284530e-04 2.257454e-06 9.579663e-05 4.594929e-04
##        13221        13239        13252        13292        13361        13381
## 2.206390e-05 1.010211e-05 9.728572e-04 1.029399e-03 1.916446e-03 7.107248e-06
##        13454        13654        13677        13743        13788        13804
## 6.328697e-03 1.105219e-03 2.002972e-04 3.462658e-04 6.812875e-04 1.969252e-03
##        13935        13957        13990        14047        14136        14224
## 3.568449e-04 4.506785e-03 7.223887e-04 7.757794e-04 2.978790e-04 4.582592e-04
##        14230        14303        14324        14363        14439        14500
## 3.121377e-05 2.848492e-04 1.084457e-02 4.894957e-04 1.033346e-03 1.846984e-04
##        14890        14945        14947        15018        15036        15042
## 5.220402e-04 5.251345e-05 5.746268e-05 5.048083e-03 4.184625e-03 1.500375e-03
##        15086        15157        15174        15224        15234        15273
## 1.142573e-03 1.336359e-04 4.949239e-05 2.136510e-06 3.515748e-06 6.780195e-04
##        15284        15287        15298        15397        15411        15475
## 1.331692e-04 1.273534e-06 1.927389e-03 1.036865e-03 8.594376e-06 1.327566e-03
##        15545        15558        15611        15713        15957        16000
## 3.525525e-03 3.894306e-04 9.731958e-04 5.106097e-04 1.027798e-03 9.240493e-03
##        16008        16025        16054        16063        16089        16137
## 1.260307e-03 4.115624e-03 6.242955e-04 9.303989e-04 4.071687e-05 4.781257e-05
##        16198        16213        16254        16288        16317        16323
## 9.401175e-04 5.009436e-04 1.747219e-04 1.821821e-03 1.362680e-04 1.107506e-03
##        16456        16508        16590        16599        16637        16842
## 2.136996e-04 1.024567e-03 3.751308e-04 2.144106e-03 4.040308e-04 7.073344e-05
##        16857        16999        17002        17054        17072        17136
## 1.094931e-03 4.614636e-05 3.352328e-03 2.359043e-04 3.456697e-04 1.604105e-03
##        17167        17174        17234        17383        17396        17507
## 3.501095e-04 9.973217e-05 1.046365e-03 1.111953e-03 5.327738e-05 5.793680e-03
##        17527        17533        17601        17695        17710        17903
## 2.682831e-04 1.899530e-05 7.682016e-04 6.211476e-05 1.306979e-04 6.138160e-04
##        17938        18032        18115        18184        18190        18257
## 5.280449e-05 2.153367e-05 9.037367e-05 4.442723e-04 2.693543e-04 7.930623e-04
##        18263        18499        18547        18562        18574        18625
## 8.589576e-05 2.526882e-04 1.011045e-03 1.344933e-05 1.651077e-05 1.923719e-05
##        18658        18659        18683        18755        18782        18841
## 2.874772e-06 3.987330e-04 1.530275e-03 7.015291e-04 1.159373e-04 2.128344e-05
##        18898        18954        18956        19015        19037        19050
## 1.553111e-05 2.722764e-04 2.982852e-04 3.154147e-04 1.984640e-05 3.505091e-04
##        19161        19205        19217        19231        19624        19720
## 3.493708e-04 2.240247e-05 1.315869e-03 5.648828e-06 3.118258e-05 9.877814e-04
##        19724        19945        19969        19974        20076        20109
## 8.972322e-06 1.248018e-02 1.865245e-02 4.985050e-04 1.659753e-03 9.252874e-05
##        20111        20114        20292        20377        20424        20475
## 1.228966e-01 3.991253e-05 4.744248e-04 1.842047e-04 1.923740e-05 6.972426e-05
##        20504        20627        20718        20791        20874        20961
## 3.484869e-03 1.130604e-03 5.189891e-05 1.814575e-04 3.507061e-05 8.130291e-04
##        20979        21088        21123        21195        21233        21266
## 2.125889e-06 3.125062e-03 6.381527e-04 5.850146e-04 5.191017e-06 2.537943e-04
##        21292        21346        21362        21614        21654        21657
## 1.536993e-04 3.790969e-04 6.395094e-04 3.716145e-04 1.156498e-03 1.640789e-04
##        21741        21750        21753        21788        21842        21846
## 1.395899e-03 2.111791e-05 3.484018e-03 2.428824e-04 1.830973e-04 4.550837e-05
##        21882        21891        21909        21959        22090        22113
## 4.396328e-04 1.331418e-03 1.171102e-05 5.074624e-04 3.901167e-03 4.714494e-04
##        22200        22232        22249        22306        22312        22369
## 8.160681e-04 3.404699e-04 6.892317e-04 2.191923e-03 5.529071e-03 4.083437e-04
##        22410        22431        22444        22494        22768        22818
## 7.917555e-04 6.866261e-04 2.405927e-03 2.476735e-03 2.701966e-05 9.706120e-07
##        22897        22944        22961        23018        23244        23357
## 1.162931e-06 1.374527e-03 1.084532e-04 3.584074e-04 3.421573e-04 2.061144e-03
##        23432        23612        23695        23698        23802        23861
```

```
## 5.869899e-05 1.425239e-04 4.576689e-04 1.010101e-06 6.288200e-04 7.810403e-04
##        23866        23901        23937        23990        24002        24067
## 1.491151e-03 1.065968e-03 2.871510e-03 1.444355e-03 1.359981e-04 1.010476e-04
##        24085        24104        24120        24138        24165        24218
## 6.956932e-05 3.211129e-03 2.054330e-06 2.429685e-03 2.835775e-04 1.201120e-03
##        24273        24344        24351        24473        24624        24658
## 1.123983e-03 2.591886e-05 2.558002e-04 1.525069e-03 1.176886e-06 1.520416e-03
##        24758        24762        24839        24851        24887        24922
## 4.430089e-04 7.740949e-05 3.870594e-04 8.158282e-04 4.502166e-04 8.787675e-05
##        24948        24954        24965        25085        25172        25174
## 1.849486e-03 4.286255e-04 7.602865e-06 5.762988e-06 1.769782e-03 6.582744e-04
##        25230        25239        25282        25292        25295        25304
## 3.340201e-05 1.270733e-05 2.372793e-03 2.511065e-04 9.576098e-04 1.043951e-03
##        25320        25322        25368        25437        25457        25474
## 4.992570e-04 5.295242e-05 5.578010e-05 1.864964e-03 1.308252e-03 6.266993e-04
##        25476        25492        25504        25517        25534        25609
## 1.109343e-03 2.543690e-06 2.922707e-04 1.952887e-04 7.086398e-05 2.783362e-04
##        25692        25775        25866        25889        25962        26026
## 1.263635e-04 1.496740e-03 3.957832e-04 8.457755e-04 7.495970e-09 2.696973e-04
##        26072        26090        26126        26144        26195        26211
## 6.845231e-04 9.819847e-05 6.134229e-04 9.005573e-04 6.464907e-06 5.535254e-04
##        26296        26303        26327        26362        26412        26447
## 1.798997e-03 1.379826e-03 1.008082e-05 1.780362e-04 3.585064e-04 2.250878e-03
##        26457        26486        26562        26598        26643        26665
## 9.785355e-04 2.148470e-05 1.898653e-04 1.159019e-02 1.421026e-04 4.124311e-04
##        26696        26747        26760        26777        26850        26893
## 7.081638e-04 9.906031e-04 7.129374e-06 1.830680e-03 8.448202e-04 4.075070e-04
##        26899        26929        26978        27009        27068        27098
## 1.904013e-02 1.159038e-03 4.935750e-04 1.410042e-03 2.349504e-03 3.921949e-04
##        27110        27253        27296        27364        27394        27466
## 1.904029e-04 1.285664e-04 1.855744e-03 5.965582e-04 1.322249e-03 1.008820e-03
##        27616        27641        27724        27990        28003        28175
## 6.948094e-04 1.087371e-03 2.521992e-03 2.872112e-03 6.999104e-05 3.023186e-03
##        28176        28206        28217        28259        28352        28403
## 5.561344e-05 5.831743e-05 2.714764e-04 4.033777e-04 1.550744e-04 2.598665e-03
##        28414        28467        28561        28608        28645        28677
## 2.146278e-09 4.944105e-05 1.585745e-04 7.950032e-04 1.687897e-05 5.812388e-05
##        28795        28941        28978        28982        28998        29036
## 9.940703e-05 5.688246e-05 7.391006e-04 4.896116e-08 5.056719e-06 3.960317e-08
##        29143        29152        29382        29412        29471        29524
## 2.947378e-05 6.731642e-05 1.913657e-04 1.702050e-04 3.897804e-04 2.718125e-04
##        29538        29567        29721        29751        29829        29857
## 5.651978e-06 4.679081e-04 9.996645e-05 5.633508e-03 2.852144e-04 9.227225e-05
##        29901        29903        30000        30022        30084        30094
## 5.262693e-04 1.118561e-02 9.381409e-06 1.188732e-04 2.433900e-04 5.297122e-04
##        30101        30178        30194        30287        30295        30358
## 3.650483e-05 1.137710e-03 2.209086e-03 9.280634e-05 1.701317e-05 1.176956e-03
##        30359        30407        30418        30427        30486        30590
## 3.439164e-05 1.186228e-03 2.701592e-04 4.462881e-04 7.409059e-05 1.292337e-04
##        30595        30782        30846        30852        30875        30900
## 2.669375e-03 2.224994e-04 1.111852e-03 4.944068e-05 7.137005e-04 2.201400e-04
##        30928        30976        31036        31132        31146        31164
## 2.821746e-04 1.074569e-04 3.071011e-04 6.269398e-04 1.092596e-03 7.786419e-04
##        31177        31241        31279        31318        31479        31580
## 4.820801e-05 5.241955e-04 3.760083e-05 6.045884e-04 4.183217e-04 9.535389e-04
##        31621        31910        31949        31975        32008        32012
## 2.990242e-04 7.804501e-04 1.916105e-05 2.958147e-05 5.710431e-04 2.072111e-05
##        32054        32102        32111        32145        32220        32250
## 1.181571e-04 6.922311e-05 7.103997e-05 1.787843e-04 1.682624e-03 1.780686e-04
##        32470        32551        32636        32724        32768        32792
## 9.781215e-04 5.615816e-04 6.541722e-04 1.576290e-05 2.172173e-04 6.377275e-04
##        32877        32917        32918        32947        33051        33061
## 2.168616e-04 1.206103e-08 4.101989e-07 7.877662e-05 4.771523e-03 2.412177e-05
##        33117        33167        33191        33204        33208        33292
```

```
## 1.494166e-03 1.139138e-04 5.408534e-04 1.543603e-03 4.369155e-04 2.487442e-03
##        33305        33336        33348        33401        33471        33484
## 3.147828e-03 1.089353e-03 1.555616e-03 1.187270e-03 3.532244e-04 6.901321e-05
##        33599        33611        33670        33717        33836        33903
## 8.692533e-05 4.623286e-06 1.144357e-03 4.291621e-04 1.717732e-04 8.849022e-04
##        33917        33918        33949        34087        34109        34114
## 5.325363e-05 8.336647e-05 5.556179e-08 1.377561e-03 3.270501e-04 6.270048e-07
##        34153        34225        34253        34273        34300        34303
## 2.265967e-04 1.445769e-03 4.804871e-04 2.171157e-03 2.872888e-04 4.829425e-04
##        34332        34423        34425        34512        34644        34650
## 8.790810e-05 2.414186e-04 1.182610e-04 1.000274e-03 5.915542e-05 6.144819e-04
##        34661        34768        34769        34788        34820        34848
## 3.489442e-04 2.001535e-07 5.332214e-05 2.373903e-05 6.306901e-04 1.222782e-03
##        34952        35060        35087        35131        35162        35239
## 4.454621e-05 4.337226e-04 1.271911e-06 1.094778e-05 3.259169e-04 6.219933e-05
##        35257        35299        35496        35567        35614        35627
## 1.706574e-04 1.993703e-04 5.579820e-04 1.570928e-04 4.338748e-04 4.948184e-04
##        35636        35690        35758        35776        35820        35904
## 6.481469e-03 2.901118e-07 2.424374e-04 3.551122e-05 8.510211e-04 7.620640e-04
##        35914        35926        36024        36061        36073        36102
## 3.588141e-04 5.111199e-04 3.750573e-04 6.025304e-05 6.713999e-05 2.167457e-04
##        36172        36246        36284        36299        36312        36372
## 2.713279e-06 3.103432e-04 3.510837e-04 5.284598e-03 1.980578e-04 1.656641e-04
##        36418        36431        36478        36485        36501        36514
## 1.633058e-04 7.788069e-04 6.895254e-04 6.076918e-04 5.507547e-04 6.445962e-04
##        36589        36632        36649        36664        36729        36855
## 2.543343e-05 5.488142e-04 2.637955e-03 1.609740e-04 8.034511e-05 2.677830e-04
##        36978        37052        37164        37213        37331        37352
## 4.929952e-06 5.709332e-04 3.287762e-05 2.724282e-05 2.158880e-05 2.349338e-10
##        37425        37432        37443        37450        37456        37501
## 4.632188e-04 1.460571e-04 2.970127e-05 2.171371e-04 7.604339e-05 3.221223e-04
##        37540        37641        37663        37665        37717        37764
## 5.508631e-05 7.802495e-05 1.005621e-04 3.872300e-04 1.344024e-03 2.048138e-05
##        37922        37927        37994        38057        38096        38154
## 1.043448e-04 3.933577e-04 6.300402e-04 3.131578e-04 3.659363e-04 1.532614e-04
##        38167        38181        38200        38265        38266        38294
## 3.012496e-06 4.386029e-04 2.225757e-03 2.180766e-06 4.535734e-05 1.839751e-05
##        38330        38350        38372        38381        38383        38401
## 2.308895e-04 5.160946e-04 1.984486e-04 1.202414e-03 9.523805e-06 8.080946e-04
##        38428        38606        38719        38853        38959        39053
## 3.863362e-05 4.895489e-04 7.510424e-06 9.503689e-05 5.383157e-04 2.618975e-05
##        39275        39365        39460        39556        39569        39612
## 4.636771e-04 7.892098e-04 7.069088e-06 5.120324e-04 1.341410e-06 2.635405e-04
##        39653        39754        39853        39868        39891        39903
## 3.786904e-04 5.052447e-04 5.260541e-06 4.691915e-04 1.027961e-03 7.574752e-06
##        39987        39988        40052        40069        40165        40252
## 4.815126e-03 2.441979e-05 9.311347e-04 1.675736e-03 2.047777e-05 2.158236e-04
##        40305        40331        40510        40611        40613        40653
## 3.169594e-03 3.520000e-03 3.131944e-04 2.904226e-04 3.130046e-04 3.811131e-04
##        40658        40776        40796        40894        40905        40947
## 1.586007e-04 6.101640e-04 1.484277e-05 5.818104e-04 1.726749e-04 2.896639e-05
##        41008        41049        41094        41107        41199        41207
## 1.003159e-03 1.364720e-03 3.167842e-04 3.524926e-06 6.407651e-04 6.413604e-07
##        41220        41241        41274        41320        41334        41373
## 6.302341e-04 2.957645e-04 1.175952e-05 4.590647e-04 2.971843e-04 3.228494e-04
##        41377        41386        41484        41515        41584        41633
## 2.069034e-04 1.463023e-04 9.616561e-04 1.729953e-03 2.150158e-03 1.049184e-03
##        41661        41736        41792        41806        41819        41968
## 4.776079e-05 7.385202e-04 2.454811e-03 6.691910e-04 3.145274e-03 1.247497e-03
##        41987        42020        42093        42125        42129        42139
## 1.880245e-03 1.456913e-03 7.149097e-04 2.810899e-03 3.509440e-03 7.145249e-04
##        42267        42319        42357        42450        42477        42503
## 9.287516e-04 1.070841e-03 2.197474e-03 1.378058e-03 1.811618e-04 5.320800e-04
##        42555        42612        42699        42714        42720        42892
```

```
## 1.236656e-03 1.525457e-03 2.720279e-03 1.670800e-03 8.918176e-04 5.895410e-04
##        42900        42931        42966        42999        43104        43119
## 1.075944e-03 1.009921e-03 1.057263e-03 2.443873e-03 8.084930e-04 1.655590e-03
##        43131        43185        43240        43254        43360        43369
## 2.306023e-03 5.991950e-04 8.092418e-04 5.973742e-04 2.485673e-03 9.786011e-04
##        43422        43584        43599        43663        43697        43779
## 7.613373e-04 2.180005e-03 1.678292e-03 8.942018e-04 1.164650e-03 5.359371e-04
##        43797        43808        43832        43863        43866        43877
## 1.498592e-03 8.700236e-04 8.870426e-04 2.261761e-03 7.583779e-04 4.076862e-05
##        43943        43945        43968        44013        44104        44107
## 1.069413e-03 2.793632e-03 2.287540e-03 8.239753e-04 1.164483e-03 5.050479e-04
##        44149        44153        44155        44159        44174        44180
## 1.293468e-04 4.116890e-04 2.824983e-03 3.161385e-03 1.529893e-03 2.688110e-03
##        44285        44347        44386        44400        44418        44512
## 7.162406e-04 1.462873e-03 1.272325e-03 1.439322e-02 9.025595e-03 1.970721e-03
##        44527        44634        44668        44677        44783        44823
## 4.950207e-04 1.532698e-03 7.070179e-05 1.537880e-03 1.730262e-03 1.136551e-04
##        44834        44842        44974        45043        45068        45128
## 1.136551e-04 4.657650e-03 9.765275e-05 6.943416e-06 3.347418e-05 1.355593e-04
##        45152        45185        45326        45426        45429        45435
## 2.022267e-05 5.665338e-05 1.117470e-03 8.934523e-05 7.392417e-05 6.835981e-05
##        45474        45483        45516        45544        45610        45618
## 2.059311e-04 1.709438e-05 7.666846e-04 1.329172e-04 2.648241e-06 8.083148e-04
##        45707        45782        45821        45864        45955        45972
## 8.947658e-05 1.689140e-04 7.019465e-06 3.073108e-04 1.509432e-05 7.396840e-06
##        46016        46041        46056        46076        46114        46355
## 4.788367e-04 9.624495e-05 5.640027e-04 1.573840e-04 3.748114e-08 3.459696e-06
##        46363        46378        46424        46511        46551        46630
## 5.016855e-05 5.657801e-04 1.845325e-06 5.229988e-04 1.300595e-05 2.372599e-04
##        46631        46643        46689        46762        46785        46797
## 4.661156e-08 4.819791e-05 1.135731e-03 3.129796e-02 5.702942e-03 8.572260e-03
##        46818        46846        46878        47003        47006        47106
## 4.603028e-03 7.641900e-03 7.908645e-03 4.159473e-03 1.433238e-03 4.655907e-03
##        47110        47118        47145        47182        47219        47238
## 6.775943e-03 5.097491e-03 3.773253e-03 9.453518e-03 5.191056e-03 1.739014e-03
##        47242        47258        47279        47368        47420        47426
## 2.040110e-03 5.556029e-02 8.756419e-03 3.467792e-03 2.760780e-03 4.003592e-03
##        47464        47523        47593        47615        47660        47736
## 5.102456e-03 7.335929e-03 7.436293e-05 7.157242e-06 5.100868e-04 5.280124e-03
##        47787        47882        47938        48007        48033        48086
## 6.139504e-04 1.781675e-03 1.260769e-03 4.505970e-04 1.342820e-03 9.179995e-07
##        48162        48260        48306        48316        48379        48510
## 1.299238e-04 2.056839e-04 5.095379e-04 2.934170e-04 1.568395e-04 2.237660e-04
##        48513        48518        48607        48610        48624        48744
## 4.036851e-05 2.524972e-04 1.236997e-04 3.362500e-04 1.221551e-04 3.669218e-04
##        48785        48920        48957        49055        49107        49130
## 1.945579e-04 4.187240e-05 5.371363e-04 1.808493e-03 7.178668e-07 2.643035e-04
##        49160        49255        49269        49446        49517        49530
## 4.443801e-04 5.792414e-04 5.037152e-04 4.515033e-04 1.874444e-03 1.336395e-03
##        49534        49555        49567
## 1.575188e-05 4.417746e-04 4.373895e-04
```

6 Adding factors

```
m6 <- update(m5, ~.+fuelType+transmission,data=df[!df$mout=="YesMOut",])
vif(m6)
```

```
##                   GVIF Df GVIF^(1/(2*Df))
## log(mileage) 2.001997  1        1.414920
## sqrt(tax)    2.090996  1        1.446028
## poly(mpg, 2) 2.671361  2        1.278448
## poly(age, 2) 2.352151  2        1.238415
```

```
## fuelType     1.519668  3        1.072239
## transmission 1.385959  2        1.085020
```

```
summary(m6)
```

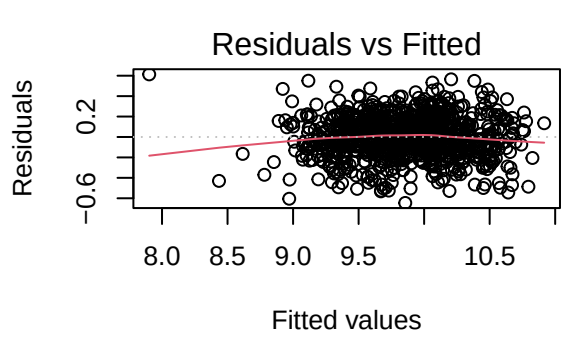
```
##
## Call:
## lm(formula = log(price) ~ log(mileage) + sqrt(tax) + poly(mpg,
##     2) + poly(age, 2) + fuelType + transmission, data = df[!df$mout ==
##     "YesMOut", ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.64566 -0.11048  0.00254  0.13311  0.61303
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)            10.132790   0.060522 167.424  < 2e-16 ***
## log(mileage)           -0.024024   0.005484  -4.381 1.32e-05 ***
## sqrt(tax)               0.011911   0.002641   4.511 7.26e-06 ***
## poly(mpg, 2)1          -6.532315   0.303736 -21.507  < 2e-16 ***
## poly(mpg, 2)2           1.743927   0.209186   8.337 2.65e-16 ***
## poly(age, 2)1          -5.874102   0.274299 -21.415  < 2e-16 ***
## poly(age, 2)2          -1.364110   0.225923  -6.038 2.24e-09 ***
## fuelTypeHybrid         -0.053710   0.139318  -0.386  0.69994
## fuelTypeOther          -0.450394   0.140110  -3.215  0.00135 **
## fuelTypePetrol         -0.273523   0.015473 -17.677  < 2e-16 ***
## transmissionManual     -0.184740   0.018193 -10.155  < 2e-16 ***
## transmissionSemi-Auto   0.057320   0.016385   3.498  0.00049 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1958 on 952 degrees of freedom
## Multiple R-squared:  0.8155, Adjusted R-squared:  0.8133
## F-statistic: 382.4 on 11 and 952 DF,  p-value: < 2.2e-16
```

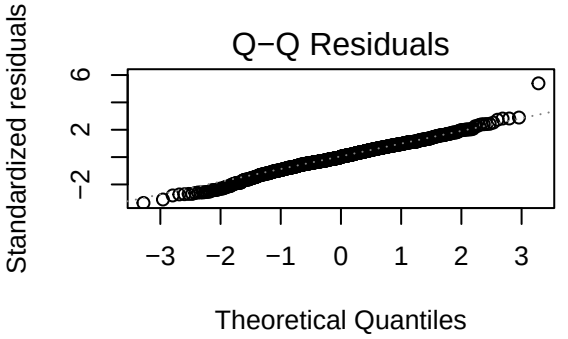
```
Anova(m6)
```

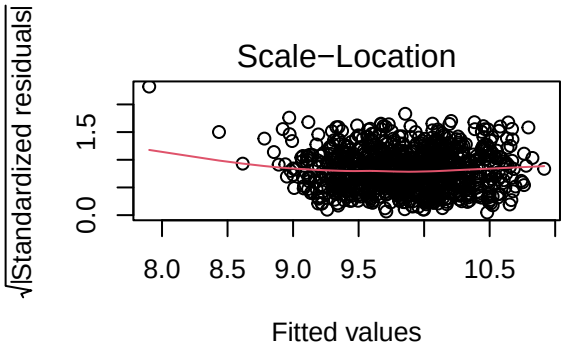
```
## Warning in printHypothesis(L, rhs, names(b)): one or more coefficients in the hypothesis include
##      arithmetic operators in their names;
##   the printed representation of the hypothesis will be omitted
```

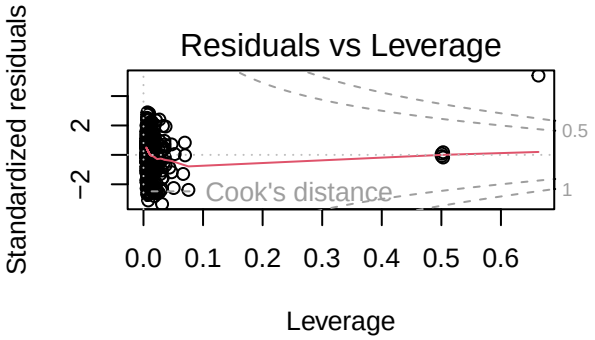
```
## Anova Table (Type II tests)
##
## Response: log(price)
##              Sum Sq  Df F value    Pr(>F)
## log(mileage)  0.736   1  19.189 1.315e-05 ***
## sqrt(tax)     0.780   1  20.348 7.262e-06 ***
## poly(mpg, 2) 23.317   2 304.110 < 2.2e-16 ***
## poly(age, 2) 24.564   2 320.382 < 2.2e-16 ***
## fuelType     12.136   3 105.524 < 2.2e-16 ***
## transmission  8.537   2 111.349 < 2.2e-16 ***
## Residuals    36.496 952
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
par(mfrow=c(2,2))
plot(m6,id.n=0)
```

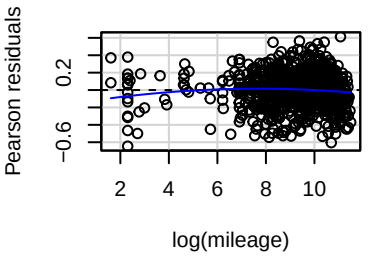


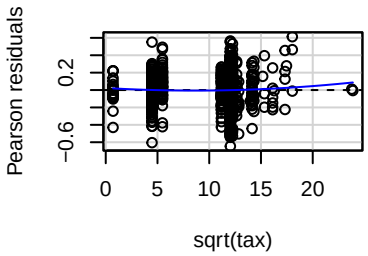


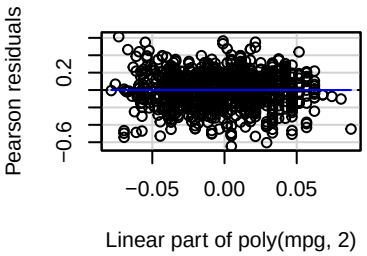


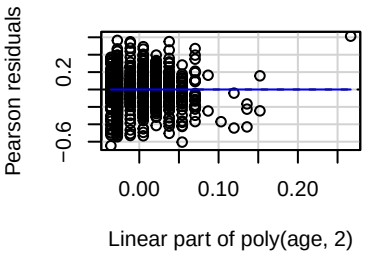


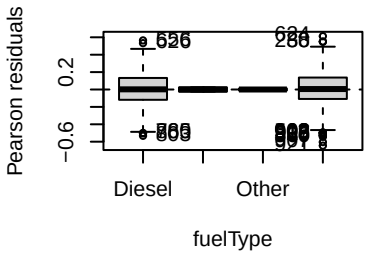

```
par(mfrow=c(1,1))
residualPlots(m6,id=list(method=cooks.distance(m5),n=10))
```

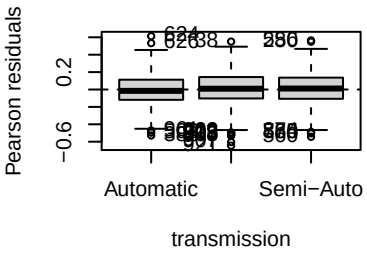


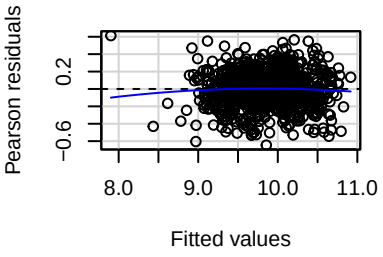










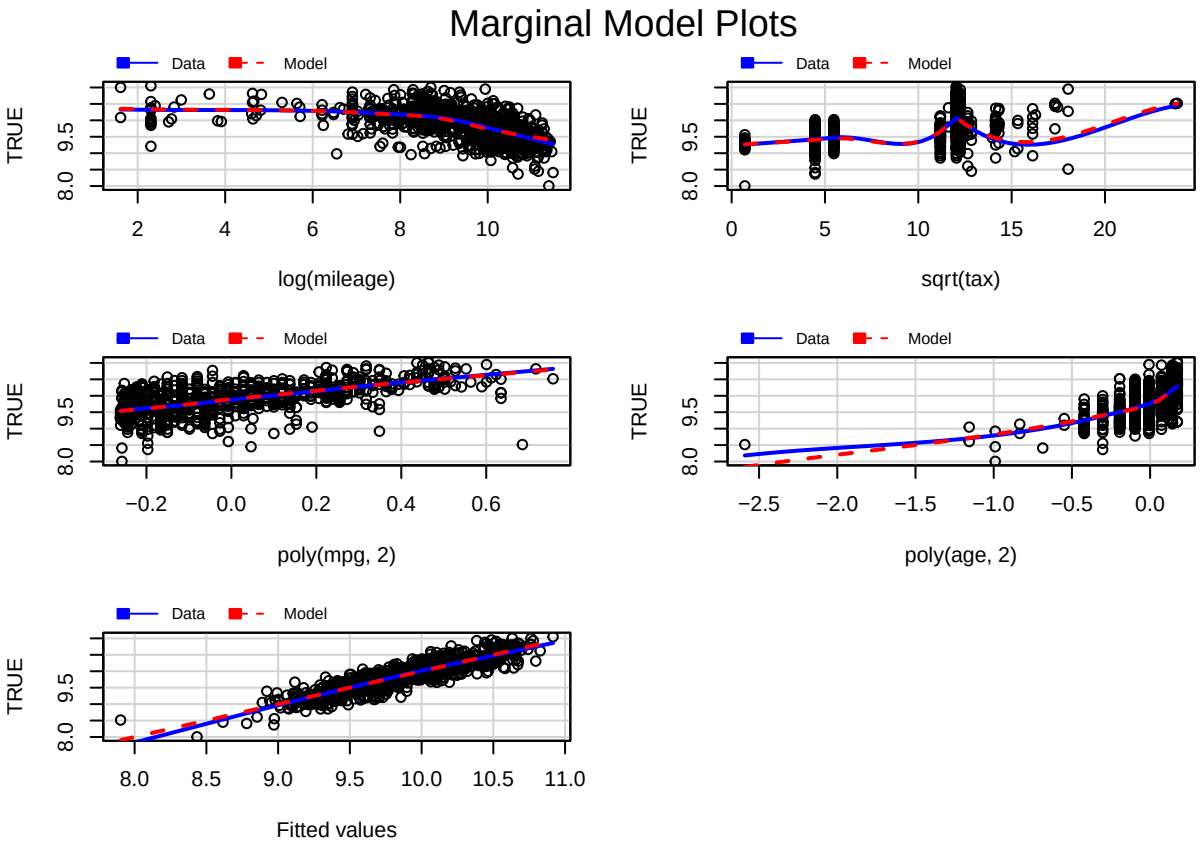



```
##              Test stat Pr(>|Test stat|)
## log(mileage)   -3.2817         0.001069 **
## sqrt(tax)       1.1432         0.253255
## poly(mpg, 2)
## poly(age, 2)
## fuelType
## transmission
## Tukey test     -1.8385         0.065985 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

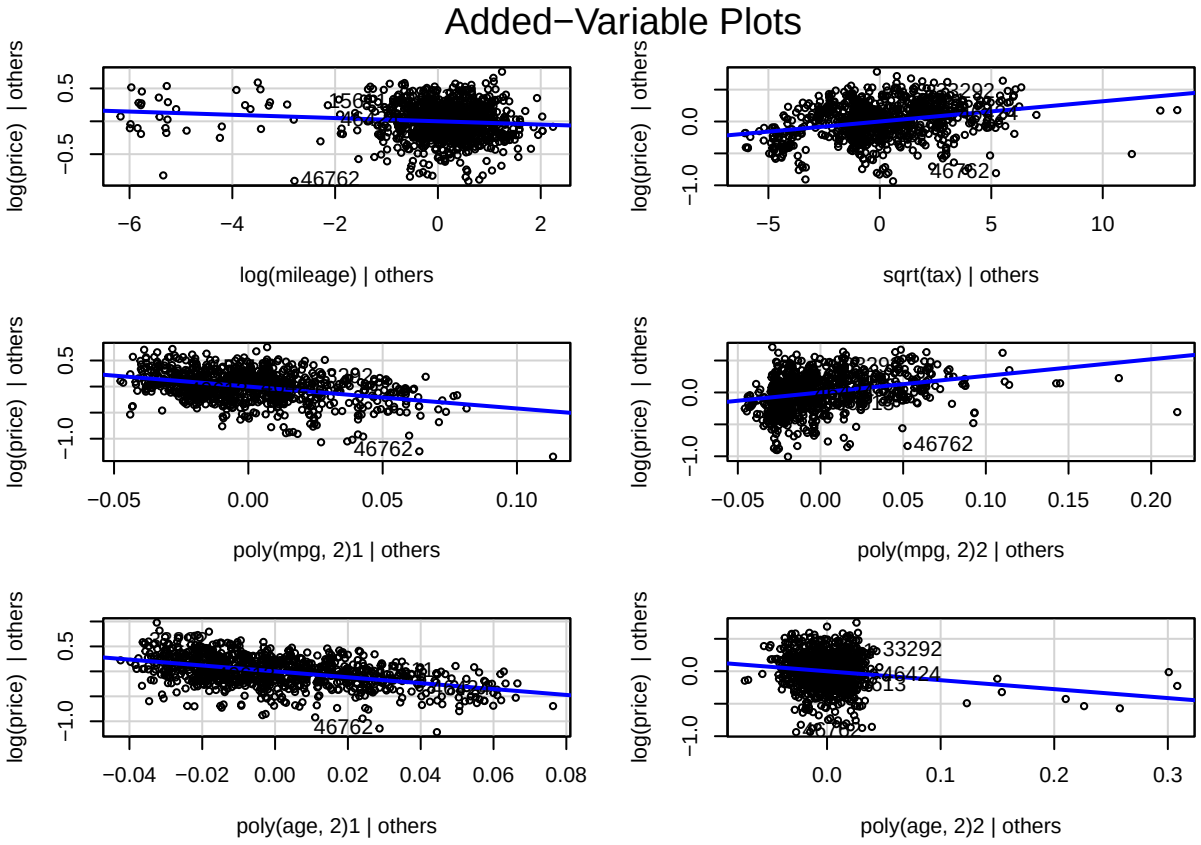
```
marginalModelPlots(m6)
```

```
## Warning in mmps(...): Splines and/or polynomials replaced by a fitted linear
## combination
```

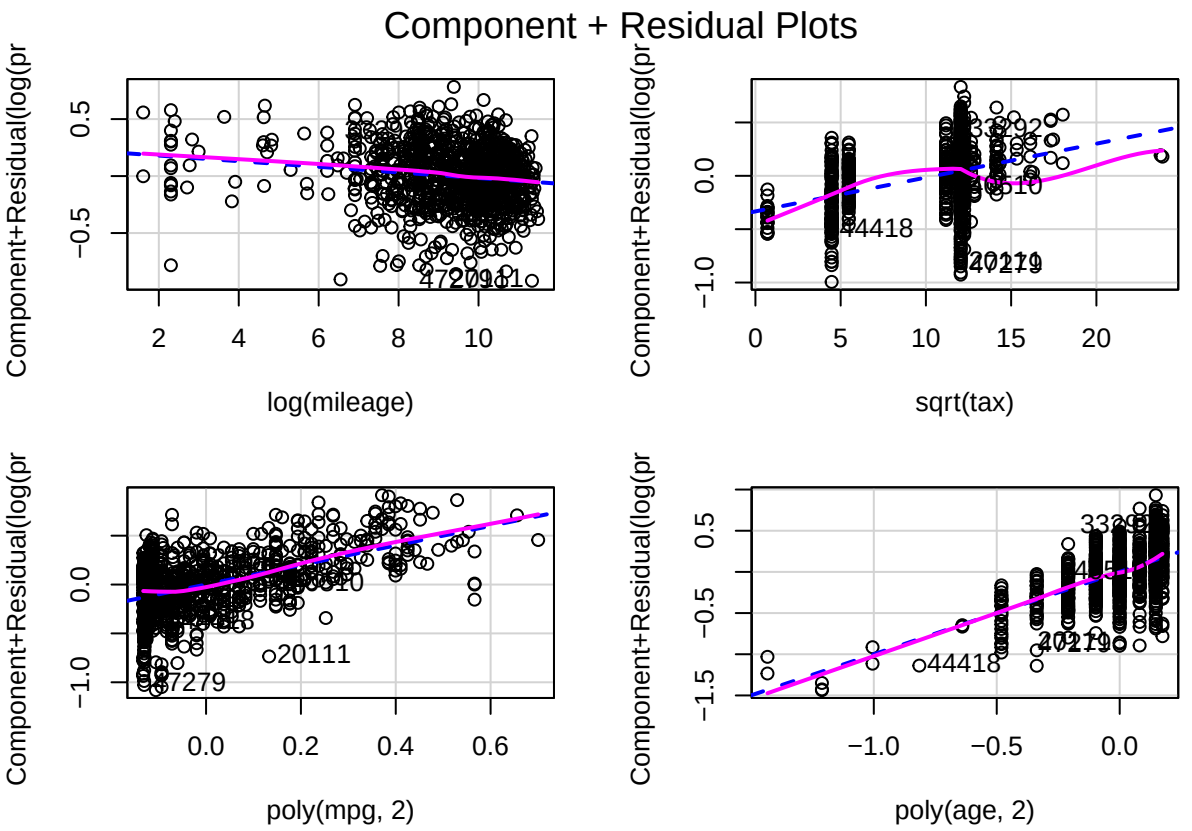
```
## Warning in mmps(...): Interactions and/or factors skipped
```



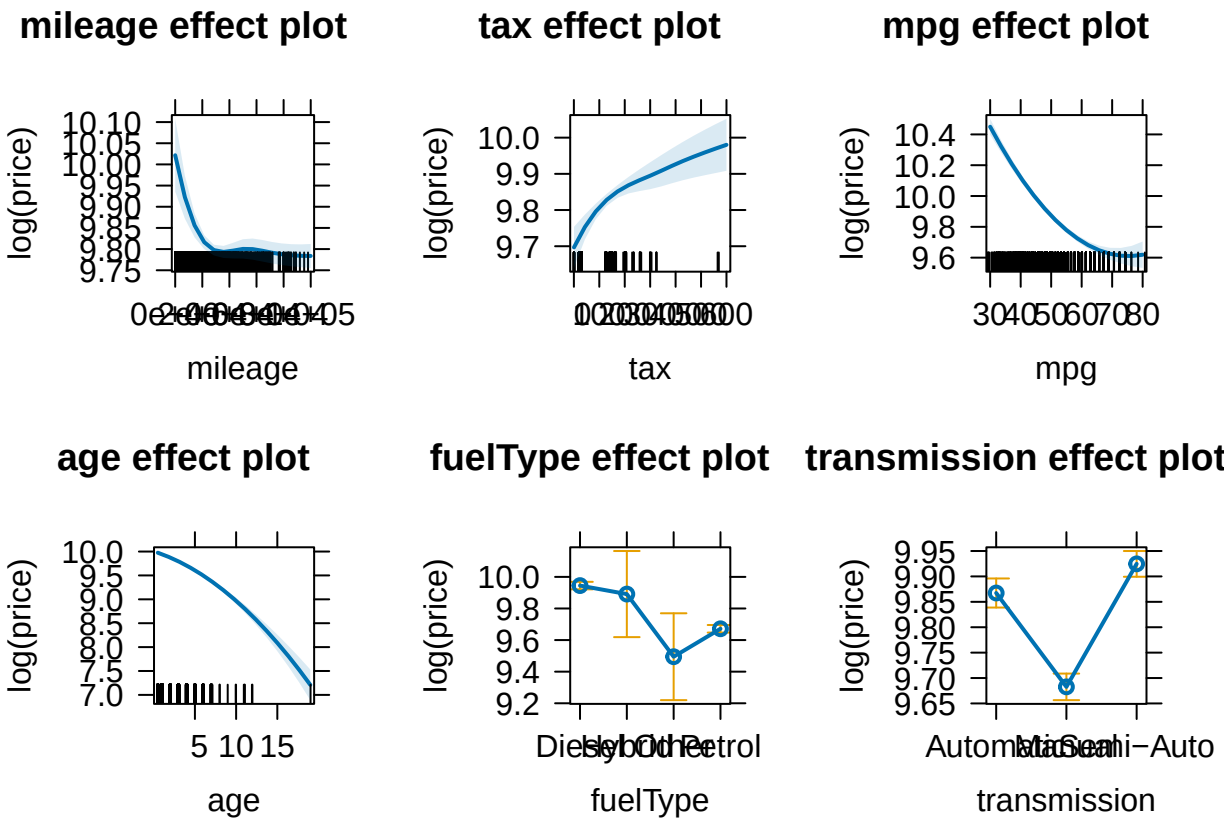

```
avPlots(m5,id=list(method=hatvalues(m6),n=5))
```




```
crPlots(m5,id=list(method=cooks.distance(m6),n=5))
```




```
library(effects)
plot(allEffects(m6))
```




```
m7<-update(m6, ~.+aux_tax*fuelType,data=df[!df$mout=="YesMOut",])
summary(m7)
```

```
##
## Call:
## lm(formula = log(price) ~ log(mileage) + sqrt(tax) + poly(mpg,
##     2) + poly(age, 2) + fuelType + transmission + aux_tax + fuelType:aux_tax,
##     data = df[!df$mout == "YesMOut", ])
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.64431 -0.11251  0.00467  0.13316  0.58010
##
## Coefficients: (4 not defined because of singularities)
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    10.137181   0.062735 161.588  < 2e-16 ***
## log(mileage)                   -0.023990   0.005472  -4.384 1.29e-05 ***
## sqrt(tax)                       0.015317   0.004050   3.782 0.000165 ***
## poly(mpg, 2)1                  -6.572979   0.304880 -21.559  < 2e-16 ***
## poly(mpg, 2)2                   1.649371   0.220424   7.483 1.66e-13 ***
## poly(age, 2)1                  -6.037158   0.324850 -18.584  < 2e-16 ***
## poly(age, 2)2                  -1.293861   0.233859  -5.533 4.08e-08 ***
## fuelTypeHybrid                 -0.053271   0.139423  -0.382 0.702488
## fuelTypeOther                  -0.455660   0.139843  -3.258 0.001160 **
## fuelTypePetrol                 -0.304046   0.027838 -10.922  < 2e-16 ***
## transmissionManual             -0.184923   0.018188 -10.167  < 2e-16 ***
## transmissionSemi-Auto           0.056055   0.016371   3.424 0.000643 ***
## aux_tax(125,145]               -0.043637   0.033707  -1.295 0.195778
## aux_tax(145,570]               -0.098106   0.037066  -2.647 0.008261 **
## fuelTypeHybrid:aux_tax(125,145]        NA         NA      NA       NA
## fuelTypeOther:aux_tax(125,145]         NA         NA      NA       NA
## fuelTypePetrol:aux_tax(125,145]  0.017298   0.031455   0.550 0.582487
## fuelTypeHybrid:aux_tax(145,570]        NA         NA      NA       NA
## fuelTypeOther:aux_tax(145,570]         NA         NA      NA       NA
## fuelTypePetrol:aux_tax(145,570]  0.107210   0.039789   2.694 0.007174 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1951 on 948 degrees of freedom
## Multiple R-squared:  0.8176, Adjusted R-squared:  0.8147
## F-statistic: 283.3 on 15 and 948 DF,  p-value: < 2.2e-16
```

```
#vif(m7)
#summary(m7)
#Anova(m7)
#plot(allEffects(m7))
m8 <- step( m7, k=log(nrow(df[!df$mout=="YesMOut",])))
```

```
## Start:  AIC=-3057.35
## log(price) ~ log(mileage) + sqrt(tax) + poly(mpg, 2) + poly(age,
##     2) + fuelType + transmission + aux_tax + fuelType:aux_tax
##
##                     Df Sum of Sq    RSS     AIC
## - fuelType:aux_tax   2    0.3288 36.401 -3062.3
## <none>                           36.072 -3057.3
## - sqrt(tax)          1    0.5443 36.616 -3049.8
## - log(mileage)       1    0.7313 36.803 -3044.9
## - transmission       2    8.4232 44.495 -2868.8
## - poly(age, 2)       2   20.7959 56.868 -2632.3
## - poly(mpg, 2)       2   22.7073 58.779 -2600.4
##
## Step:  AIC=-3062.35
## log(price) ~ log(mileage) + sqrt(tax) + poly(mpg, 2) + poly(age,
##     2) + fuelType + transmission + aux_tax
##
##                Df Sum of Sq    RSS     AIC
## - aux_tax       2    0.0950 36.496 -3073.6
## <none>                      36.401 -3062.3
## - sqrt(tax)     1    0.6554 37.056 -3052.0
## - log(mileage)  1    0.7350 37.136 -3049.9
## - transmission  2    8.4612 44.862 -2874.6
## - fuelType      3   11.9717 48.373 -2808.9
## - poly(age, 2)  2   20.9083 57.309 -2638.6
```

```
## - poly(mpg, 2)  2   23.1239 59.525 -2602.0
##
## Step:  AIC=-3073.58
## log(price) ~ log(mileage) + sqrt(tax) + poly(mpg, 2) + poly(age,
##     2) + fuelType + transmission
##
##                Df Sum of Sq    RSS     AIC
## <none>                      36.496 -3073.6
## - log(mileage)  1    0.7356 37.232 -3061.2
## - sqrt(tax)     1    0.7801 37.276 -3060.1
## - transmission  2    8.5374 45.033 -2884.7
## - fuelType      3   12.1362 48.632 -2817.4
## - poly(mpg, 2)  2   23.3168 59.813 -2611.1
## - poly(age, 2)  2   24.5644 61.060 -2591.2
```

```
m9 <- step( m6, k=log(nrow(df[!df$mout=="YesMOut",])))
```

```
## Start:  AIC=-3073.58
## log(price) ~ log(mileage) + sqrt(tax) + poly(mpg, 2) + poly(age,
##     2) + fuelType + transmission
##
##                Df Sum of Sq    RSS     AIC
## <none>                      36.496 -3073.6
## - log(mileage)  1    0.7356 37.232 -3061.2
## - sqrt(tax)     1    0.7801 37.276 -3060.1
## - transmission  2    8.5374 45.033 -2884.7
## - fuelType      3   12.1362 48.632 -2817.4
## - poly(mpg, 2)  2   23.3168 59.813 -2611.1
## - poly(age, 2)  2   24.5644 61.060 -2591.2
```

```
AIC(m5,m6,m7,m8,m9)
```

```
## Warning in AIC.default(m5, m6, m7, m8, m9): no tots els models estan ajustats
## al mateix nombre d'observacions
```

```
##    df       AIC
## m5  8  178.7587
## m6 13 -394.3162
## m7 17 -397.5772
## m8 13 -394.3162
## m9 13 -394.3162
```

```
summary(m7)
```

```
##
## Call:
## lm(formula = log(price) ~ log(mileage) + sqrt(tax) + poly(mpg,
##     2) + poly(age, 2) + fuelType + transmission + aux_tax + fuelType:aux_tax,
##     data = df[!df$mout == "YesMOut", ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.64431 -0.11251  0.00467  0.13316  0.58010
##
## Coefficients: (4 not defined because of singularities)
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   10.137181   0.062735 161.588  < 2e-16 ***
## log(mileage)                  -0.023990   0.005472  -4.384 1.29e-05 ***
## sqrt(tax)                      0.015317   0.004050   3.782 0.000165 ***
## poly(mpg, 2)1                 -6.572979   0.304880 -21.559  < 2e-16 ***
## poly(mpg, 2)2                  1.649371   0.220424   7.483 1.66e-13 ***
## poly(age, 2)1                 -6.037158   0.324850 -18.584  < 2e-16 ***
## poly(age, 2)2                 -1.293861   0.233859  -5.533 4.08e-08 ***
## fuelTypeHybrid                -0.053271   0.139423  -0.382 0.702488
## fuelTypeOther                 -0.455660   0.139843  -3.258 0.001160 **
## fuelTypePetrol                -0.304046   0.027838 -10.922  < 2e-16 ***
```

```
## transmissionManual              -0.184923   0.018188 -10.167  < 2e-16 ***
## transmissionSemi-Auto            0.056055   0.016371   3.424 0.000643 ***
## aux_tax(125,145]                -0.043637   0.033707  -1.295 0.195778
## aux_tax(145,570]                -0.098106   0.037066  -2.647 0.008261 **
## fuelTypeHybrid:aux_tax(125,145]        NA         NA      NA       NA
## fuelTypeOther:aux_tax(125,145]         NA         NA      NA       NA
## fuelTypePetrol:aux_tax(125,145]  0.017298   0.031455   0.550 0.582487
## fuelTypeHybrid:aux_tax(145,570]        NA         NA      NA       NA
## fuelTypeOther:aux_tax(145,570]         NA         NA      NA       NA
## fuelTypePetrol:aux_tax(145,570]  0.107210   0.039789   2.694 0.007174 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1951 on 948 degrees of freedom
## Multiple R-squared:  0.8176, Adjusted R-squared:  0.8147
## F-statistic: 283.3 on 15 and 948 DF,  p-value: < 2.2e-16
```

7 Diagnostics for Influential data:

```
dfwork <- df[!df$mout=="YesMOut",]
m5<-lm(log(price)~log(mileage)+sqrt(tax)+poly(mpg,2)+poly(age,2),data=df2)

vif(m5)
```

```
##                  GVIF Df GVIF^(1/(2*Df))
## log(mileage) 2.052541  1        1.432669
## sqrt(tax)    1.932571  1        1.390169
## poly(mpg, 2) 1.917313  2        1.176720
## poly(age, 2) 2.391310  2        1.243538
```

```
summary(m5)
```

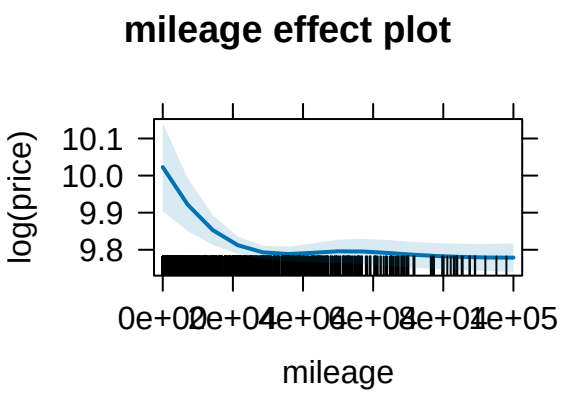
```
##
## Call:
## lm(formula = log(price) ~ log(mileage) + sqrt(tax) + poly(mpg,
##     2) + poly(age, 2), data = df2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.97502 -0.16879  0.01518  0.18209  0.78473
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    9.759964   0.077254 126.337  < 2e-16 ***
## log(mileage)  -0.024655   0.007500  -3.287  0.00105 **
## sqrt(tax)      0.031730   0.003435   9.238  < 2e-16 ***
## poly(mpg, 2)1 -4.201080   0.354069 -11.865  < 2e-16 ***
## poly(mpg, 2)2  2.598306   0.276577   9.395  < 2e-16 ***
## poly(age, 2)1 -5.905876   0.380619 -15.516  < 2e-16 ***
## poly(age, 2)2 -1.376830   0.300255  -4.586 5.13e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2643 on 956 degrees of freedom
## Multiple R-squared:  0.6593, Adjusted R-squared:  0.6571
## F-statistic: 308.3 on 6 and 956 DF,  p-value: < 2.2e-16
```

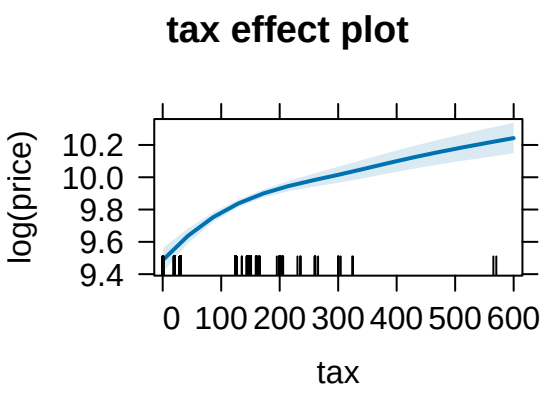
```
Anova(m5)
```

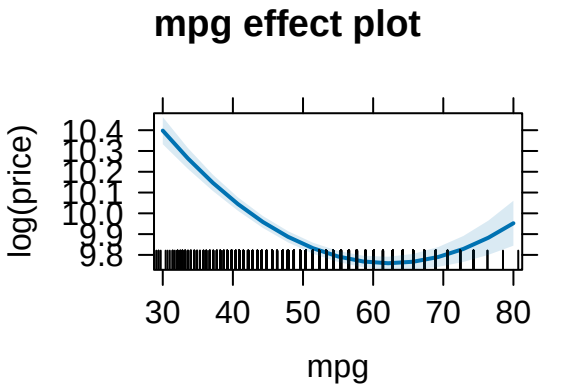
```
## Anova Table (Type II tests)
##
## Response: log(price)
##              Sum Sq  Df F value    Pr(>F)
```

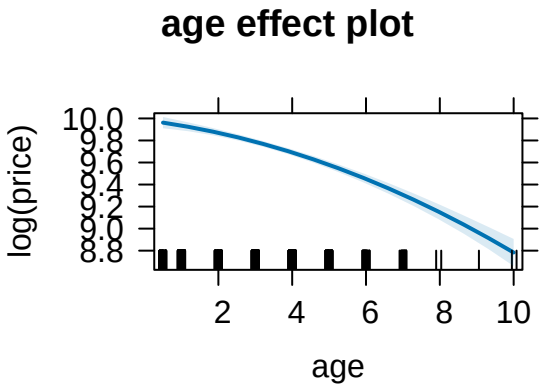
```
## log(mileage)  0.755   1  10.807  0.001048 **
## sqrt(tax)     5.960   1  85.338 < 2.2e-16 ***
## poly(mpg, 2) 18.919   2 135.451 < 2.2e-16 ***
## poly(age, 2) 24.113   2 172.633 < 2.2e-16 ***
## Residuals    66.766 956
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(allEffects(m5))
```

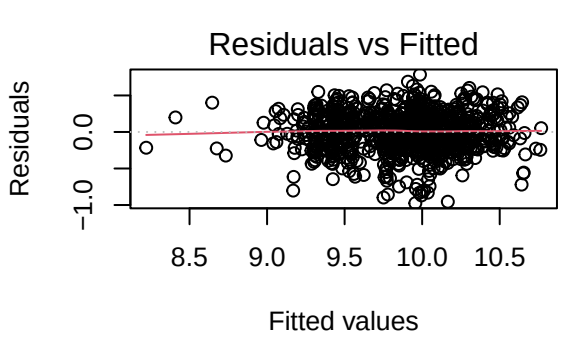


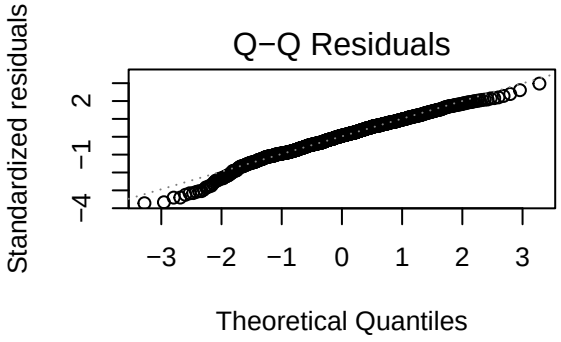


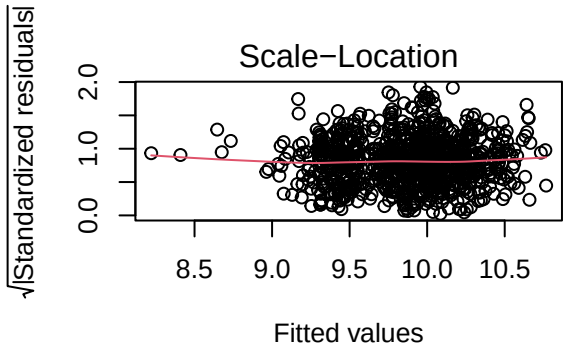


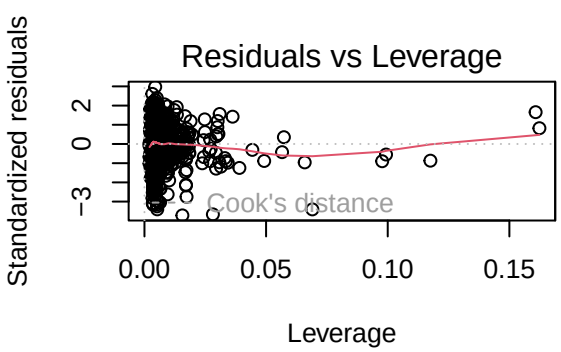



```
par(mfrow=c(2,2))
plot(m5,id.n=0)
```









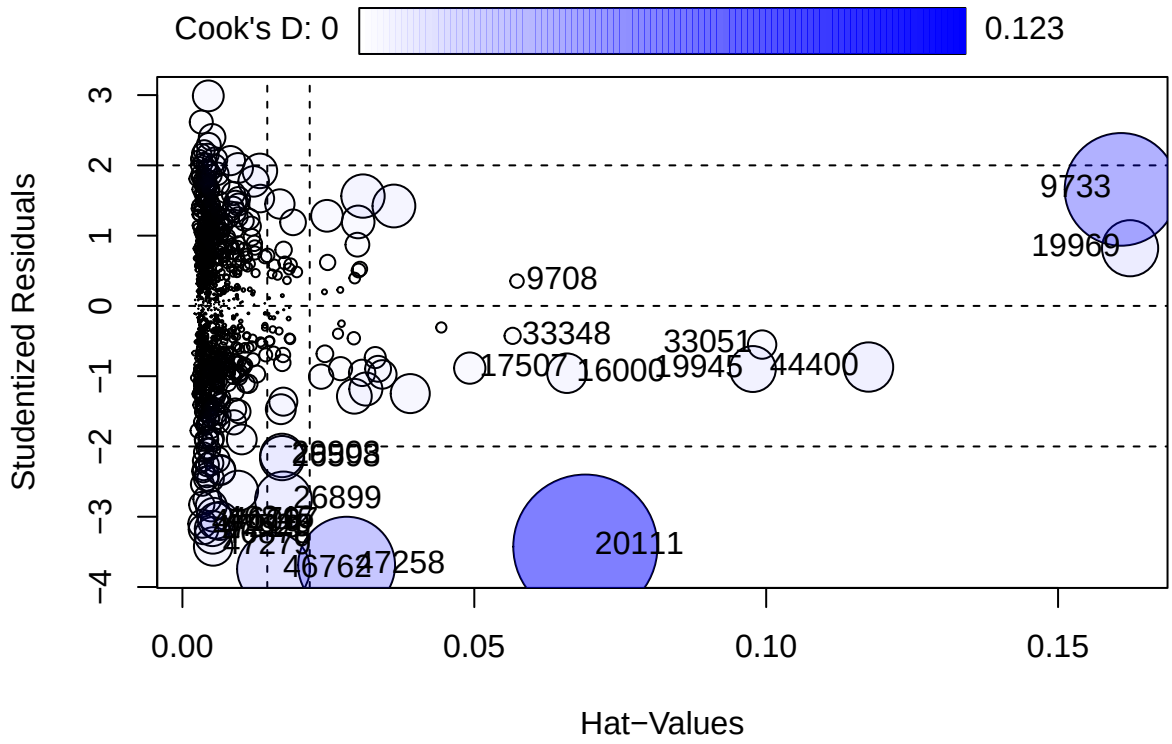
```
par(mfrow=c(1,1))

llres <- which(abs(rstudent(m5))>3);llres
```

```
## 20111 46762 46797 46846 46878 47106 47219 47258 47279 47523
##   373   904   906   908   909   912   917   920   921   926
```

```
which(row.names(df2) %in% names(rstudent(m5)[llres]))
```

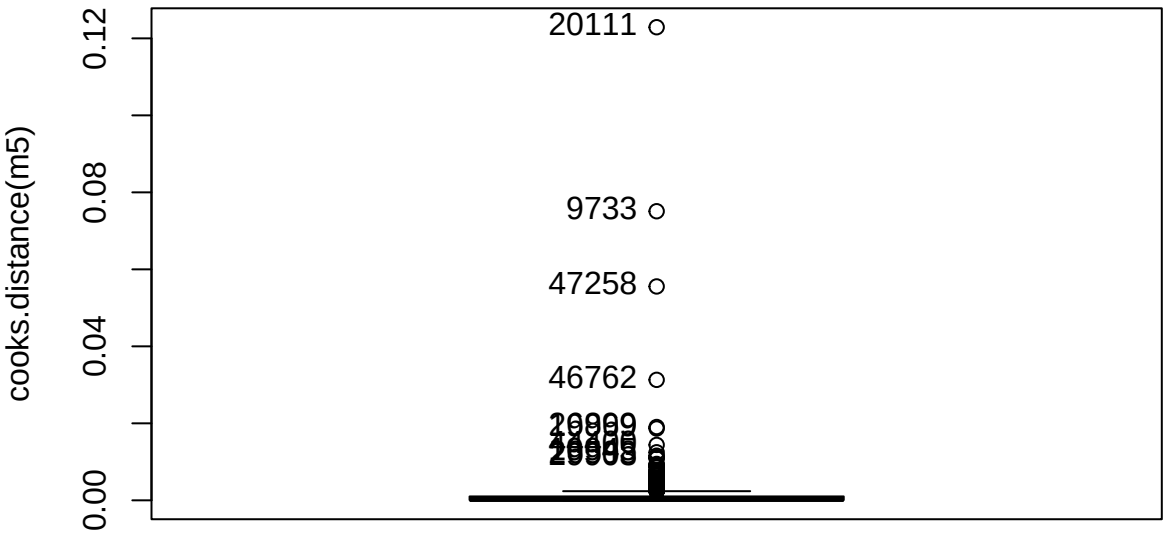
```
##  [1] 373 904 906 908 909 912 917 920 921 926
```

```
influencePlot(m5, id=list(n=10))
```




```
##          StudRes         Hat       CookD
## 9708   0.3529672 0.057314963 0.001083103
## 9733   1.6579262 0.160786129 0.075095616
## 16000 -0.9577222 0.065869454 0.009240493
## 17507 -0.8850064 0.049219957 0.005793680
## 19945 -0.8980776 0.097712134 0.012480183
## 19969  0.8206597 0.162340476 0.018652447
## 20111 -3.4254539 0.069022383 0.122896623
## 26598 -2.1654359 0.017072300 0.011590186
## 26899 -2.7604927 0.017306529 0.019040131
## 29903 -2.1279707 0.017059012 0.011185613
## 33051 -0.5501960 0.099307054 0.004771523
## 33348 -0.4259872 0.056564935 0.001555616
## 44400 -0.8696283 0.117536955 0.014393220
## 46762 -3.7438451 0.015596414 0.031297962
## 46797 -3.0594208 0.006425350 0.008572260
## 46846 -3.0565697 0.005742528 0.007641900
## 46878 -3.2654477 0.005216880 0.007908645
## 47106 -3.0992775 0.003411843 0.004655907
## 47219 -3.1799516 0.003614597 0.005191056
## 47258 -3.6900067 0.028126256 0.055560292
## 47279 -3.4229888 0.005262155 0.008756419
## 47523 -3.1673855 0.005140402 0.007335929
```

```
Boxplot(cooks.distance(m5),id=list(labels=row.names(df2)))
```




```
##  [1] "20111" "9733"  "47258" "46762" "26899" "19969" "44400" "19945" "26598"
## [10] "29903"
```

```
llout<-which(abs(cooks.distance(m5))>0.05);length(llout)
```

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

```
which(row.names(df2) %in% names(cooks.distance(m5)[llout]))
```

```
## [1] 190 373 920
```

```
llrem<-unique(c(llout,llres));llrem
```

```
##  [1] 190 373 920 904 906 908 909 912 917 921 926
```

```
m7<-lm(log(price)~log(mileage)+sqrt(tax)+poly(mpg,2)+poly(age,2),data=df2[-llrem,])

vif(m7)
```

```
##                  GVIF Df GVIF^(1/(2*Df))
## log(mileage) 2.064748  1        1.436923
## sqrt(tax)    1.988671  1        1.410203
## poly(mpg, 2) 1.983094  2        1.186686
## poly(age, 2) 2.393081  2        1.243768
```

```
summary(m7)
```

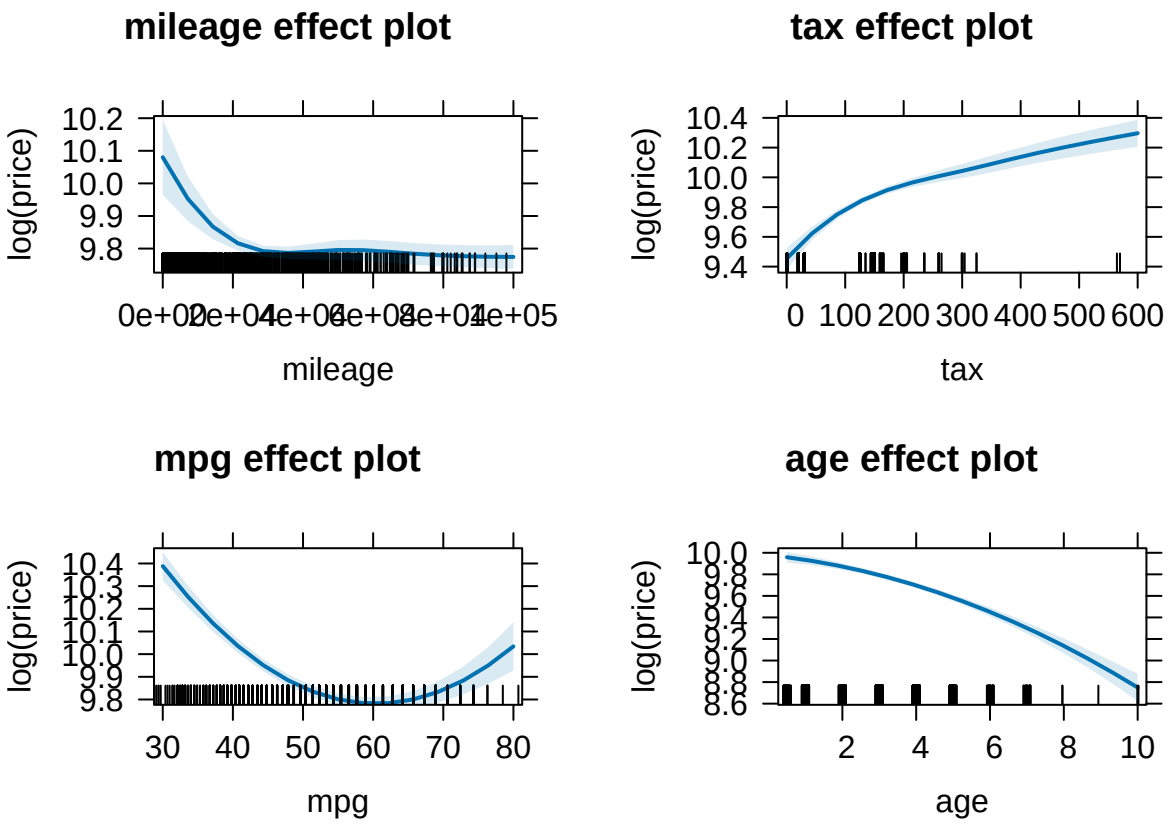
```
##
## Call:
## lm(formula = log(price) ~ log(mileage) + sqrt(tax) + poly(mpg,
##     2) + poly(age, 2), data = df2[-llrem, ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.81572 -0.16624  0.00947  0.16907  0.77931
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    9.790348   0.074237 131.879  < 2e-16 ***
## log(mileage)  -0.030873   0.007205  -4.285 2.02e-05 ***
## sqrt(tax)      0.035332   0.003300  10.706  < 2e-16 ***
## poly(mpg, 2)1 -3.671710   0.337915 -10.866  < 2e-16 ***
## poly(mpg, 2)2  2.725630   0.263342  10.350  < 2e-16 ***
## poly(age, 2)1 -5.731455   0.361285 -15.864  < 2e-16 ***
## poly(age, 2)2 -1.433643   0.281551  -5.092 4.28e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2494 on 945 degrees of freedom
## Multiple R-squared:  0.6864, Adjusted R-squared:  0.6844
```

```
## F-statistic: 344.7 on 6 and 945 DF,  p-value: < 2.2e-16
```

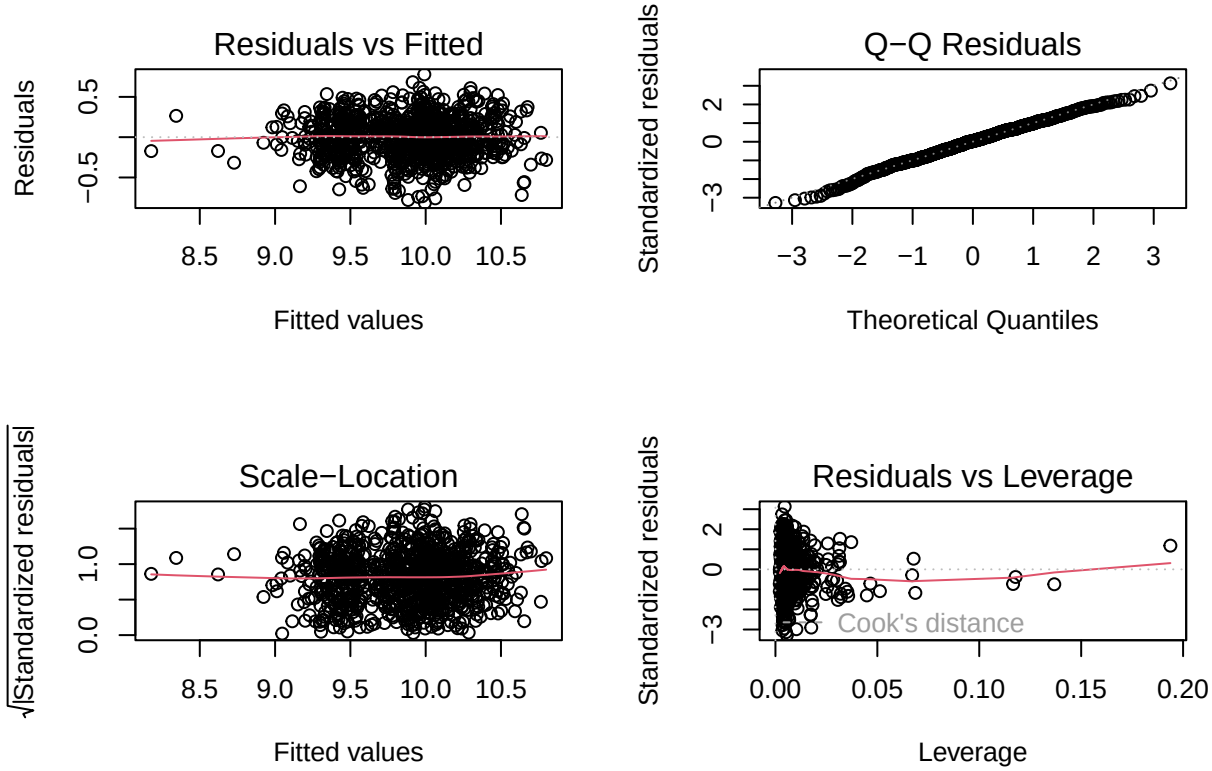
```
Anova(m7)
```

```
## Anova Table (Type II tests)
##
## Response: log(price)
##               Sum Sq  Df F value    Pr(>F)
## log(mileage)   1.142   1  18.358 2.017e-05 ***
## sqrt(tax)      7.131   1 114.625 < 2.2e-16 ***
## poly(mpg, 2) 16.980   2 136.463 < 2.2e-16 ***
## poly(age, 2) 22.926   2 184.244 < 2.2e-16 ***
## Residuals    58.794 945
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(allEffects(m7))
```

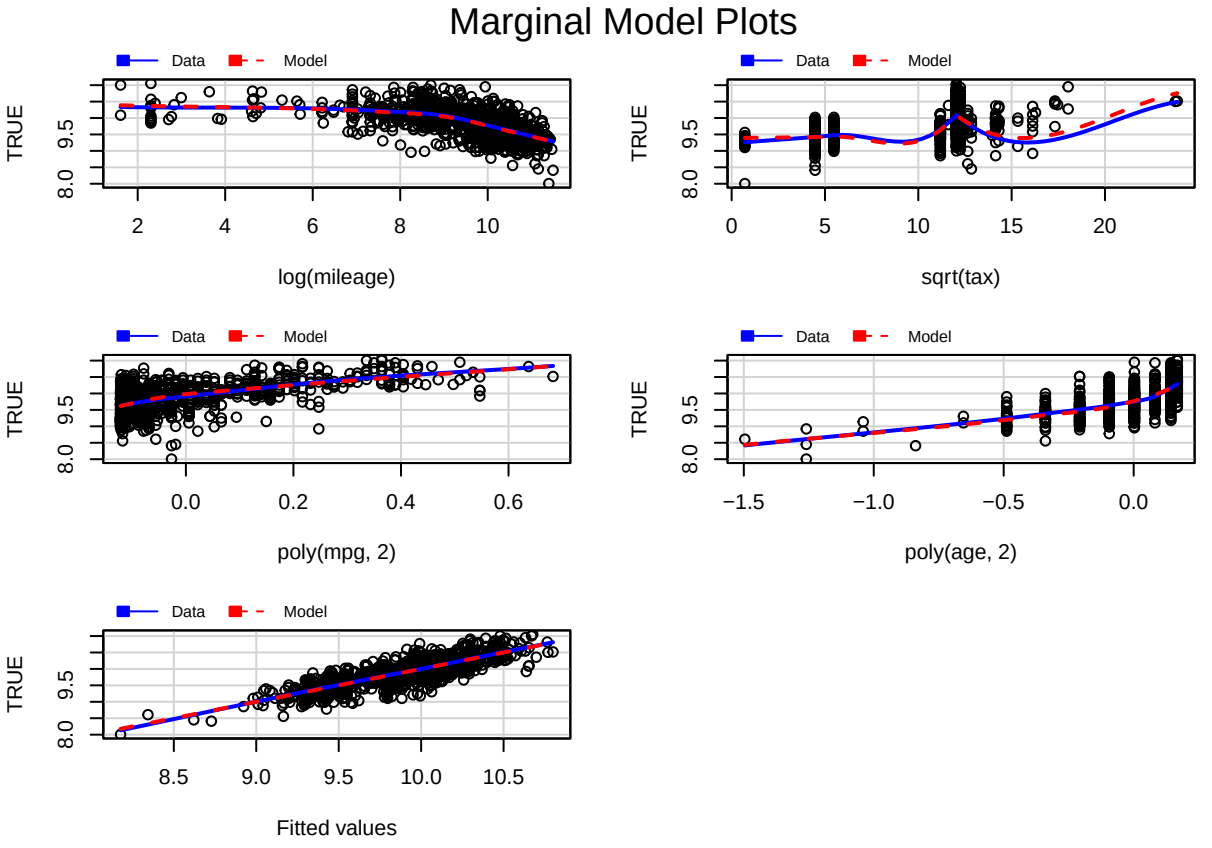



```
par(mfrow=c(2,2))
plot(m7,id.n=0)
```

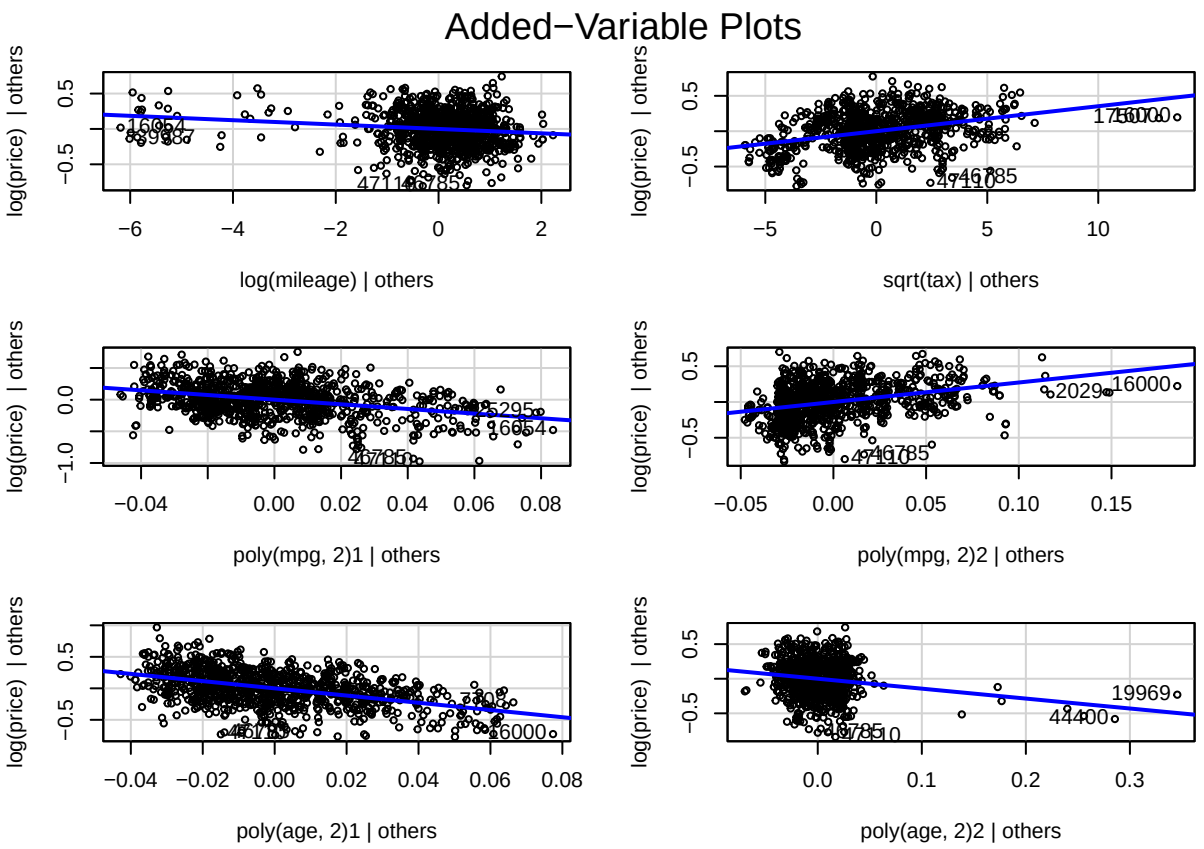



```
par(mfrow=c(1,1))

marginalModelPlots(m7)
```

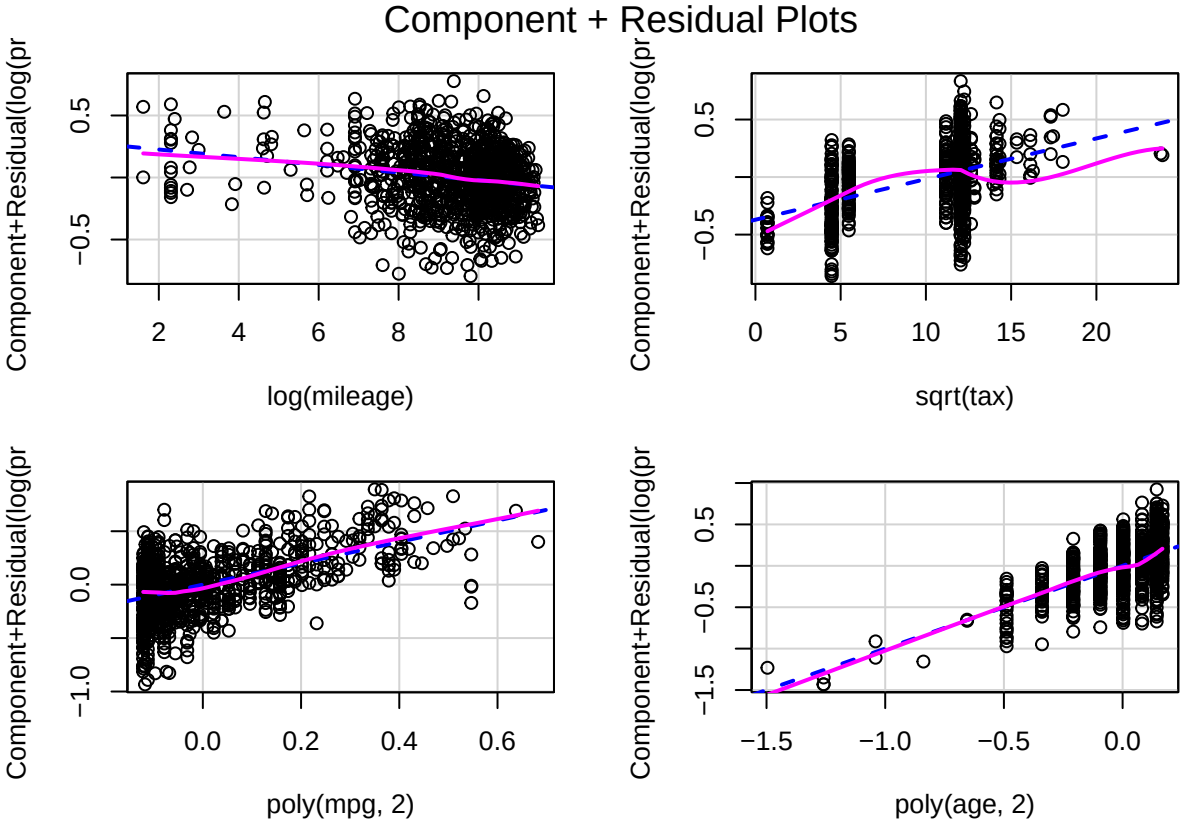
```
## Warning in mmps(...): Splines and/or polynomials replaced by a fitted linear
## combination
```



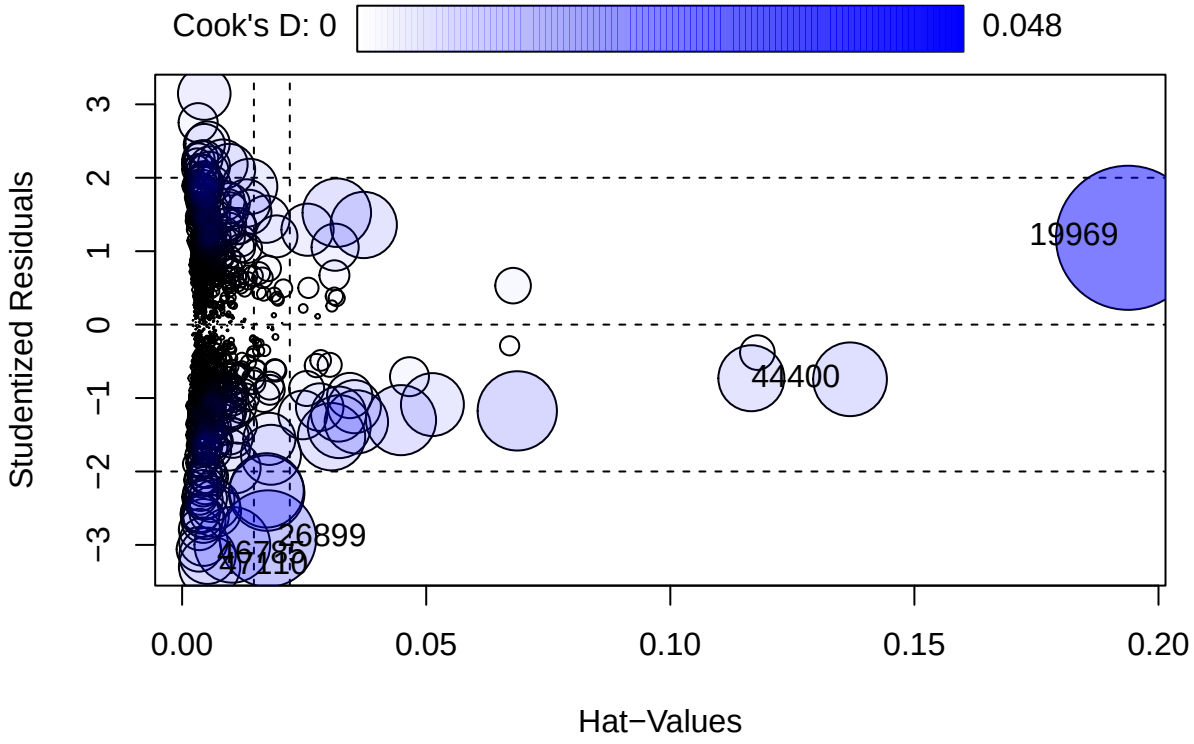

```
avPlots(m7)
```




```
crPlots(m7)
```




```
influencePlot(m7)
```




```
##           StudRes         Hat       CookD
## 19969   1.1823087 0.193770402 0.047974412
## 26899  -2.9141422 0.017574969 0.021532227
## 44400  -0.7443964 0.136813800 0.012552808
## 46785  -3.1478639 0.005198103 0.007327689
## 47110  -3.2966300 0.005619625 0.008683319
```