Deliverable 1

Data Processing, Description, Validation and Profiling

Pere Arnau Alegre & Andrés Jiménez González

March 27, 2022

Contents

1 Data description

- Description https://www.kaggle.com/datasets/adityadesai13/used-car-dataset-ford-and-mercedes
- Data Dictionary - Scraped data of used cars, which have been separated into files corresponding to each car manufacturer (only Mercedes, BMW, Volkswagen and Audi cars are to be considered).

1.1 Variables

- Model
  - A string indicating the model of the car.
- Year
  - A discrete numeric variable to indicate the year the car was sold
- Price
  - Continuous variable indicating the price at which the car was sold
- Transmission
  - Categorical variable that indicates the type of transmission of the car
  - Values:
    * Automatic
    * Manual
    * Semi-Automatic
    * Other
- Mileage
  - A discrete numeric variable to indicate the number of miles the car had when it was sold
- Fuel Type
  - Categorical variable that indicates the type of fuel of the car
  - Values:
    * Diesel
    * Electric
    * Hybrid
    * Petrol
    * Other
- Tax
  - A discrete numeric variable to indicate the road tax of the vehicle.
- MPG
  - Continuous variable indicating the fuel consumption of the car
- Engine Size
  - Continuous variable indicating the size of the engine
- Manufacturer
  - Categorical variable that indicates the manufacturer brand of the car.
  - Values:
    * Mercedes
    * Audi
    * Volkswagen
    * BMW

2 Loading of Required Packages for the deliverable

We load the necessary packages and set the working directory

```
# setwd('C:/Users/TOREROS-II/Documents/ANDRES/UNI/ADEI/trabajo/deliverable1')
setwd("C:/Users/Arnau/Desktop/adei/deliverable1")
# Load Required Packages
options(contrasts = c("contr.treatment", "contr.treatment"))
requiredPackages <- c("missMDA", "chemometrics", "mvoutlier", "effects", "FactoMineR",
    "car", "factoextra", "RColorBrewer", "dplyr", "ggmap", "ggthemes", "knitr", "corrplot")
missingPackages <- requiredPackages[!(requiredPackages %in% installed.packages()[,
    "Package"])]
if (length(missingPackages)) install.packages(missingPackages)
lapply(requiredPackages, require, character.only = TRUE)
```

2.1 Select a sample of 5000 records

From the proposed database, we need to select a sample of 5000 records randomly so we can start analyzing our data.

```
if (!is.null(dev.list())) dev.off()  # Clear plots
rm(list = ls())  # Clean workspace
```

Data: used_car_dataset.csv

```
# filepath<-'C:/Users/TOREROS-II/Documents/ANDRES/UNI/ADEI/trabajo/deliverable1'
filepath <- "C:/Users/Arnau/Desktop/adei/deliverable1"
df <- read.table(paste0(filepath, "/sample_5000.csv"), header = T, sep = ",")[c(-1)]

# dim(df) # Displays the sample size names(df) # Displays the names of the
# sample variables summary(df)
```

tinytex::install_tinytex() ## Some useful functions

```
calcQ <- function(x) {
    # Function to calculate the different quartiles
    s.x <- summary(x)
    iqr <- s.x[5] - s.x[2]
    list(souti = s.x[2] - 3 * iqr, mouti = s.x[2] - 1.5 * iqr, min = s.x[1], q1 = s.x[2],
        q2 = s.x[3], q3 = s.x[5], max = s.x[6], mouts = s.x[5] + 1.5 * iqr, souts = s.x[5] +
            3 * iqr)
}
countNA <- function(x) {
    # Function to count the NA values
    mis_x <- NULL
    for (j in 1:ncol(x)) {
        mis_x[j] <- sum(is.na(x[, j]))
    }
    mis_x <- as.data.frame(mis_x)
    rownames(mis_x) <- names(x)
    mis_i <- rep(0, nrow(x))
    for (j in 1:ncol(x)) {
        mis_i <- mis_i + as.numeric(is.na(x[, j]))
    }
    list(mis_col = mis_x, mis_ind = mis_i)
}
countX <- function(x, X) {
    # Function to count a specific number of appearences
    n_x <- NULL
    for (j in 1:ncol(x)) {
        n_x[j] <- sum(x[, j] == X)
```

```
    }
    n_x <- as.data.frame(n_x)
    rownames(n_x) <- names(x)
    nx_i <- rep(0, nrow(x))
    for (j in 1:ncol(x)) {
        nx_i <- nx_i + as.numeric(x[, j] == X)
    }
    list(nx_col = n_x, nx_ind = nx_i)
}
```

3 Univariate Description and Preprocessing

3.1 Variable initialization of missings, outliers and errors for columns

```
jmis <- rep(0, 2 * ncol(df))  # columns - variables

mis1 <- countNA(df)
# mis1$mis_ind # Number of missings for the current set of cars (observations)
# mis1$mis_col # Number of missings for the current set of variables

jouts <- rep(0, ncol(df))  # columns - variables

jerrs <- rep(0, ncol(df))  # columns - variables
```

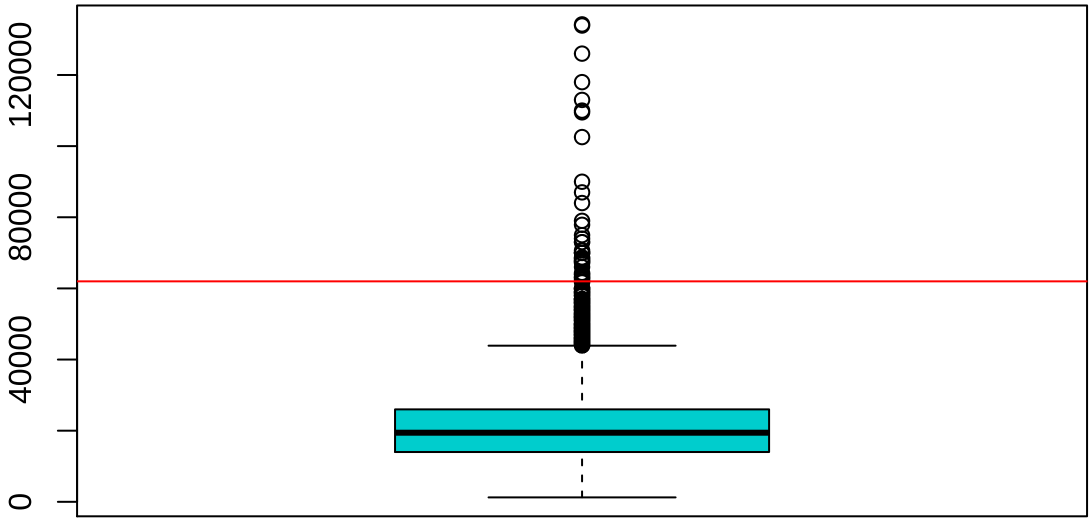
3.2 Initialization of the response variable Price

We know that the price should be positive, so we will treat as errors the prices <= 0. We don't count the errors by rows for the variable price because we erase that rows.

```
summary(df$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1250   14000   19430   21419   25995  134219
```

```
sel <- which(df$price <= 0)
jerrs[which(colnames(df) == "price")] <- length(sel)
# We will delete the rows with errors in the price because we cannot make
# imputations for our target variable.
df <- df[which(df$price > 0), ]

boxplot(df$price, col = "cyan3")
var_out <- calcQ(df$price)
abline(h = var_out$souts, col = "red")
abline(h = var_out$souti, col = "red")
```




```
# We can see there are outliers in the dataset so we will treat them. As this
# is the response variable, we will delete the outlier rows because we cannot
# delete the value and impute it.
llout_price <- which((df$price > var_out$souts) | (df$price < var_out$souti))
# iouts[llout] <- iouts[llout]+1
jouts[which(colnames(df) == "price")] <- length(llout_price)
df <- df[-llout_price, ]
```

3.3 Variable initialization of missings, outliers and errors for rows

Initialization of counts for missings, outliers and errors. All numerical variables have to be checked before.

```
imis <- rep(0, nrow(df))  # rows - cars

iouts <- rep(0, nrow(df))  # rows - cars

ierrs <- rep(0, nrow(df))  # rows - cars
```

3.4 Preprocessing of Qualitative/Categorical & Numerical variables

Description: We need to do an analysis of all the variables to be able to identify missings, errors and outliers. We will also try to factorize each variable to make it easier to understand the sample.

3.4.1 Model

This variable indicates the model of the car.

```
df$model <- factor(paste0(df$manufacturer, "-", df$model))
# levels(df$model)
summary(df$model)
```

```
##           Audi- A1            Audi- A3            Audi- A4            Audi- A5 
##                130                 196                 143                  84 
##           Audi- A6            Audi- A7            Audi- A8            Audi- Q2 
##                 89                   8                  12                  74 
##           Audi- Q3            Audi- Q5            Audi- Q7            Audi- Q8 
##                155                  94                  41                   5 
##           Audi- R8           Audi- RS3           Audi- RS4           Audi- RS5 
##                  2                   3                   1                   1 
##          Audi- RS6            Audi- S3            Audi- S4            Audi- S8 
##                  5                   1                   1                   1 
##          Audi- SQ5           Audi- SQ7            Audi- TT       BMW- 1 Series 
##                  2                   2                  28                 190 
##      BMW- 2 Series       BMW- 3 Series       BMW- 4 Series       BMW- 5 Series 
##                129                 251                 113                  94 
##      BMW- 6 Series       BMW- 7 Series       BMW- 8 Series            BMW- i3 
##                 16                  12                   1                   5 
##            BMW- M3             BMW- M4             BMW- M5            BMW- X1 
##                  2                  14                   1                  81 
##            BMW- X2             BMW- X3             BMW- X4            BMW- X5 
##                 30                  50                  21                  41 
##            BMW- X6             BMW- Z3             BMW- Z4   Mercedes- A Class 
##                  5                   2                   8                 262 
##  Mercedes- B Class   Mercedes- C Class  Mercedes- CL Class Mercedes- CLA Class 
##                 60                 394                  57                   7 
## Mercedes- CLS Class   Mercedes- E Class  Mercedes- GL Class Mercedes- GLA Class 
##                 25                 199                  12                  69 
## Mercedes- GLB Class Mercedes- GLC Class Mercedes- GLE Class Mercedes- GLS Class 
##                  1                  77                  39                   6 
##  Mercedes- M Class   Mercedes- S Class  Mercedes- SL CLASS      Mercedes- SLK 
##                  9                  20                  29                  10 
##  Mercedes- V Class   Mercedes- X-CLASS        Mercedes-180         VW- Amarok 
##                 23                  10                   1                  10 
##         VW- Arteon          VW- Beetle           VW- Caddy     VW- Caddy Life 
##                 25                   4                   1                   1 
## VW- Caddy Maxi Life      VW- California      VW- Caravelle             VW- CC 
##                  4                   2                   9                   8 
##            VW- Eos             VW- Fox            VW- Golf        VW- Golf SV 
##                  1                   1                 488                  21 
##          VW- Jetta          VW- Passat            VW- Polo       VW- Scirocco 
##                  1                  89                 330                  27 
##         VW- Sharan         VW- Shuttle         VW- T-Cross          VW- T-Roc 
##                 25                   6                  22                  64 
##         VW- Tiguan VW- Tiguan Allspace         VW- Touareg         VW- Touran 
##                184                  12                  39                  32 
##             VW- Up 
##                100
```

```
# Too many models to represent them in a graph The is not missing data or
# erroneous data, so we will not make any change in the model column
```

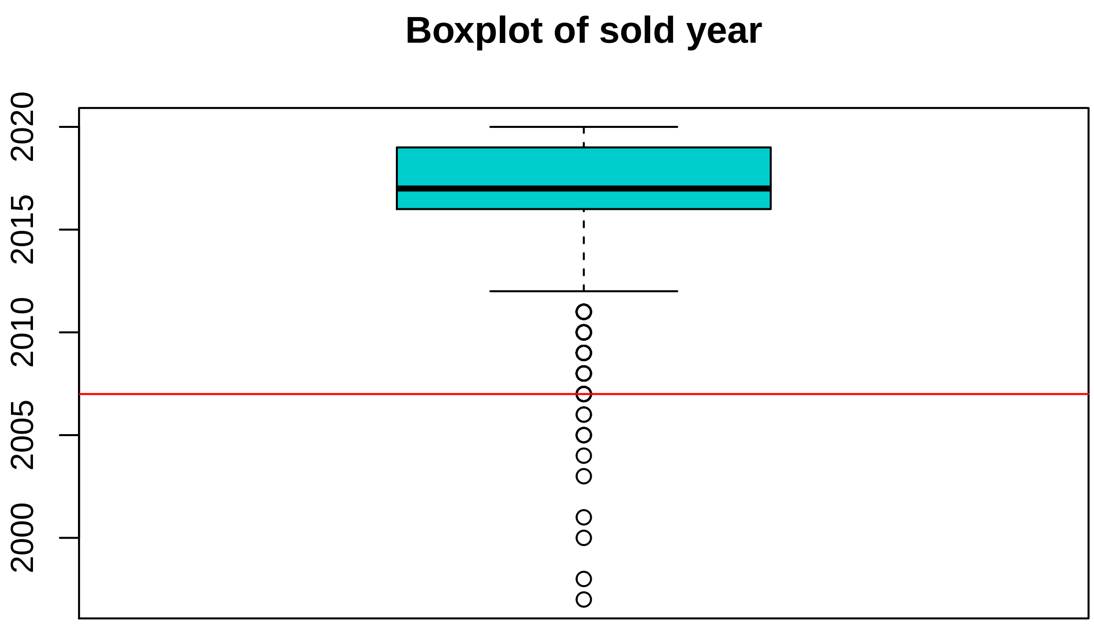
3.4.2 Year

A discrete numeric variable to indicate the year the car was sold, ranging from 1970 to 2020

```
boxplot(df$year, main = "Boxplot of sold year", col = "cyan3")
# df$year <- factor(df$year) We can see that there are outliers in the dataset,
# so we will treat them.
summary(df$year)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    1997    2016    2017    2017    2019    2020
```

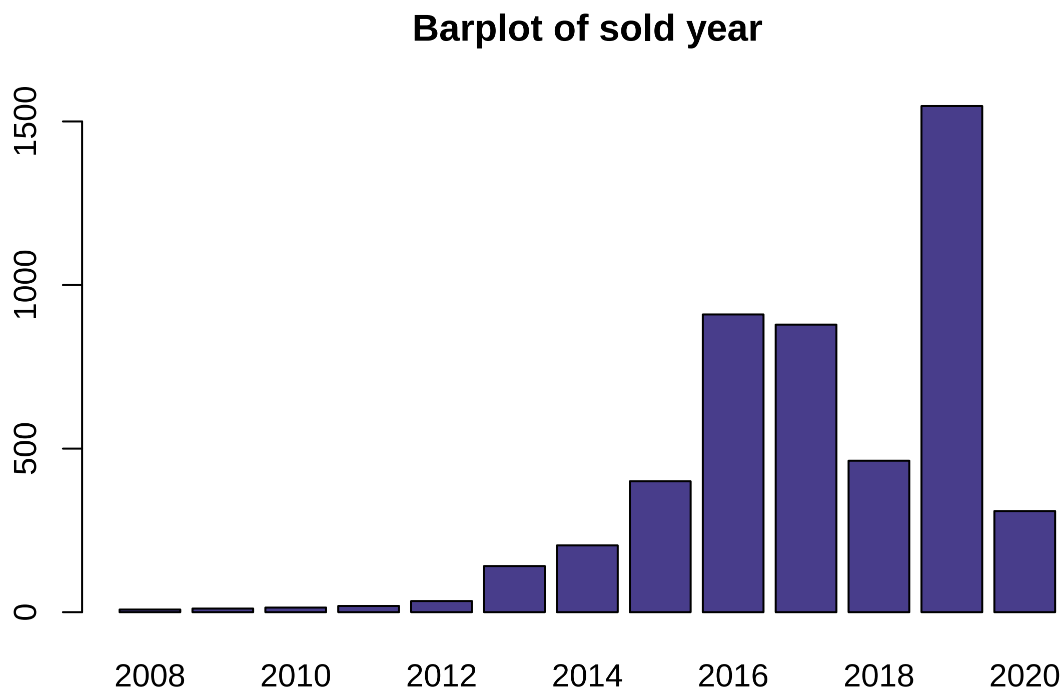
```
var_out <- calcQ(df$year)
abline(h = var_out$souts, col = "red")
abline(h = var_out$souti, col = "red")
```




```
llout <- which((df$year <= var_out$souti))
iouts[llout] <- iouts[llout] + 1
jouts[which(colnames(df) == "year")] <- length(llout)

# We will group all the inferior outliers into one variable
df[llout, "year"] <- NA
summary(df$year)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    2008    2016    2017    2017    2019    2020      21
```

```
# df[which(df$year<=var_out$souti),'year'] <- paste0(var_out$souti, ' or
# before')
barplot(table(df$year), main = "Barplot of sold year", col = "darkslateblue")
```



In orther to better analyze the price of the cars and to group them, we will create a categorical variable representing the price of the car.

```
df$price_type <- df$price
df$price_type[which(df$price >= var_out$min & df$price_type < var_out$q1)] <- "super cheap"
df$price_type[which(df$price >= var_out$q1 & df$price_type < var_out$q2)] <- "cheap"
df$price_type[which(df$price >= var_out$q2 & df$price_type < var_out$q3)] <- "expensive"
df$price_type[which(df$price >= var_out$q3 & df$price_type < var_out$mouts)] <- "very expensive"
df$price_type[which(df$price >= var_out$mouts)] <- "extremely expensive"
table(df$price_type)
```

```
##
##                1250                1450                1490                1990
##                   1                   1                   1                   1
##                1995 extremely expensive         super cheap
##                   1                4954                   1
```

3.4.3 Transmission

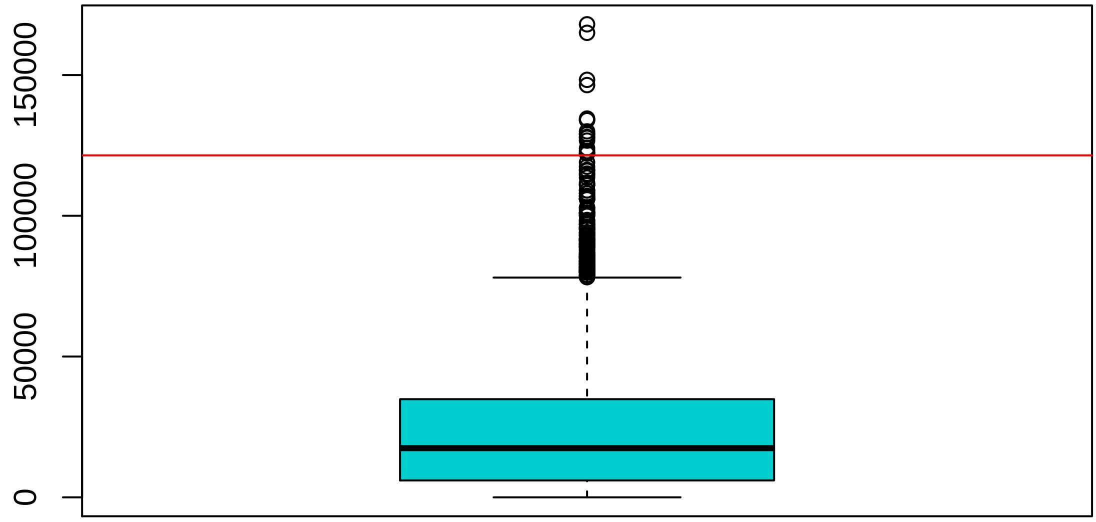
```
df$transmission <- factor(df$transmission)
levels(df$transmission)
```

```
## [1] "Automatic" "Manual"    "Other"     "Semi-Auto"
```

```
df$transmission <- factor(df$transmission, levels = c("Manual", "Semi-Auto", "Automatic"),
    labels = paste0("f.Trans-", c("Manual", "SemiAuto", "Automatic")))
# All transmission not listed above have been replaced as NA
```

3.4.4 Mileage

```r
boxplot(df$mileage, col = "cyan3")
var_out <- calcQ(df$mileage)
abline(h = var_out$souts, col = "red")
abline(h = var_out$souti, col = "red")
```




```r
llout_mil <- which((df$mileage < var_out$souti) | (df$mileage > var_out$souts))
iouts[llout_mil] <- iouts[llout_mil] + 1
df[llout_mil, "mileage"] <- NA
```

3.4.5 fuelType

Andres

```r
df$fuelType <- factor(df$fuelType)
levels(df$fuelType)
```

```
## [1] "Diesel" "Hybrid" "Other"  "Petrol"
```

```r
df$fuelType <- factor(df$fuelType, levels = c("Diesel", "Petrol", "Hybrid"), labels = paste0("f.Fuel-",
    c("Diesel", "Petrol", "Hybrid")))
# All fuelTypes not listed above have been replaced as NA
```

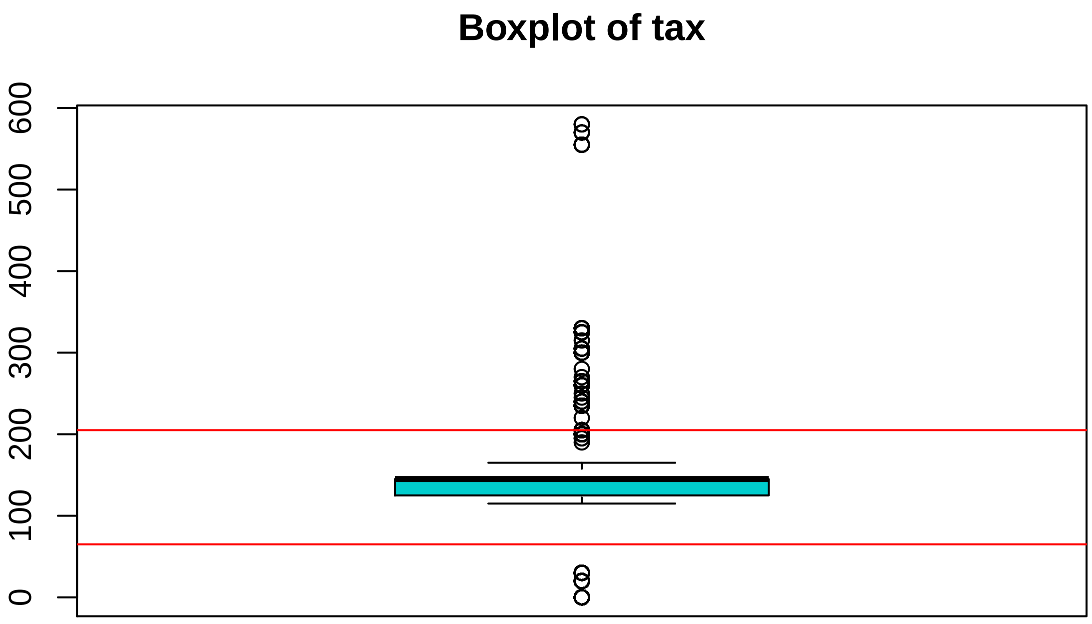
3.4.6 Tax

Andres

```r
boxplot(df$tax, main = "Boxplot of tax", col = "cyan3")
# df$year <- factor(df$year) We can see that there are outliers in the dataset,
# so we will treat them.
summary(df$tax)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   125.0   145.0   122.7   145.0   580.0
```

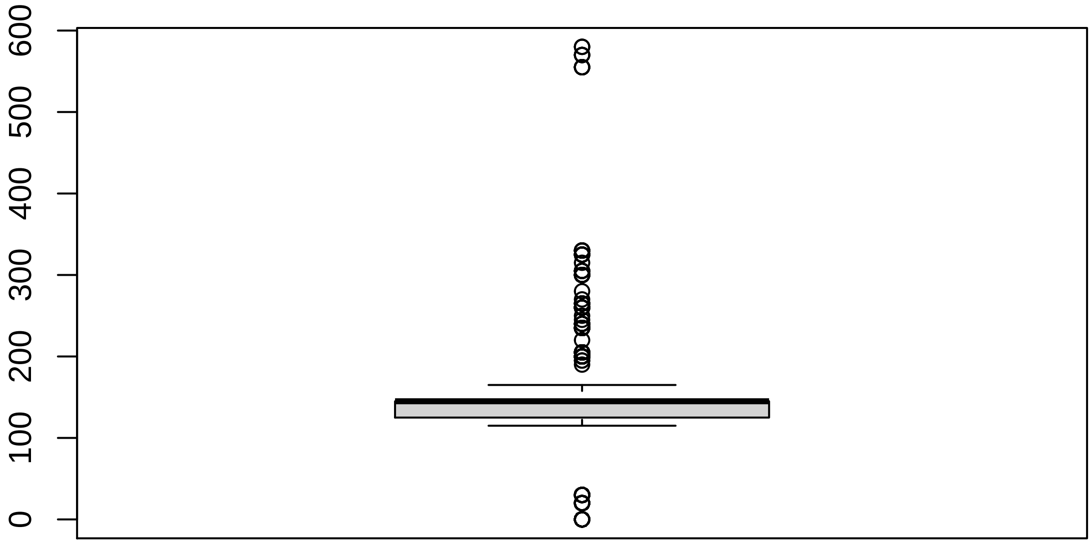
```
var_out <- calcQ(df$tax)
abline(h = var_out$souts, col = "red")
abline(h = var_out$souti, col = "red")
```




```
llout <- which((df$tax <= var_out$souti & df$tax >= var_out$souts))
iouts[llout] <- iouts[llout] + 1
jouts[which(colnames(df) == "tax")] <- length(llout)
df[llout, "tax"] <- NA

summary(df$tax)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   125.0   145.0   122.7   145.0   580.0
```

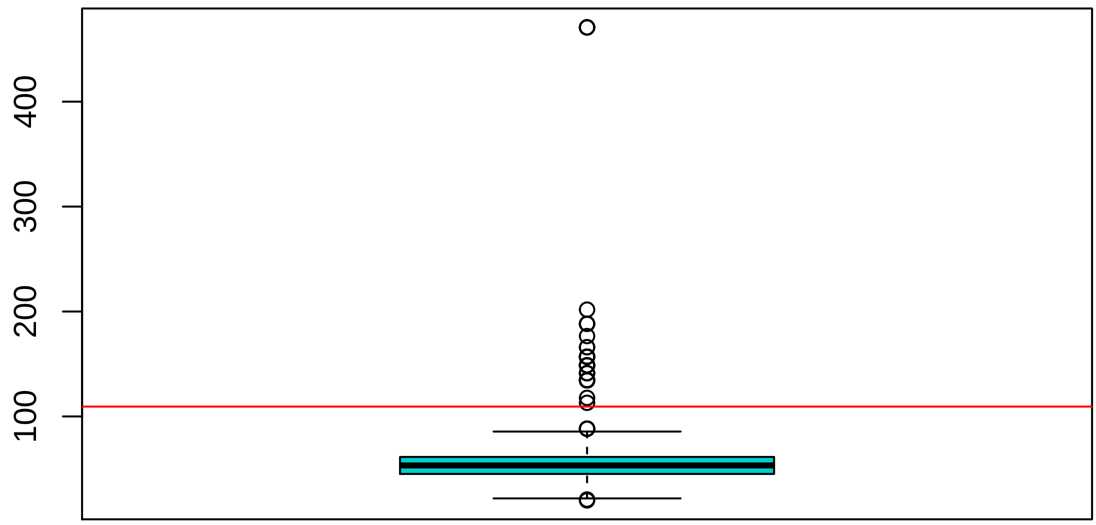
```
boxplot(df$tax)
```



3.4.7 MPG

Andres

```
# Outliers are replaced by NA
boxplot(df$mpg, col = "cyan3")
var_out <- calcQ(df$mpg)
abline(h = var_out$souts, col = "red")
abline(h = var_out$souti, col = "red")
```




```
llout_mpg <- which((df$mpg < var_out$souti) | (df$mpg > var_out$souts))
iouts[llout_mpg] <- iouts[llout_mpg] + 1
jouts[which(colnames(df) == "mpg")] <- length(llout)
df[llout_mpg, "mpg"] <- NA
```

3.4.8 EngineSyze

Andres –> contabilizar errores

```
df$engineSize <- factor(df$engineSize)
levels(df$engineSize)
```

```
##  [1] "0"   "1"   "1.2" "1.3" "1.4" "1.5" "1.6" "1.8" "1.9" "2"   "2.1" "2.2"
## [13] "2.3" "2.5" "2.9" "3"   "3.2" "3.5" "3.7" "4"   "4.1" "4.2" "4.4" "4.7"
## [25] "5.2" "5.5" "6.2" "6.6"
```

```
df[which(df[, "engineSize"] == 0), ]
```

```
##                  model year price      transmission mileage      fuelType tax
## 777           Audi- Q3 2020 33333 f.Trans-Automatic    1500 f.Fuel-Diesel 145
## 789           Audi- Q2 2020 24990    f.Trans-Manual    1500 f.Fuel-Petrol 145
## 795          Audi- SQ5 2020 56450 f.Trans-Automatic    1500 f.Fuel-Diesel 145
## 796           Audi- Q3 2020 33990 f.Trans-Automatic    4000 f.Fuel-Diesel 145
## 812           Audi- Q3 2017 19300    f.Trans-Manual   16051 f.Fuel-Diesel 150
## 815           Audi- TT 2016 22500 f.Trans-Automatic   45182 f.Fuel-Petrol 200
## 821           Audi- Q3 2020 32000 f.Trans-Automatic    1500 f.Fuel-Petrol 145
## 1356           BMW- i3 2016 19490 f.Trans-Automatic    8421 f.Fuel-Hybrid   0
## 1450           BMW- i3 2016 16482 f.Trans-Automatic   43695 f.Fuel-Hybrid   0
## 1687           BMW- i3 2014 14182 f.Trans-Automatic   37161 f.Fuel-Hybrid   0
## 1710           BMW- i3 2017 23751 f.Trans-Automatic   28169 f.Fuel-Hybrid   0
## 1803           BMW- i3 2017 19948 f.Trans-Automatic   20929 f.Fuel-Hybrid 135
## 3144 Mercedes- A Class 2016 17800 f.Trans-Automatic   21913 f.Fuel-Diesel  20
## 3302 Mercedes- E Class 2018 22738 f.Trans-Automatic   24000 f.Fuel-Diesel 150
```

```
## 3552         VW- T-Roc 2019 22000 f.Trans-Automatic    2009 f.Fuel-Petrol 145
## 3965          VW- Golf 2017 12600    f.Trans-Manual   20340 f.Fuel-Diesel   0
##       mpg engineSize manufacturer           price_type
## 777  47.1          0         Audi extremely expensive
## 789  43.5          0         Audi extremely expensive
## 795  34.5          0         Audi extremely expensive
## 796  47.1          0         Audi extremely expensive
## 812  52.3          0         Audi extremely expensive
## 815  40.9          0         Audi extremely expensive
## 821  31.4          0         Audi extremely expensive
## 1356   NA          0          BMW extremely expensive
## 1450   NA          0          BMW extremely expensive
## 1687   NA          0          BMW extremely expensive
## 1710   NA          0          BMW extremely expensive
## 1803   NA          0          BMW extremely expensive
## 3144 68.9          0     Mercedes extremely expensive
## 3302 61.4          0     Mercedes extremely expensive
## 3552 39.8          0           VW extremely expensive
## 3965 74.3          0           VW extremely expensive
```

```
# It is a quantitive variable Non-possible values will be recoded to NA
sel <- which(df$engineSize == 0)
ierrs[sel] <- ierrs[sel] + 1  #Vector of errors per individual update
sel  #### sel contains the rownames of the individuals with '0'
```

```
##  [1]  766  778  784  785  801  804  810 1340 1431 1665 1687 1775 3104 3262 3512
## [16] 3925
```

```
# as value for engineSize We should update jerrs vector: errors per variable

# df[sel,'engineSize']<-3 # non-possible values are replaced by NA, missing
# value symbol in R NA assignment for forward imputation:
df[sel, "engineSize"] <- NA
```

#Imputation What we do with imputation is be able to eliminate all those values that may be missings, outliers or errors to turn them into values that can be realistic within our sample.

3.5 Imputation of numeric variables

```
library(missMDA)
# Now one by one describe vars and put them on lists
vars_con <- c("year", "mileage", "tax", "mpg")
vars_res <- c("price")

summary(df[, vars_con])
```

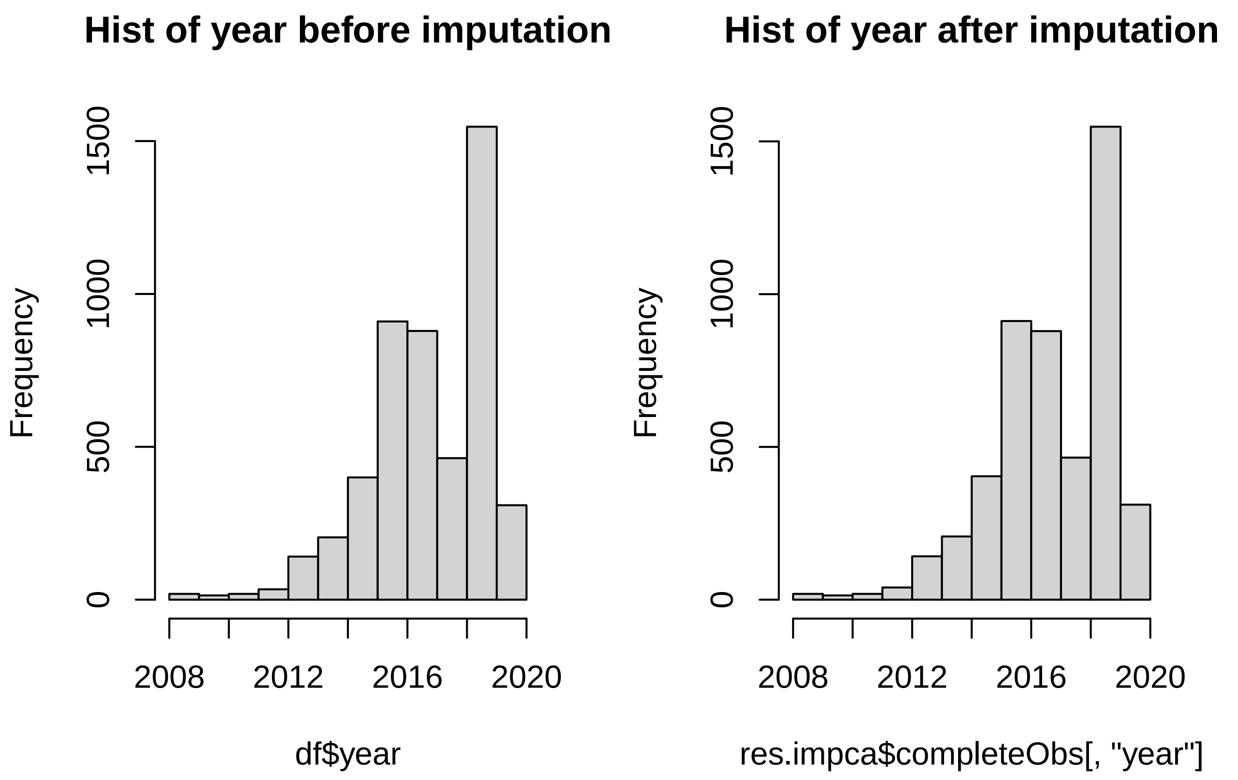
```
##       year         mileage             tax             mpg
##  Min.   :2008   Min.   :     1   Min.   :  0.0   Min.   :20.00
##  1st Qu.:2016   1st Qu.:  6000   1st Qu.:125.0   1st Qu.:44.80
##  Median :2017   Median : 17371   Median :145.0   Median :53.30
##  Mean   :2017   Mean   : 23379   Mean   :122.7   Mean   :53.14
##  3rd Qu.:2019   3rd Qu.: 34593   3rd Qu.:145.0   3rd Qu.:61.40
##  Max.   :2020   Max.   :119000   Max.   :580.0   Max.   :88.30
##  NA's   :21     NA's   :17                       NA's   :54
```

```
res.impca <- imputePCA(df[, vars_con], ncp = 3)
summary(res.impca$completeObs)
```

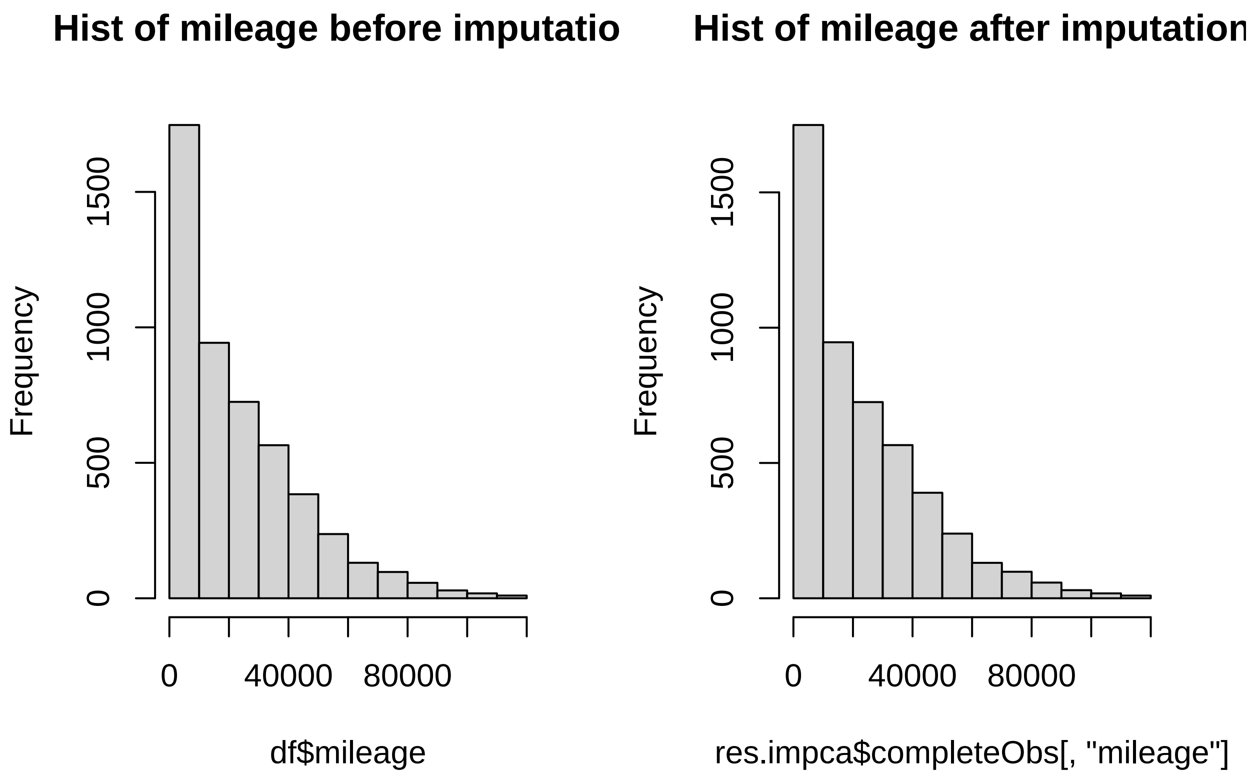
```
##       year         mileage             tax             mpg
```

```
##  Min.   :2008   Min.   :      1   Min.   :  0.0   Min.   :20.00
##  1st Qu.:2016   1st Qu.:   6000   1st Qu.:125.0   1st Qu.:45.40
##  Median :2017   Median : 17415   Median :145.0   Median :53.30
##  Mean   :2017   Mean   : 23441   Mean   :122.7   Mean   :53.21
##  3rd Qu.:2019   3rd Qu.: 34768   3rd Qu.:145.0   3rd Qu.:61.40
##  Max.   :2020   Max.   :119000   Max.   :580.0   Max.   :88.30
```

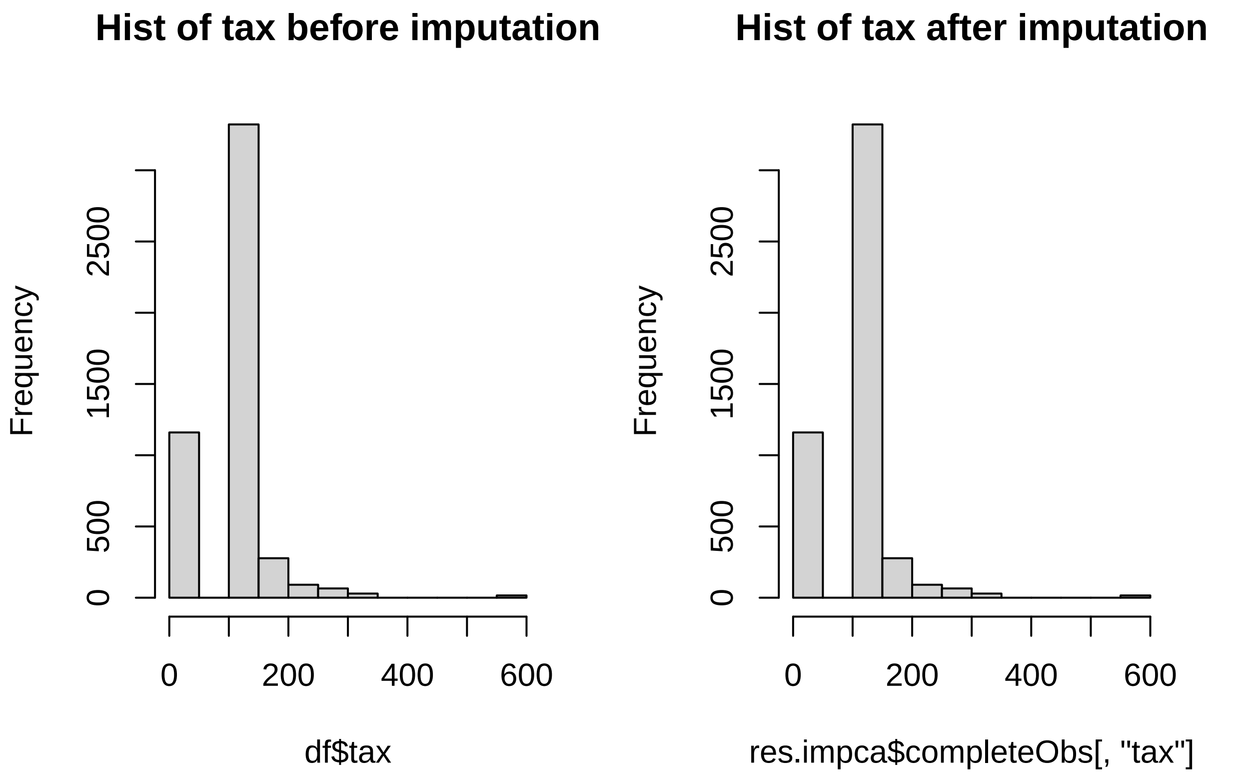
```
# Check one by one:
par(mfrow = c(1, 2))
hist(df$year, main = "Hist of year before imputation")
hist(res.impca$completeObs[, "year"], main = "Hist of year after imputation")
```



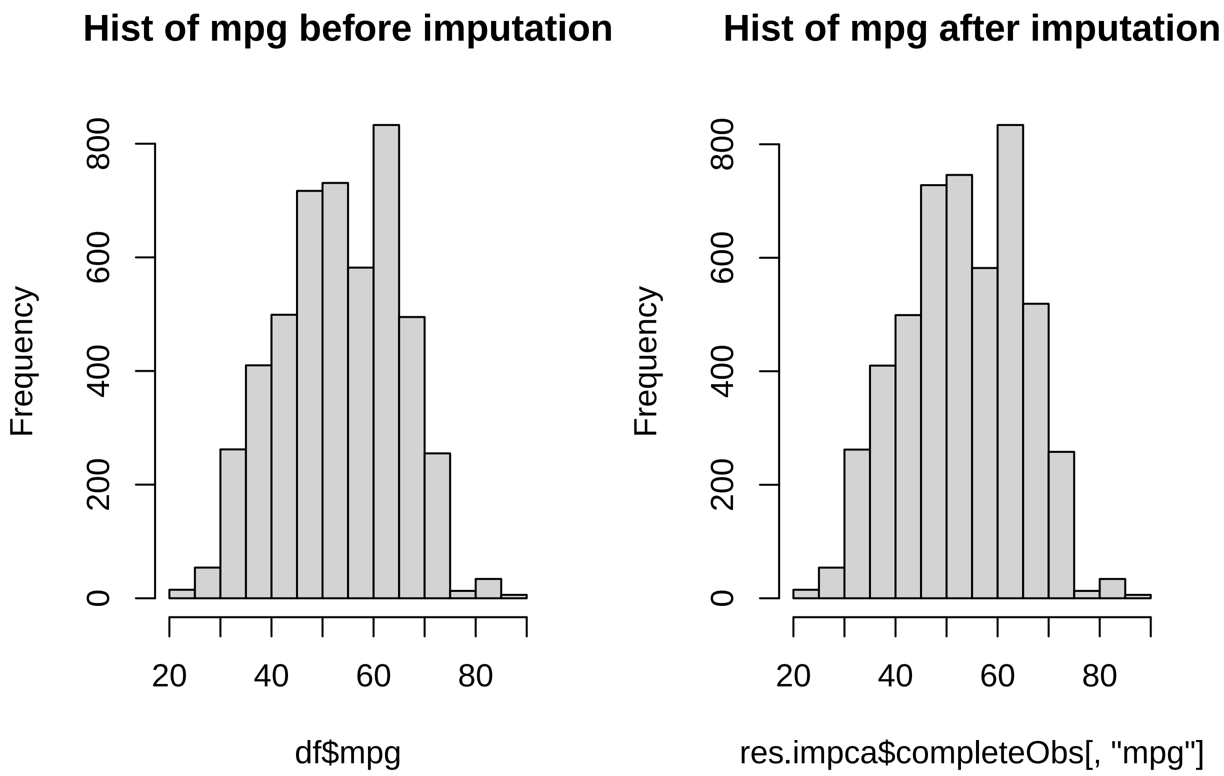

```
hist(df$mileage, main = "Hist of mileage before imputation")
hist(res.impca$completeObs[, "mileage"], main = "Hist of mileage after imputation")
```




```r
hist(df$tax, main = "Hist of tax before imputation")
hist(res.impca$completeObs[, "tax"], main = "Hist of tax after imputation")
```




```r
hist(df$mpg, main = "Hist of mpg before imputation")
hist(res.impca$completeObs[, "mpg"], main = "Hist of mpg after imputation")
```




```
# Once you have validated the process:
df[, vars_con] <- res.impca$completeObs
```

3.6 Imputation of qualitative variables

```
vars_dis <- c("model", "transmission", "fuelType", "engineSize", "manufacturer")
summary(df[, vars_dis])
```

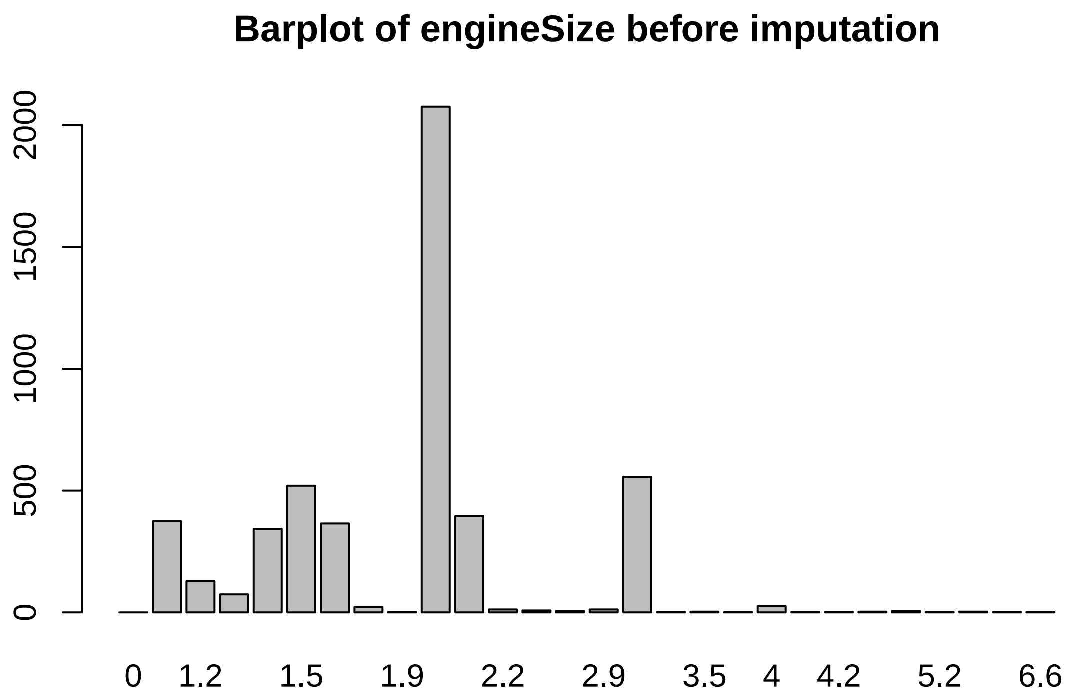
```
##                 model                  transmission             fuelType
##  VW- Golf          : 488   f.Trans-Manual    :1741   f.Fuel-Diesel:2837
##  Mercedes- C Class: 394   f.Trans-SemiAuto  :1906   f.Fuel-Petrol:2044
##  VW- Polo          : 330   f.Trans-Automatic:1312   f.Fuel-Hybrid:   66
##  Mercedes- A Class: 262   NA's              :    1   NA's         :   13
##  BMW- 3 Series     : 251
##  Mercedes- E Class: 199
##  (Other)           :3036
##    engineSize   manufacturer
##  2       :2076   Length:4960
##  3       : 556   Class :character
##  1.5     : 520   Mode  :character
##  2.1     : 395
##  1       : 374
##  (Other):1023
##  NA's    :  16
```

```
res.immca <- imputeMCA(df[, vars_dis], ncp = 4)
summary(res.immca$completeObs)
```

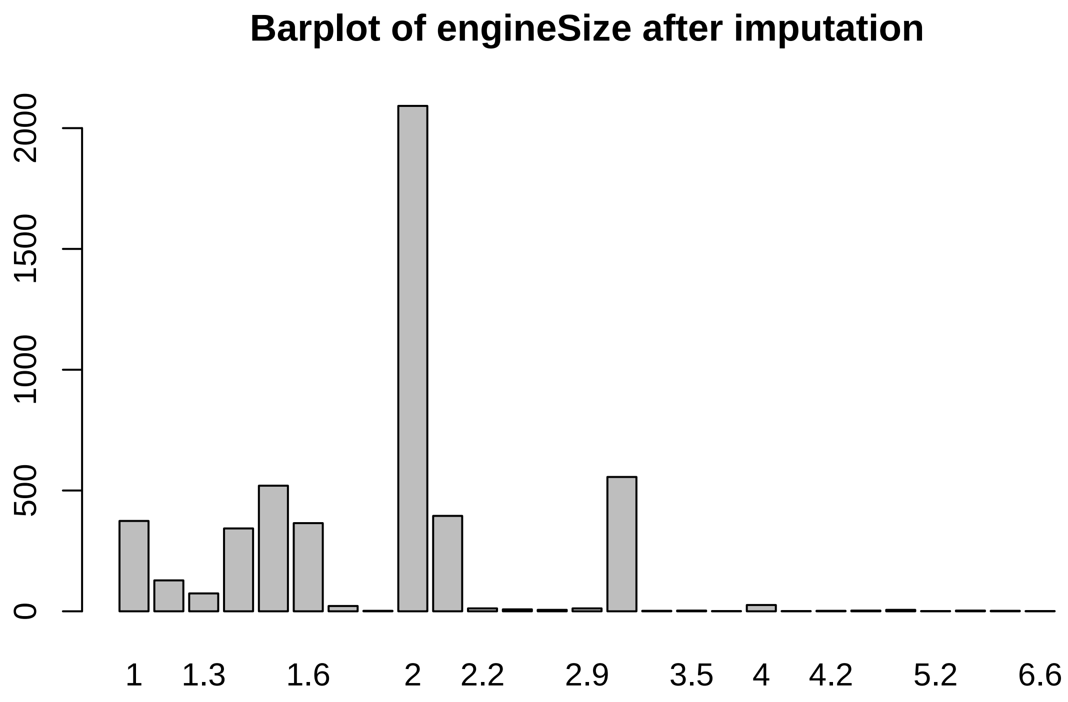
```
##                 model                  transmission             fuelType
##  VW- Golf          : 488   f.Trans-Manual    :1741   f.Fuel-Diesel:2846
##  Mercedes- C Class: 394   f.Trans-SemiAuto  :1907   f.Fuel-Petrol:2048
##  VW- Polo          : 330   f.Trans-Automatic:1312   f.Fuel-Hybrid:   66
```

```
##  Mercedes- A Class: 262
##  BMW- 3 Series    : 251
##  Mercedes- E Class: 199
##  (Other)          :3036
##    engineSize     manufacturer
##  2      :2092   Audi    :1078
##  3      : 556   BMW     :1066
##  1.5    : 520   Mercedes:1310
##  2.1    : 395   VW      :1506
##  1      : 374
##  1.6    : 365
##  (Other): 658
```

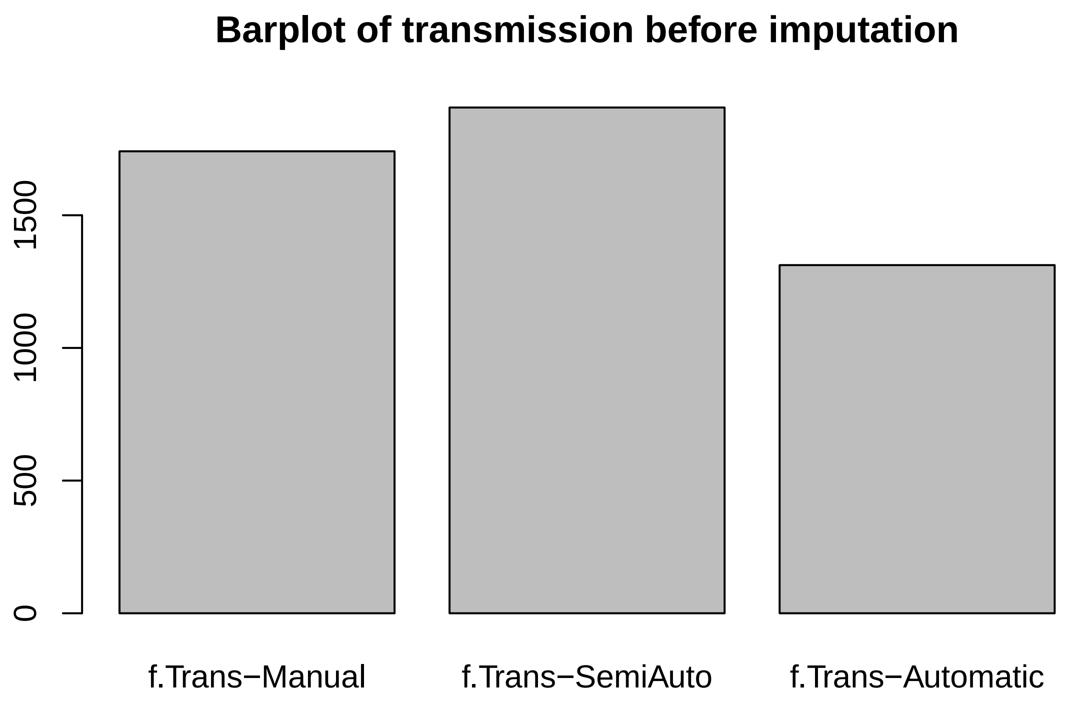
```r
# Check one by one (we only have enginesize, transmission & fuelType)
barplot(table(df$engineSize), main = "Barplot of engineSize before imputation")
```



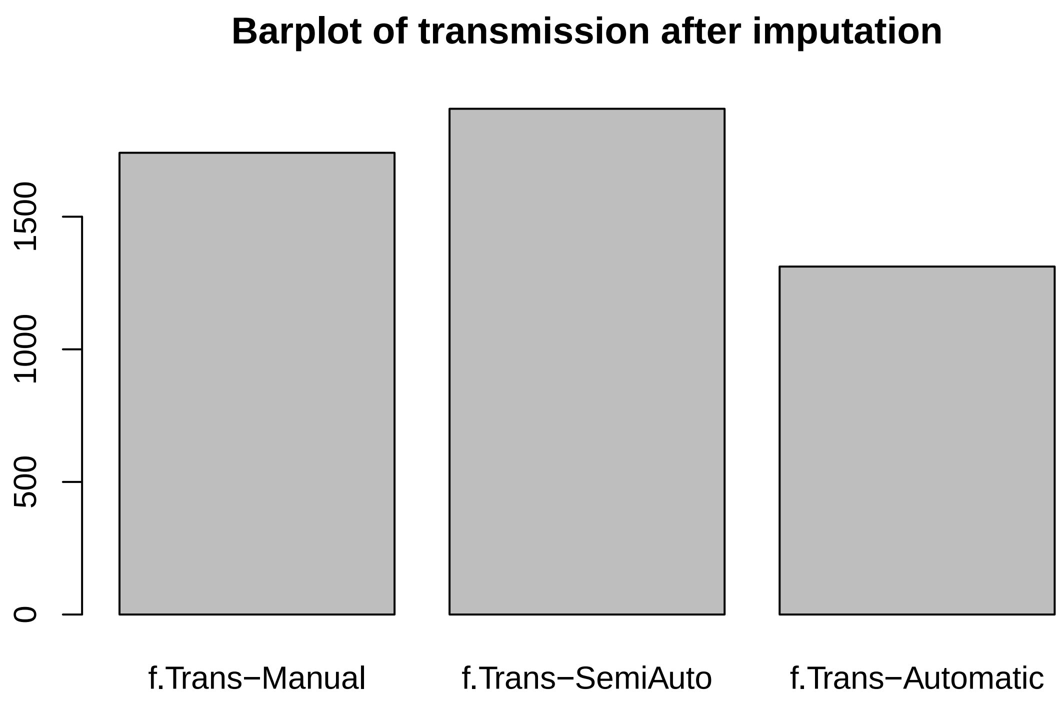

```r
barplot(table(res.immca$completeObs[, "engineSize"]), main = "Barplot of engineSize after imputation")
```



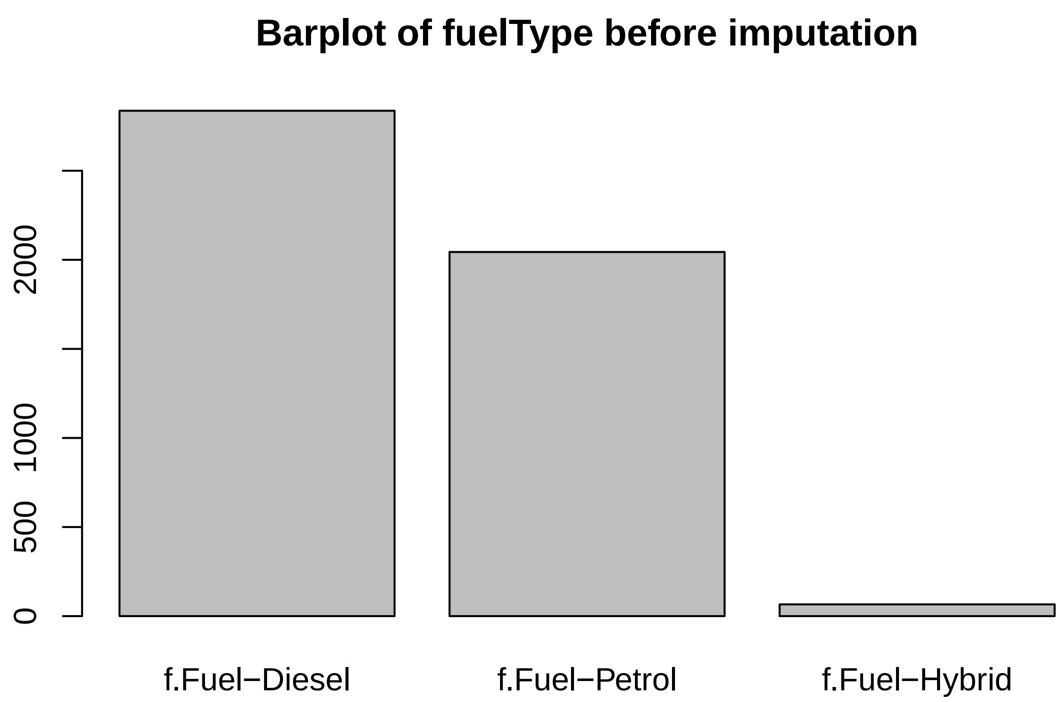

```r
barplot(table(df$transmission), main = "Barplot of transmission before imputation")
```



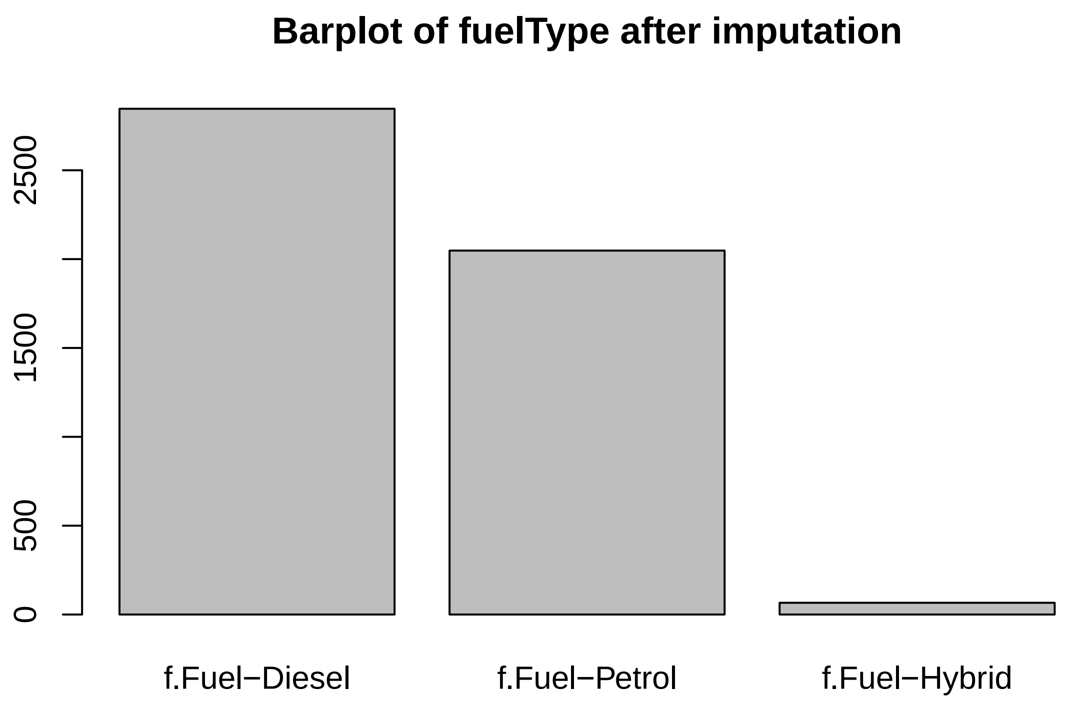
```
barplot(table(res.immca$completeObs[, "transmission"]), main = "Barplot of transmission after imputatio
```




```
barplot(table(df$fuelType), main = "Barplot of fuelType before imputation")
```



```r
barplot(table(res.immca$completeObs[, "fuelType"]), main = "Barplot of fuelType after imputation")
```




```r
# Once you have validated the process
df[, vars_dis] <- res.immca$completeObs

# Are there NA?
sum(countNA(df)$mis_ind) == 0
```

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

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

4 Creation and discretization of new variables

4.1 New variable: Audi/Not Audi

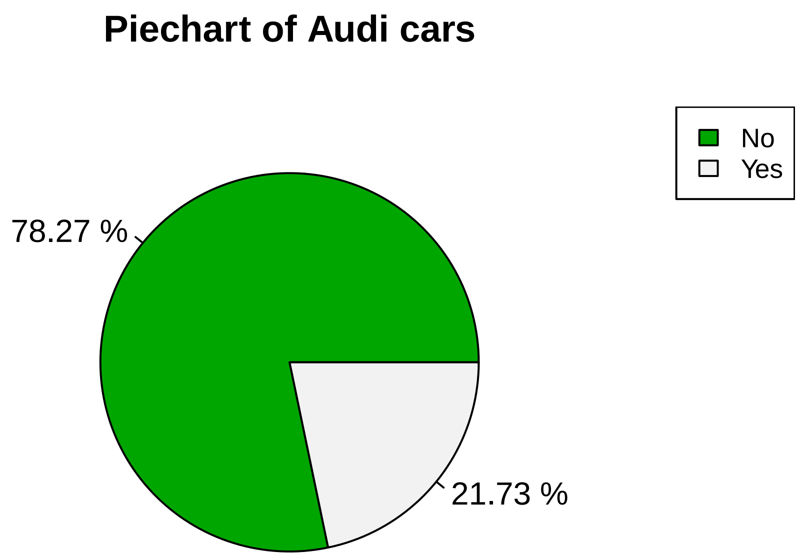
```r
# Binary Target: Audi?

df$Audi <- ifelse(df$manufacturer == "Audi", 1, 0)
df$Audi <- factor(df$Audi, labels = c("No", "Yes"))
summary(df$Audi)
```

```
##   No  Yes
## 3882 1078
```

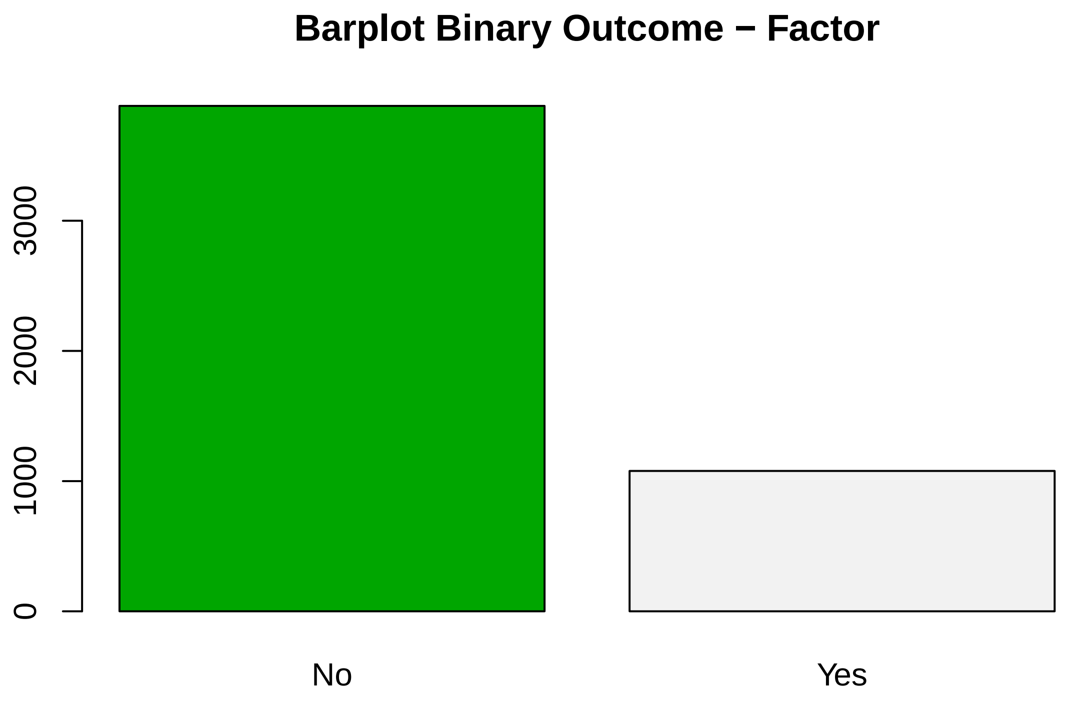
```r
# Pie
piepercent <- round(100 * (table(df$Audi)/nrow(df)), dig = 2)
piepercent
```

```
##
##    No   Yes
## 78.27 21.73
```

```r
pie(table(df$Audi), col = terrain.colors(2), labels = paste(piepercent, "%"), main = "Piechart of Audi c
legend("topright", levels(df$Audi), cex = 0.8, fill = terrain.colors(2))
```



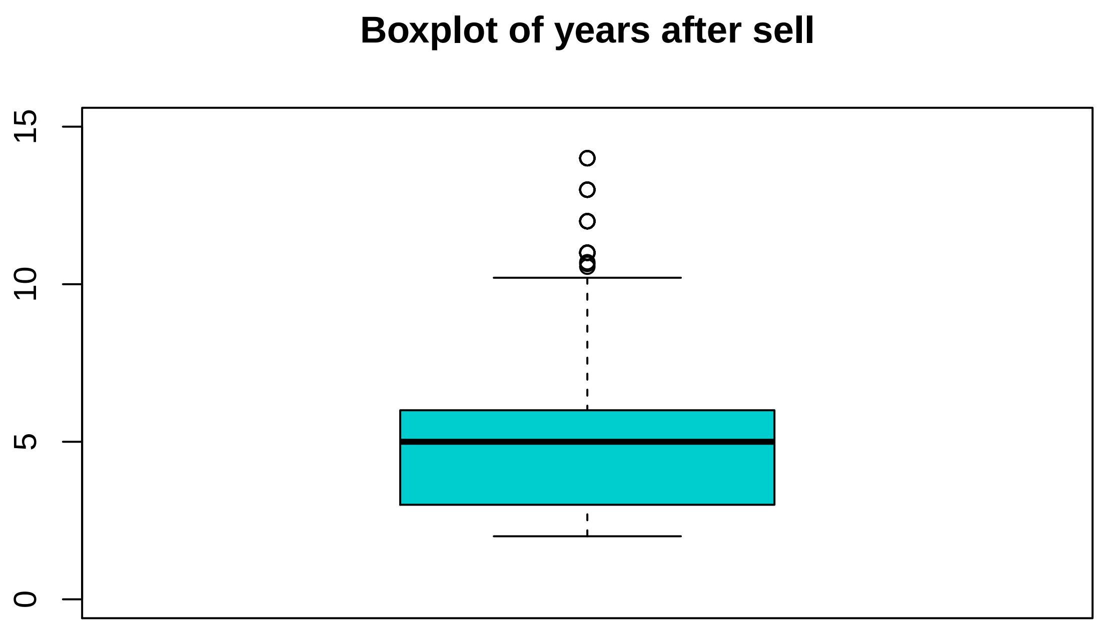

```r
# Bar Chart
barplot(table(df$Audi), main = "Barplot Binary Outcome - Factor", col = terrain.colors(2))
```



4.2 New variable: yearsAferSell

A discrete numeric variable to indicate how many years have passed from when the car was sold since 2022.

```
df$years_after_sell <- 2022 - df$year
boxplot(df$years_after_sell, main = "Boxplot of years after sell", col = "cyan3",
    ylim = c(0, 15))
```



```
# There are no extreme outliers in the variable because we treated outliers in
# the variable year.
summary(df$years_after_sell)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   3.000   5.000   4.798   6.000  14.000
```

4.3 Discretization of the variable Tax

```
quantile(df$tax, seq(0, 1, 0.25), na.rm = TRUE)
```

```
##   0%  25%  50%  75% 100%
##    0  125  145  145  580
```

```
quantile(df$tax, seq(0, 1, 0.1), na.rm = TRUE)
```

```
##   0%  10%  20%  30%  40%  50%  60%  70%  80%  90% 100%
##    0   20   30  145  145  145  145  145  145  150  580
```

```
quants <- calcQ(df$tax)

# df$aux<-factor(cut(df$tax,breaks=quantile(df$tax,seq(0,1,0.25),na.rm=TRUE),include.lowest
# = T )) # Does not work Reconsiderations of limits bc mean and 3rd quantile
# are the same aux<-factor(cut(df$tax,breaks=c(0, 125, 145,
# quants),include.lowest = T )) summary(aux) tapply(df$tax,aux,median)
df$f.tax <- factor(cut(df$tax, breaks = c(quants$min, quants$q1, quants$q2, quants$q3 +
    10, quants$max), include.lowest = T))
levels(df$f.tax) <- paste("f.tax-", levels(df$f.tax), sep = "")
table(df$f.tax, useNA = "always")
```

```
##
##   f.tax-[0,125] f.tax-(125,145] f.tax-(145,155] f.tax-(155,580]            <NA>
##            1447            2537             499             477               0
```

4.4 Discretization of the variable mileage

```
df$f.mileage <- factor(cut(df$mileage, breaks = c(quantile(df$mileage, seq(0, 1,
    0.25), na.rm = TRUE)), include.lowest = T))
levels(df$f.mileage) <- paste("f.mileage-", levels(df$f.mileage), sep = "")
table(df$f.mileage, useNA = "always")
```

```
##
##           f.mileage-[1,6e+03]    f.mileage-(6e+03,1.74e+04]
##                          1253                          1227
## f.mileage-(1.74e+04,3.48e+04] f.mileage-(3.48e+04,1.19e+05]
##                          1240                          1240
##                          <NA>
##                             0
```

4.5 Discretization of the variable mpg

```
df$f.mpg <- factor(cut(df$mpg, breaks = c(quantile(df$mpg, seq(0, 1, 0.25), na.rm = TRUE)),
    include.lowest = T))
levels(df$f.mpg) <- paste("f.mpg-", levels(df$f.mpg), sep = "")
table(df$f.mpg, useNA = "always")
```

```
##
##   f.mpg-[20,45.4] f.mpg-(45.4,53.3] f.mpg-(53.3,61.4] f.mpg-(61.4,88.3]
##              1240              1328              1208              1184
##              <NA>
##                 0
```

4.6 Discretization of the variable year

```
df$f.year <- factor(cut(df$year, breaks = c(quantile(df$year, seq(0, 1, 0.25), na.rm = TRUE)),
    include.lowest = T))
levels(df$f.year) <- paste("f.mpg-", levels(df$f.year), sep = "")
table(df$f.year, useNA = "always")
```

```
##
## f.mpg-[2008,2016] f.mpg-(2016,2017] f.mpg-(2017,2019] f.mpg-(2019,2020]
##              1757               879              2013               311
##              <NA>
##                 0
```

5 Create variable adding the total number missing values, outliers and errors.

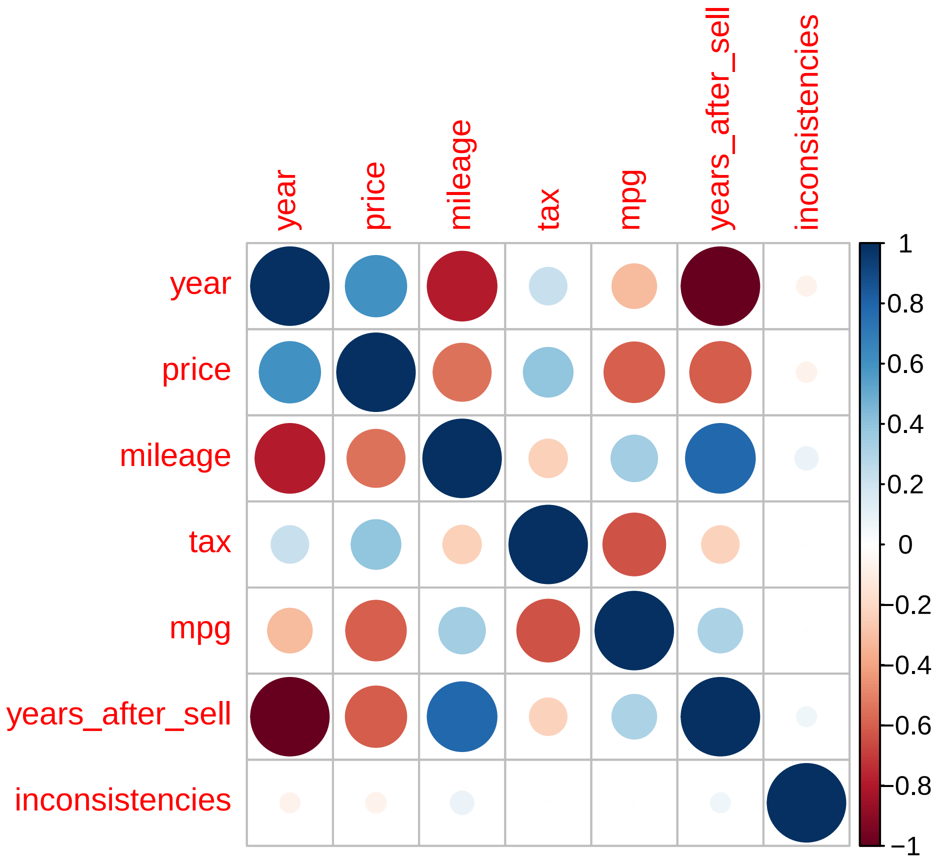
Describe these variables, to which other variables exist higher associations.

5.1 Compute the correlation with all other variables. Rank these variables according the correlation

```
df$inconsistencies <- imis + iouts + ierrs
vars_quanti <- c(2, 3, 5, 7, 8, 13, 18)
res <- cor(df[, vars_quanti])
round(res, 2)
```

```
##                  year price mileage   tax   mpg years_after_sell
## year             1.00  0.61   -0.79  0.22 -0.32            -1.00
## price            0.61  1.00   -0.54  0.39 -0.59            -0.61
## mileage         -0.79 -0.54    1.00 -0.23  0.35             0.79
## tax              0.22  0.39   -0.23  1.00 -0.63            -0.22
## mpg             -0.32 -0.59    0.35 -0.63  1.00             0.32
## years_after_sell -1.00 -0.61    0.79 -0.22  0.32             1.00
## inconsistencies -0.06 -0.06    0.08  0.00 -0.01             0.06
##                 inconsistencies
## year                      -0.06
## price                     -0.06
## mileage                    0.08
## tax                        0.00
## mpg                       -0.01
## years_after_sell           0.06
## inconsistencies            1.00
```

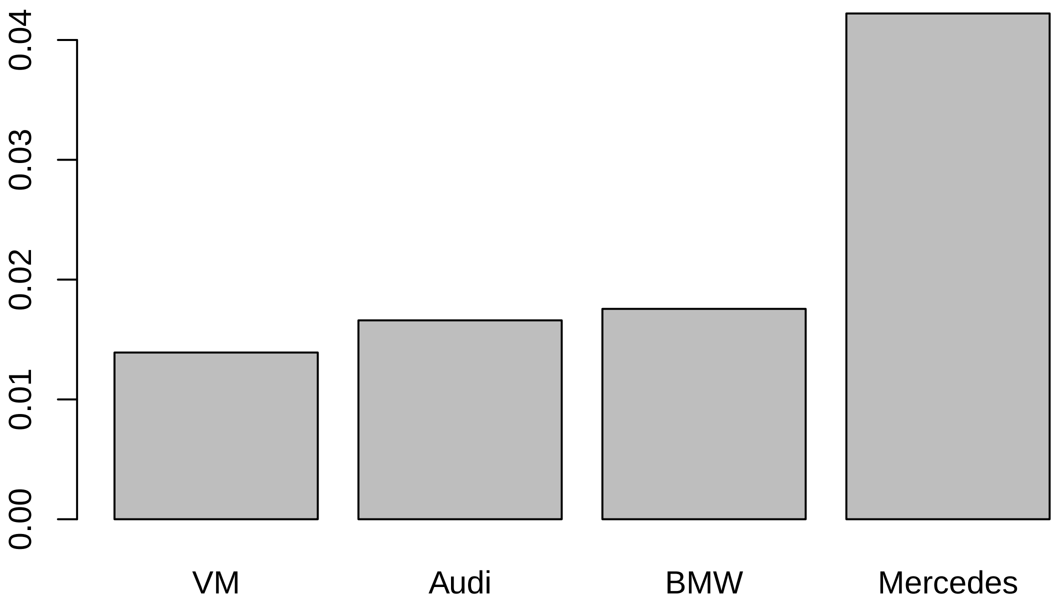
```
corrplot(res)
```



5.2 Mean of missing/outliers/errors per groups

Compute for every group of individuals (group of age, etc, ...) the mean of missing/outliers/errors values. Rank the groups according the computed mean.

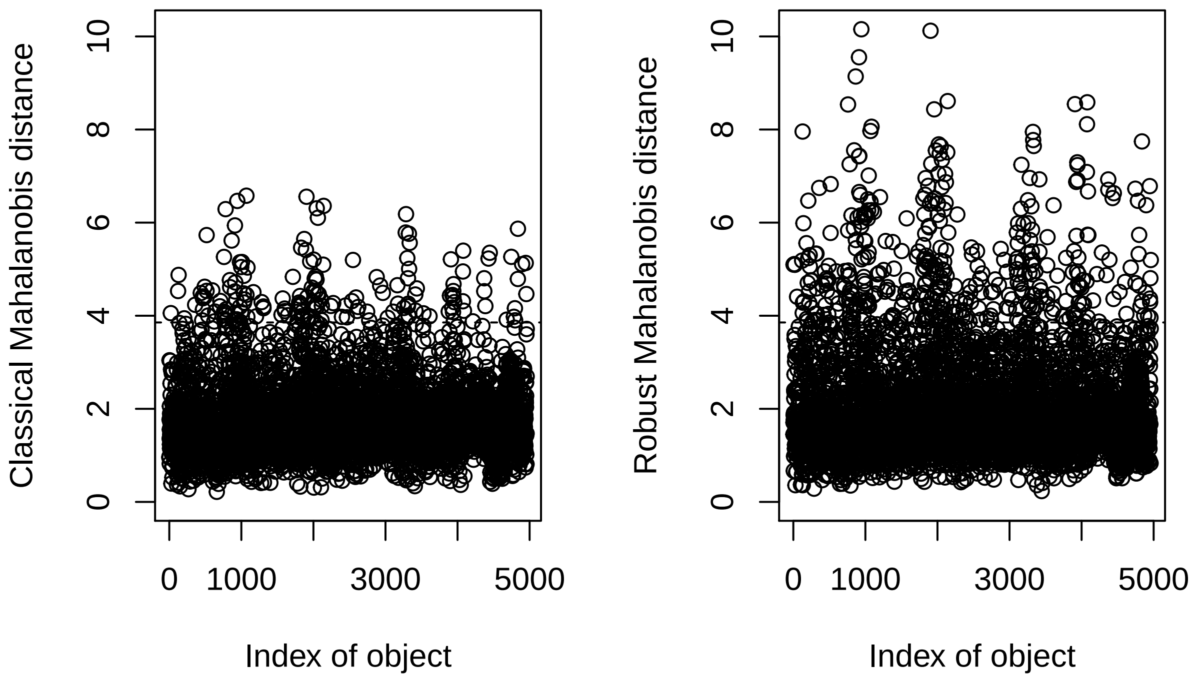
```
dfInconsistencias <- data.frame(Modelo = c("Audi", "VM", "Mercedes", "BMW"))
vw_inconsis <- mean(df$inconsistencies[which(df$manufacturer == "VW")])
audi_inconsis <- mean(df$inconsistencies[which(df$manufacturer == "Audi")])
bmw_inconsis <- mean(df$inconsistencies[which(df$manufacturer == "BMW")])
merc_inconsis <- mean(df$inconsistencies[which(df$manufacturer == "Mercedes")])
dfInconsistencias$incons <- c(vw_inconsis, audi_inconsis, bmw_inconsis, merc_inconsis)
dfInconsistencias <- dfInconsistencias[order(dfInconsistencias$incons), ]
barplot(dfInconsistencias$incons, names.arg = dfInconsistencias$Modelo)
```



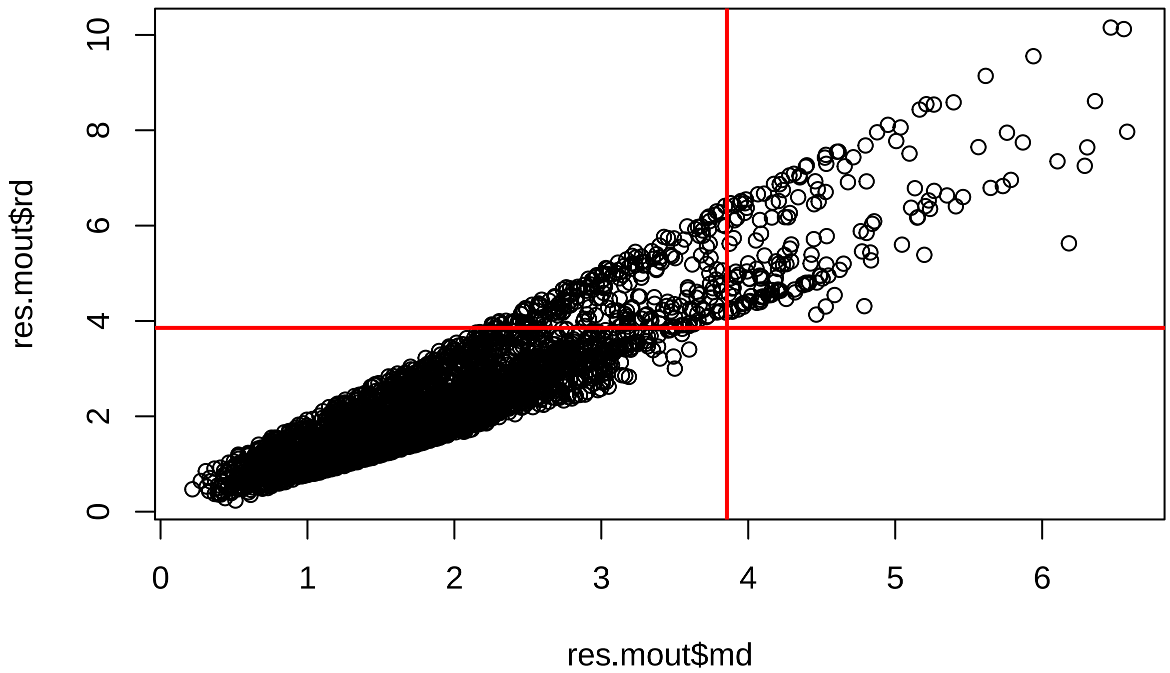
6 Multivariant outliers

We don't use the variable tax for the searching of multivariant outliers because it is a column linearly dependent with other column.

```
res.mout <- Moutlier(df[, c(2, 3, 5, 8)], quantile = 0.995)
```



```r
par(mfrow = c(1, 1))
plot(res.mout$md, res.mout$rd)
abline(h = res.mout$cutoff, lwd = 2, col = "red")
abline(v = res.mout$cutoff, lwd = 2, col = "red")
```




```r
llmout <- which((res.mout$md > res.mout$cutoff) & (res.mout$rd > res.mout$cutoff))
llmout
```

```
##    21  123  130  209  361  440  462  470  471  472  492  497  524  525  546  605
##    21  122  129  208  359  437  457  465  466  467  487  491  517  518  538  596
##   633  645  713  719  770  771  773  787  790  792  795  800  852  855  876  902
##   624  636  703  709  759  760  762  776  779  781  784  789  841  844  865  891
##   921  922  926  927  940  945  955  991  992  994 1008 1009 1020 1045 1046 1049
##   910  911  915  916  929  934  944  980  981  983  997  998 1009 1034 1035 1038
## 1057 1081 1082 1094 1096 1180 1216 1296 1321 1322 1577 1595 1622 1626 1670 1738
## 1046 1070 1071 1083 1085 1167 1203 1281 1306 1307 1557 1574 1601 1605 1648 1715
## 1797 1808 1830 1832 1842 1845 1857 1863 1864 1877 1881 1901 1924 1932 1942 1951
## 1770 1780 1802 1804 1814 1817 1829 1835 1836 1849 1853 1873 1896 1904 1914 1923
## 1953 1983 1994 2006 2029 2031 2032 2041 2046 2060 2061 2067 2068 2071 2074 2075
## 1925 1955 1966 1978 2001 2003 2004 2013 2018 2032 2033 2039 2040 2043 2046 2047
## 2088 2090 2103 2135 2142 2146 2165 2171 2271 2304 2427 2478 2497 2567 2582 2626
## 2060 2062 2074 2106 2113 2117 2136 2142 2242 2275 2396 2446 2465 2534 2549 2592
## 2641 2657 2780 2809 2915 2964 3001 3071 3148 3205 3213 3295 3317 3321 3325 3343
## 2606 2622 2745 2774 2878 2926 2963 3031 3108 3165 3173 3255 3277 3281 3285 3303
## 3349 3351 3352 3356 3360 3361 3365 3368 3376 3454 3455 3475 3567 3651 3918 3935
## 3309 3311 3312 3316 3320 3321 3325 3328 3336 3414 3415 3435 3527 3611 3878 3895
## 3948 3966 3969 3977 3981 3983 3984 3990 3995 4114 4115 4119 4129 4253 4410 4411
## 3908 3926 3929 3937 3941 3943 3944 3950 3955 4074 4075 4079 4089 4213 4370 4371
## 4426 4470 4487 4722 4786 4822 4836 4839 4877 4878 4936 4986 4995
## 4386 4430 4447 4682 4746 4782 4796 4799 4837 4838 4896 4946 4955
```

```r
df$mout <- 0
df$mout[llmout] <- 1
df$mout <- factor(df$mout, labels = c("MvOut.No", "MvOut.Yes"))
res.mout$cutoff
```

```
## [1] 3.854901
```

```r
res.cat <- catdes(df[, c(2:8, 10, 18:19)], 10)
res.cat$category
```

```
## $MvOut.No
##                                Cla/Mod  Mod/Cla   Global      p.value
## transmission=f.Trans-SemiAuto  98.11222 39.08502 38.44758 4.698928e-07
## manufacturer=VW                97.80876 30.77084 30.36290 6.675927e-04
## manufacturer=Audi              95.26902 21.45394 21.73387 1.503873e-02
## manufacturer=BMW               95.02814 21.16148 21.49194 4.212406e-03
## transmission=f.Trans-Automatic 93.90244 25.73637 26.45161 1.358601e-08
##                                  v.test
## transmission=f.Trans-SemiAuto   5.038215
## manufacturer=VW                 3.402554
## manufacturer=Audi              -2.431445
## manufacturer=BMW               -2.861802
## transmission=f.Trans-Automatic -5.678526
##
## $MvOut.Yes
##                                Cla/Mod  Mod/Cla   Global      p.value
## transmission=f.Trans-Automatic 6.097561 46.24277 26.45161 1.358601e-08
## manufacturer=BMW               4.971857 30.63584 21.49194 4.212406e-03
## manufacturer=Audi              4.730983 29.47977 21.73387 1.503873e-02
## manufacturer=VW                2.191235 19.07514 30.36290 6.675927e-04
## transmission=f.Trans-SemiAuto  1.887782 20.80925 38.44758 4.698928e-07
##                                  v.test
## transmission=f.Trans-Automatic  5.678526
## manufacturer=BMW                2.861802
## manufacturer=Audi               2.431445
## manufacturer=VW                -3.402554
## transmission=f.Trans-SemiAuto  -5.038215
```

```r
# The cars with Automatic transmission are overrepresented in multivariant
# outliers. And also there is a high percentage of automatic cars that are
# outliers (6.1%) in comparison to cars with other types of transmission. There
# is a relative low amount of semiautomatic cars that are outliers (20.81%)
# compared to the global amount of semiautomatic cars (38.45%).

summary(df[df$mout == "MvOut.Yes", ])
```

```
##                   model          year          price
##  Audi- Q7           : 14   Min.   :2008   Min.   : 1450
##  VW- Golf           : 13   1st Qu.:2011   1st Qu.: 7750
##  BMW- 3 Series      : 12   Median :2015   Median :12990
##  Audi- A3           : 10   Mean   :2015   Mean   :25146
##  BMW- X5            : 10   3rd Qu.:2018   3rd Qu.:53950
##  Mercedes- GLE Class:  8   Max.   :2020   Max.   :61682
##  (Other)            :106
##             transmission    mileage                 fuelType        tax
##  f.Trans-Manual    :57    Min.   :     10   f.Fuel-Diesel:110   Min.   :  0.0
##  f.Trans-SemiAuto :36    1st Qu.: 10782   f.Fuel-Petrol: 60   1st Qu.:125.0
##  f.Trans-Automatic:80    Median : 65000   f.Fuel-Hybrid:  3   Median :145.0
##                          Mean   : 57541                       Mean   :176.6
##                          3rd Qu.: 91969                       3rd Qu.:235.0
##                          Max.   :119000                       Max.   :580.0
##
##       mpg          engineSize   manufacturer  price_type          Audi
##  Min.   :20.0   2       :59   Audi    :51    Length:173          No :122
##  1st Qu.:32.5   3       :51   BMW     :53    Class :character    Yes: 51
##  Median :41.5   4       :12   Mercedes:36    Mode  :character
##  Mean   :45.5   1.6     :11   VW      :33
```

```
##  3rd Qu.:58.9   1.4    : 6
##  Max.   :88.3   2.1    : 5
##                 (Other):29
##  years_after_sell             f.tax                                f.mileage
##  Min.   : 2.000   f.tax-[0,125]   :47   f.mileage-[1,6e+03]          : 23
##  1st Qu.: 4.000   f.tax-(125,145]:55   f.mileage-(6e+03,1.74e+04]   : 27
##  Median : 7.000   f.tax-(145,155]:10   f.mileage-(1.74e+04,3.48e+04]: 11
##  Mean   : 7.198   f.tax-(155,580]:61   f.mileage-(3.48e+04,1.19e+05]:112
##  3rd Qu.:10.700
##  Max.   :14.000
##
##               f.mpg                    f.year    inconsistencies
##  f.mpg-[20,45.4]   :101   f.mpg-[2008,2016]:108   Min.   :0.0000
##  f.mpg-(45.4,53.3]: 19   f.mpg-(2016,2017]: 16   1st Qu.:0.0000
##  f.mpg-(53.3,61.4]: 19   f.mpg-(2017,2019]: 37   Median :0.0000
##  f.mpg-(61.4,88.3]: 34   f.mpg-(2019,2020]: 12   Mean   :0.1214
##                                                  3rd Qu.:0.0000
##                                                  Max.   :2.0000
##
##        mout
##  MvOut.No :  0
##  MvOut.Yes:173
##
##
##
##
##
```

```r
summary(df)
```

```
##                model           year          price
##  VW- Golf          : 488   Min.   :2008   Min.   : 1250
##  Mercedes- C Class: 394   1st Qu.:2016   1st Qu.:13999
##  VW- Polo          : 330   Median :2017   Median :19310
##  Mercedes- A Class: 262   Mean   :2017   Mean   :20947
##  BMW- 3 Series     : 251   3rd Qu.:2019   3rd Qu.:25950
##  Mercedes- E Class: 199   Max.   :2020   Max.   :61682
##  (Other)           :3036
##            transmission     mileage                fuelType         tax
##  f.Trans-Manual    :1741   Min.   :      1   f.Fuel-Diesel:2846   Min.   :  0.0
##  f.Trans-SemiAuto :1907   1st Qu.:   6000   f.Fuel-Petrol:2048   1st Qu.:125.0
##  f.Trans-Automatic:1312   Median : 17415   f.Fuel-Hybrid:   66   Median :145.0
##                           Mean   : 23441                         Mean   :122.7
##                           3rd Qu.: 34768                         3rd Qu.:145.0
##                           Max.   :119000                         Max.   :580.0
##
##       mpg          engineSize     manufacturer   price_type         Audi
##  Min.   :20.00   2       :2092   Audi     :1078   Length:4960        No :3882
##  1st Qu.:45.40   3       : 556   BMW      :1066   Class :character   Yes:1078
##  Median :53.30   1.5     : 520   Mercedes:1310   Mode  :character
##  Mean   :53.21   2.1     : 395   VW       :1506
##  3rd Qu.:61.40   1       : 374
##  Max.   :88.30   1.6     : 365
##                  (Other): 658
##  years_after_sell             f.tax                                 f.mileage
##  Min.   : 2.000   f.tax-[0,125]   :1447   f.mileage-[1,6e+03]          :1253
##  1st Qu.: 3.000   f.tax-(125,145]:2537   f.mileage-(6e+03,1.74e+04]   :1227
##  Median : 5.000   f.tax-(145,155]: 499   f.mileage-(1.74e+04,3.48e+04]:1240
##  Mean   : 4.798   f.tax-(155,580]: 477   f.mileage-(3.48e+04,1.19e+05]:1240
##  3rd Qu.: 6.000
##  Max.   :14.000
##
##               f.mpg                    f.year     inconsistencies
```

```
##  f.mpg-[20,45.4]   :1240   f.mpg-[2008,2016]:1757   Min.   :0.00000
##  f.mpg-(45.4,53.3]:1328   f.mpg-(2016,2017]: 879   1st Qu.:0.00000
##  f.mpg-(53.3,61.4]:1208   f.mpg-(2017,2019]:2013   Median :0.00000
##  f.mpg-(61.4,88.3]:1184   f.mpg-(2019,2020]: 311   Mean   :0.02177
##                                                     3rd Qu.:0.00000
##                                                     Max.   :2.00000
##
##          mout
##  MvOut.No :4787
##  MvOut.Yes: 173
##
##
##
##
##
```

```
# The cars that are outliers tend to be more expensive, have more mileage, have
# to pay more tax. The manufacturers Mercedes and VW have a low percentage of
# outliers cars.
```

7 Profiling with FactoMineR

7.1 Profiling of the numeric target variable "price"

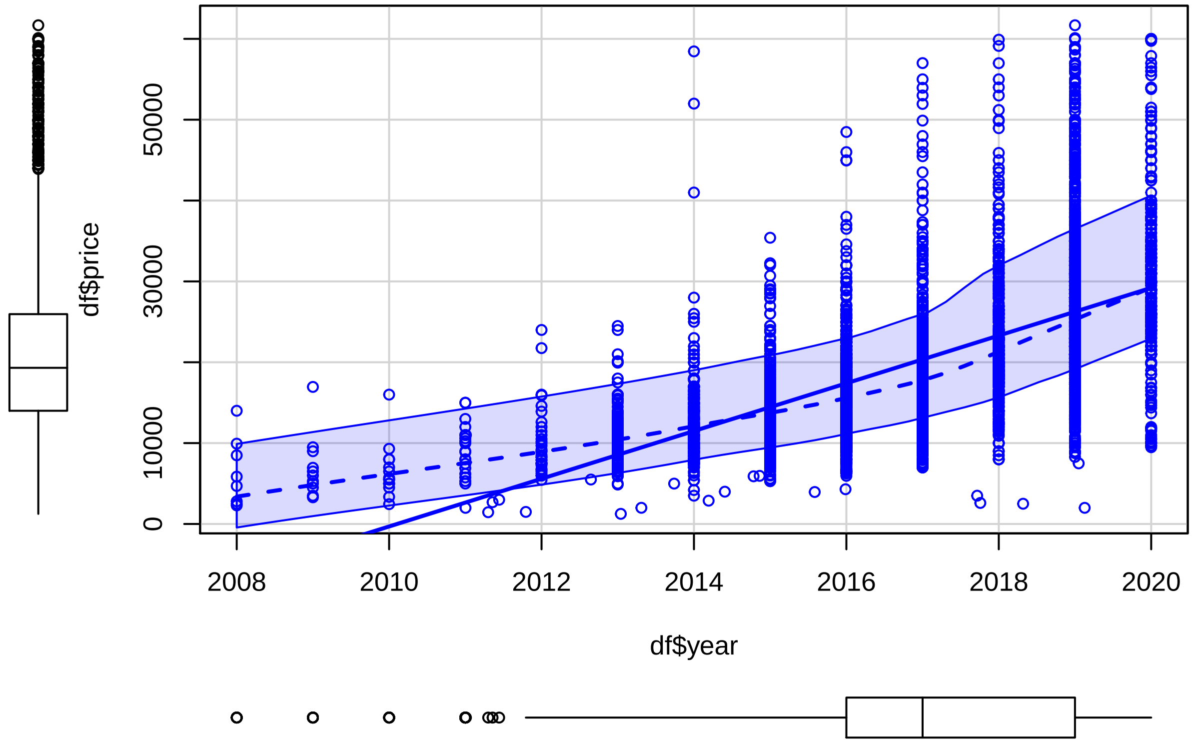
```
summary(df$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1250   13999   19310   20947   25950   61682
```

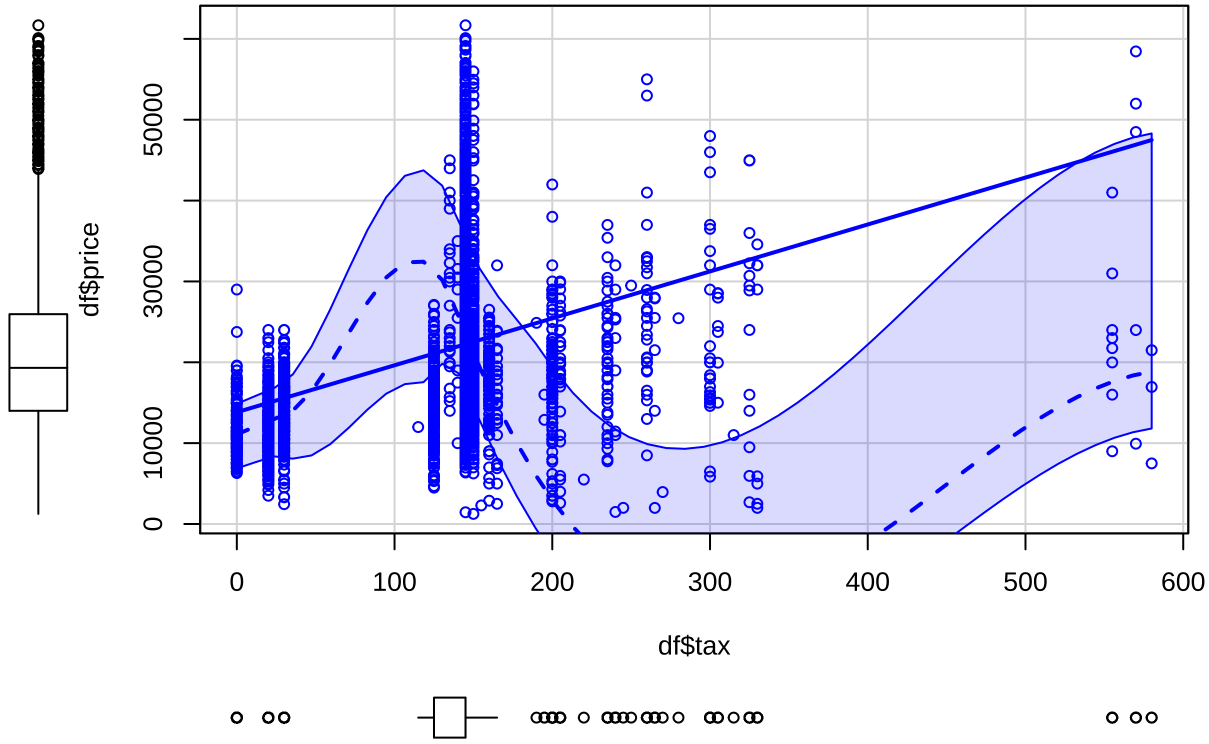
```
# The 'variable to describe cannot have NA ###################################
res.condes <- condes(df, 3, proba = 0.01)

res.condes$quanti  # Global association to numeric variables
```

```
##                  correlation       p.value
## year              0.60503789  0.000000e+00
## tax               0.39426511 3.704156e-184
## inconsistencies  -0.06457838  5.321817e-06
## mileage          -0.54093485  0.000000e+00
## mpg              -0.59022331  0.000000e+00
## years_after_sell -0.60503789  0.000000e+00
```

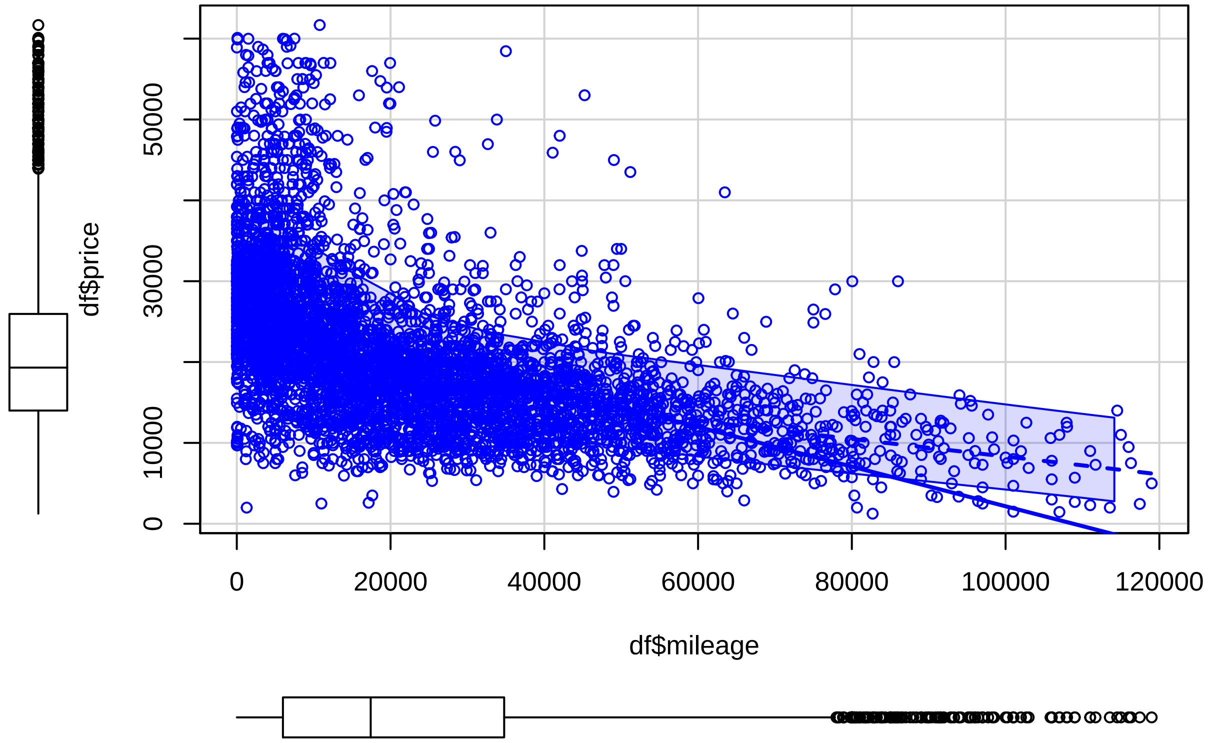
```
# The response variable has a strong correlation with the following variables:
# year, tax, mileage and mpg.
scatterplot(df$year, df$price)
```



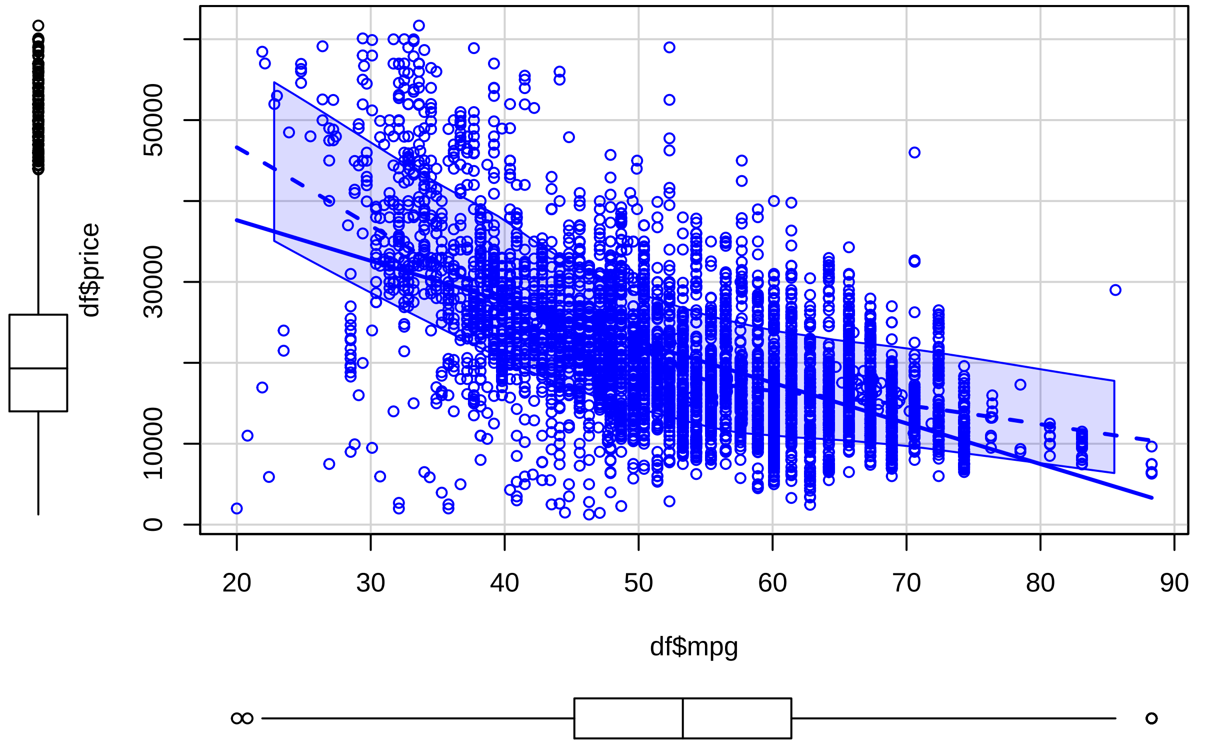

```
scatterplot(df$tax, df$price)
```



```r
scatterplot(df$mileage, df$price)
```




```r
scatterplot(df$mpg, df$price)
```



```r
res.condes$quali  # Global association to factors
```

```
##                      R2       p.value
## model       0.474917753  0.000000e+00
## engineSize  0.367067198  0.000000e+00
## f.tax       0.291181741  0.000000e+00
## f.mileage   0.313153495  0.000000e+00
## f.mpg       0.317185000  0.000000e+00
## f.year      0.358780233  0.000000e+00
## transmission 0.248322843 5.542527e-308
## manufacturer 0.084110220  4.575385e-94
## Audi        0.007068328  3.028155e-09
## mout        0.006743925  6.958923e-09
## fuelType    0.006538543  8.685789e-08
## price_type  0.004751292  6.188509e-04
```

```r
# P-values indicate whether the correlation is statistically different from 0
# or not. p-values < 0.05 reject the null hypothesis (correlation statistically
# equal to 0).  The response variable has a strong correlation with the
# following variables: model, engineSize and transmission, according to the R2
# stadistic value.

res.condes$category  # Partial association to significative levels in factors
```

```
##                                          Estimate       p.value
## f.mpg=f.mpg-[20,45.4]                   8960.96525  0.000000e+00
## f.tax=f.tax-(125,145]                   5439.00703 3.955692e-273
## f.year=f.mpg-(2017,2019]                4198.97742 5.711012e-269
## f.mileage=f.mileage-[1,6e+03]           7176.59346 9.098787e-221
## engineSize=3                            6353.43300 3.214706e-188
## transmission=f.Trans-SemiAuto           3963.60468 1.297279e-126
## f.year=f.mpg-(2019,2020]                7569.79477  6.938730e-64
## model=Audi- Q7                         16219.30863  1.023298e-44
## engineSize=4                           19642.95126  1.232148e-37
## model=Mercedes- GLE Class              14362.72576  1.481729e-35
## transmission=f.Trans-Automatic          2687.03006  3.264985e-34
## f.mileage=f.mileage-(6e+03,1.74e+04]    2913.62688  6.295063e-34
## manufacturer=Mercedes                   2435.17606  2.875906e-32
## model=Audi- Q5                          5765.03618  1.190792e-26
## model=BMW- X5                          11214.67448  1.882828e-26
## model=Mercedes- GLC Class               6293.29786  4.001020e-24
## engineSize=2.9                         19179.08587  2.025514e-17
## model=BMW- 7 Series                    18054.00782  2.950833e-16
## model=VW- Touareg                       7513.62320  1.751379e-15
## model=BMW- M4                          15328.03163  7.507226e-15
## model=BMW- X3                           5547.69448  3.543783e-14
## model=Audi- Q8                         25943.47448  1.352148e-12
## model=Mercedes- GLS Class              21090.50782  6.260528e-11
## model=Audi- RS6                        21839.27448  8.342478e-10
## model=Mercedes- S Class                 8344.57448  1.329486e-09
## Audi=Yes                                 990.78475  3.028155e-09
## manufacturer=Audi                       1279.89752  3.028155e-09
## model=BMW- X4                           7613.38877  4.424204e-09
## mout=MvOut.Yes                          2175.50315  6.958923e-09
## model=VW- Caravelle                    12833.56337  5.034238e-08
## model=BMW- X2                           4197.24115  3.702210e-07
## model=Audi- R8                         29484.67448  5.971910e-07
## price_type=extremely expensive         16521.06459  1.183034e-06
## engineSize=4.7                         14653.33587  1.269323e-06
## model=Audi- A8                          8732.34115  1.375702e-06
## model=VW- California                   28259.67448  1.484821e-06
## model=Mercedes- SL CLASS                3578.12276  3.281241e-06
```

```
## model=Mercedes- V Class                      3702.23970  2.653816e-05
## engineSize=5.2                              32960.33587  1.132480e-04
## model=Audi- SQ7                             21254.67448  1.498318e-04
## manufacturer=BMW                              725.16008  1.581738e-04
## engineSize=6.6                              31510.33587  2.065764e-04
## engineSize=5.5                              15586.66921  3.317837e-04
## model=VW- Tiguan Allspace                    4956.59115  5.049066e-04
## model=Audi- RS5                             28764.67448  5.539851e-04
## model=VW- Arteon                             1700.27448  8.169826e-04
## model=Mercedes- X-CLASS                      5479.27448  8.241239e-04
## model=BMW- 8 Series                         26244.67448  1.405268e-03
## engineSize=2.5                               7654.50254  2.097729e-03
## model=Audi- RS4                             24254.67448  2.802698e-03
## engineSize=1.9                             -22769.66413  7.989989e-03
## model=Mercedes- SLK                        -13727.12552  3.599545e-03
## engineSize=1.5                              -5726.63528  3.326063e-03
## f.tax=f.tax-(155,580]                         -367.05562  1.719542e-03
## model=VW- CC                               -16106.07552  9.784006e-04
## model=VW- Passat                            -8402.62889  3.989194e-04
## model=VW- Scirocco                         -11704.21441  2.085085e-04
## model=Mercedes- A Class                     -7065.66903  9.690277e-05
## engineSize=2.1                              -6458.93754  4.386782e-05
## engineSize=1.8                             -13283.98231  2.338396e-05
## model=Mercedes- C Class                     -2760.10978  1.563117e-05
## fuelType=f.Fuel-Diesel                        -65.99571  4.439815e-07
## fuelType=f.Fuel-Petrol                      -1574.14232  3.300020e-08
## mout=MvOut.No                               -2175.50315  6.958923e-09
## Audi=No                                      -990.78475  3.028155e-09
## model=Audi- A3                              -8860.15715  2.061653e-09
## model=Mercedes- E Class                       -85.27527  2.918196e-12
## model=BMW- 1 Series                        -10034.88868  2.819099e-14
## model=Audi- A1                             -11482.76398  1.458366e-15
## engineSize=2                                -3210.80896  1.363936e-16
## f.year=f.mpg-(2016,2017]                    -3842.76289  1.277636e-17
## engineSize=1.6                              -9928.96824  1.887986e-28
## model=VW- Golf                              -9565.78863  1.214110e-30
## engineSize=1.4                             -10492.05480  2.793009e-32
## engineSize=1.2                             -15682.25788  4.604953e-40
## model=VW- Up                               -17472.45552  2.437578e-40
## f.mileage=f.mileage-(1.74e+04,3.48e+04]     -3171.95614  3.897821e-41
## f.mpg=f.mpg-(53.3,61.4]                     -3699.90942  7.884861e-56
## engineSize=1                               -13280.49300  1.817435e-75
## model=VW- Polo                             -14504.39521  1.268340e-81
## manufacturer=VW                             -4440.23366  3.900322e-92
## f.mpg=f.mpg-(61.4,88.3]                     -5246.53100 9.896542e-109
## f.mileage=f.mileage-(3.48e+04,1.19e+05]     -6918.26420 4.345811e-202
## transmission=f.Trans-Manual                 -6650.63474 8.023404e-306
## f.year=f.mpg-[2008,2016]                    -7926.00930 5.285099e-318
## f.tax=f.tax-[0,125]                         -6683.08887 1.100225e-318
```

```
# With this output we can see from different categories the mean difference in
# price compared to the mean price of the dataset The cars that have low mpg
# are more expensive. We can also see that the cars with an engine size = 4 has
# an estimate of +19400$ We can also see that the cars with an engine size =
# 2.9 has an estimate of +19179$ We can also see that the cars with an
# model=BMW- 7 Series has an estimate of +18054$ We can also see that the cars
# with an model=Audi- Q8 has an estimate of +25943$ We can also see that the
# cars with an engine size = 5.2 has an estimate of +32960$ We can also see
# that the cars with an model=VW- Up has an estimate of -17472$ We can also see
# that the cars with an engine size = 1.2 has an estimate of -15682$
```

7.2 Profiling of the categorical target variable "Audi"

```r
summary(df$Audi)
```

```
##   No  Yes 
## 3882 1078
```

```r
# The 'variable to describe cannot have NA
res.catdes <- catdes(df[, -c(1)], 11, proba = 0.01)
# We exclude the model of the car from the analysis because it doesn't bring
# useful information.
res.catdes$quanti.var  # Global association to numeric variables
```

```
##                Eta2      P-value
## mpg     0.012673720 1.837841e-15
## price   0.007068328 3.028155e-09
## mileage 0.002584532 3.412143e-04
```

Miles per galon (mpg), price and mileage are statistically significant variables as they have a p-value less than 0.01. Despite that fact, the effect size associated with them is quite small as they have a small Eta2 value. This means that these variables are not quite significant at predicting if a car is an Audi or not.

```r
res.catdes$quanti  # Partial association of numeric variables to levels of outcome factor
```

```
## $No
##            v.test Mean in category Overall mean sd in category  Overall sd
## mpg      7.927735         53.88327     53.20574        11.2462    11.42077
## mileage -3.580041      22866.50952  23440.58117     21143.0294 21428.58621
## price   -5.920459      20516.25425  20946.92601      9453.3950  9720.89901
##              p.value
## mpg     2.231792e-15
## mileage 3.435401e-04
## price   3.210437e-09
## 
## $Yes
##            v.test Mean in category Overall mean sd in category  Overall sd
## price    5.920459      22497.82375  20946.92601    10483.00980  9720.89901
## mileage  3.580041      25507.87814  23440.58117    22304.73395 21428.58621
## mpg     -7.927735         50.76588     53.20574       11.70803    11.42077
##              p.value
## price   3.210437e-09
## mileage 3.435401e-04
## mpg     2.231792e-15
```

With this output we can see that Audi cars have a little more price and mileage than the global average and have fewer mpg than the global average. The opposite is true for cars that are not Audi.

```r
# mean(df$tax[which(df$Audi=='No')])-mean(df$tax[which(df$Audi=='Yes')])
res.catdes$test.chi2  # Global association to factors
```

```
##                   p.value df
## manufacturer 0.000000e+00  3
## engineSize   7.120314e-87 26
## f.mpg        9.083976e-18  3
## fuelType     1.798737e-06  2
## f.mileage    9.535424e-06  3
## transmission 1.866325e-05  2
```

```r
res.catdes$category  # Partial association to significative levels in factors
```

```
## $No
##                                     Cla/Mod     Mod/Cla     Global
## manufacturer=VW                   100.00000 38.79443586 30.3629032
## manufacturer=Mercedes             100.00000 33.74549201 26.4112903
## manufacturer=BMW                  100.00000 27.46007213 21.4919355
## engineSize=2.1                    100.00000 10.17516744  7.9637097
## engineSize=1.2                     98.43750  3.24574961  2.5806452
## f.mpg=f.mpg-(61.4,88.3]            84.37500 25.73415765 23.8709677
## engineSize=1.5                     87.88462 11.77228233 10.4838710
## engineSize=1.3                    100.00000  1.90623390  1.4919355
## f.mileage=f.mileage-(6e+03,1.74e+04]  83.04808 26.24935600 24.7379032
## fuelType=f.Fuel-Hybrid             96.96970  1.64863472  1.3306452
## engineSize=1                       85.29412  8.21741370  7.5403226
## transmission=f.Trans-SemiAuto      80.75511 39.67027306 38.4475806
## f.year=f.mpg-(2017,2019]           80.52658 41.75682638 40.5846774
## fuelType=f.Fuel-Diesel             79.79621 58.50077280 57.3790323
## f.mpg=f.mpg-(53.3,61.4]            81.29139 25.29623905 24.3548387
## engineSize=2.5                     16.66667  0.02575992  0.1209677
## fuelType=f.Fuel-Petrol             75.53711 39.85059248 41.2903226
## transmission=f.Trans-Manual        74.61229 33.46213292 35.1008065
## engineSize=4                       34.61538  0.23183926  0.5241935
## engineSize=2                       74.61759 40.21123132 42.1774194
## f.mpg=f.mpg-[20,45.4]              70.00000 22.35960845 25.0000000
## engineSize=1.4                     46.93878  4.14734673  6.9153226
## manufacturer=Audi                   0.00000  0.00000000 21.7338710
##                                         p.value     v.test
## manufacturer=VW                   3.211328e-197  29.960501
## manufacturer=Mercedes             1.991990e-166  27.495413
## manufacturer=BMW                  9.503811e-131  24.329726
## engineSize=2.1                     8.850459e-45  14.040166
## engineSize=1.2                     1.116329e-11   6.790645
## f.mpg=f.mpg-(61.4,88.3]            2.053470e-09   5.993520
## engineSize=1.5                     2.798423e-09   5.943006
## engineSize=1.3                     1.142992e-08   5.708018
## f.mileage=f.mileage-(6e+03,1.74e+04]  1.852532e-06   4.768880
## fuelType=f.Fuel-Hybrid             1.760962e-05   4.293225
## engineSize=1                       3.803700e-04   3.553342
## transmission=f.Trans-SemiAuto      7.385074e-04   3.374869
## f.year=f.mpg-(2017,2019]           1.368379e-03   3.201239
## fuelType=f.Fuel-Diesel             2.502241e-03   3.023070
## f.mpg=f.mpg-(53.3,61.4]            3.067589e-03   2.960883
## engineSize=2.5                     2.471532e-03  -3.026805
## fuelType=f.Fuel-Petrol             9.932039e-05  -3.892246
## transmission=f.Trans-Manual        5.346875e-06  -4.550696
## engineSize=4                       2.262154e-06  -4.728472
## engineSize=2                       1.176070e-07  -5.297176
## f.mpg=f.mpg-[20,45.4]              1.918626e-15  -7.946495
## engineSize=1.4                     5.929733e-40 -13.229479
## manufacturer=Audi                  0.000000e+00       -Inf
##
## $Yes
##                                     Cla/Mod     Mod/Cla     Global
## manufacturer=Audi                100.000000 100.0000000 21.7338710
## engineSize=1.4                    53.061224  16.8831169  6.9153226
## f.mpg=f.mpg-[20,45.4]             30.000000  34.5083488 25.0000000
## engineSize=2                      25.382409  49.2578850 42.1774194
## engineSize=4                      65.384615   1.5769944  0.5241935
## transmission=f.Trans-Manual       25.387708  41.0018553 35.1008065
## fuelType=f.Fuel-Petrol            24.462891  46.4749536 41.2903226
## engineSize=2.5                    83.333333   0.4638219  0.1209677
## f.mpg=f.mpg-(53.3,61.4]           18.708609  20.9647495 24.3548387
```

```
## fuelType=f.Fuel-Diesel                   20.203795  53.3395176 57.3790323
## f.year=f.mpg-(2017,2019]                 19.473423  36.3636364 40.5846774
## transmission=f.Trans-SemiAuto            19.244887  34.0445269 38.4475806
## engineSize=1                             14.705882   5.1020408  7.5403226
## fuelType=f.Fuel-Hybrid                    3.030303   0.1855288  1.3306452
## f.mileage=f.mileage-(6e+03,1.74e+04]     16.951915  19.2949907 24.7379032
## engineSize=1.3                            0.000000   0.0000000  1.4919355
## engineSize=1.5                           12.115385   5.8441558 10.4838710
## f.mpg=f.mpg-(61.4,88.3]                  15.625000  17.1614100 23.8709677
## engineSize=1.2                            1.562500   0.1855288  2.5806452
## engineSize=2.1                            0.000000   0.0000000  7.9637097
## manufacturer=BMW                          0.000000   0.0000000 21.4919355
## manufacturer=Mercedes                     0.000000   0.0000000 26.4112903
## manufacturer=VW                           0.000000   0.0000000 30.3629032
##                                               p.value     v.test
## manufacturer=Audi                        0.000000e+00        Inf
## engineSize=1.4                           5.929733e-40  13.229479
## f.mpg=f.mpg-[20,45.4]                    1.918626e-15   7.946495
## engineSize=2                             1.176070e-07   5.297176
## engineSize=4                             2.262154e-06   4.728472
## transmission=f.Trans-Manual              5.346875e-06   4.550696
## fuelType=f.Fuel-Petrol                   9.932039e-05   3.892246
## engineSize=2.5                           2.471532e-03   3.026805
## f.mpg=f.mpg-(53.3,61.4]                  3.067589e-03  -2.960883
## fuelType=f.Fuel-Diesel                   2.502241e-03  -3.023070
## f.year=f.mpg-(2017,2019]                 1.368379e-03  -3.201239
## transmission=f.Trans-SemiAuto            7.385074e-04  -3.374869
## engineSize=1                             3.803700e-04  -3.553342
## fuelType=f.Fuel-Hybrid                   1.760962e-05  -4.293225
## f.mileage=f.mileage-(6e+03,1.74e+04]     1.852532e-06  -4.768880
## engineSize=1.3                           1.142992e-08  -5.708018
## engineSize=1.5                           2.798423e-09  -5.943006
## f.mpg=f.mpg-(61.4,88.3]                  2.053470e-09  -5.993520
## engineSize=1.2                           1.116329e-11  -6.790645
## engineSize=2.1                           8.850459e-45 -14.040166
## manufacturer=BMW                        9.503811e-131 -24.329726
## manufacturer=Mercedes                   1.991990e-166 -27.495413
## manufacturer=VW                         3.211328e-197 -29.960501
```

With this final categorical analysis we can see that: For cars that are not Audi: *We have smaller engine sizes overall.* The percentage of cars with diesel and hybrid engines is slightly higher than the global mean. *We have more cars with a lower mileage.

For cars that are Audi: *The percentage of engines with a size of 1.4 is higher than the global mean (16.9 vs 6.9).* The percentage of Audis with a manual transmission is higher than the global mean (41 vs 35).