UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Departament d'Estadística
i Investigació Operativa

CAR PRICES

Projects form an important part of the education of software engineers. They form an active method of teaching, as defined by Piaget, leading to a "training in self-discipline and voluntary effort", which is important to software engineering professionals. Two purposes served by these projects are: education in professional practice, and outcome-based assessment.

Data cleaning or data scrubbing is one of the most important steps previous to any data decision-making or modelling process. Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset.

Data cleaning is the process that removes data that does not belong to the dataset or it is not useful for modelling purposes. Data transformation is the process of converting data from one format or structure into another format. Transformation processes can also be referred to as data wrangling, or data munging, transforming and mapping data from one "raw" data form into another format. **Essentially, real-world data is messy data and for model building: garbage data in is garbage analysis out**.

Any dataset for modelling purposes should include a first methodological step on **data preparation** about:

- Removing duplicate or irrelevant observations
- Fix structural errors (usually coding errors, trailing blanks in labels, lower/upper case consistency, etc.).
- Check data types. Dates should be coded as such and factors should have level names (if possible, levels have to be set and clarify the variable they belong to). This point is sometimes included under data transformation process. New derived variables are to be produced sometimes scaling and/or normalization (range/shape changes to numeric variables) or category regrouping for factors (nominal/ordinal).
- Filter unwanted outliers. Univariate and multivariate outliers have to be highlighted. Remove register/erase values and set NA for univariate outiers.
- Handle missing data: figure out why the data is missing. Data imputation is to be considered when the aim is modelling (imputation has to be validated).
- Data validation is mixed of 'common sense and sector knowledge': Does the data make sense? Does the data follow the appropriate rules for its field? Does it prove or disprove the working theory, or bring any insight to light? Can you find trends in the data to help you form a new theory? If not, is that because of a data quality issue?

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 to be randomly selected from Mercedes, BMW, Volkwagen and Audi manufacturers. So, firstly you have to combine used car from the 4 manufacturers into 1 dataframe, adding a column manufacturer containing the vehicle brand.

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 required.

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

This project deals with numeric model building for 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): Y- Price (Numeric Target).

Aim is to predict how much you should sell your old car. It involves a numeric outcome.
A random sample containing 5000 registers combining Audi, VW, Merc and BMW registers has to be retained by each group. Data from:
https://www.kaggle.com/adityadesai13/used-car-dataset-ford-and-mercedes

Data Preparation outline:

Univariate Descriptive Analysis (to be included for each variable**):**

- Original numeric variables corresponding to qualitative concepts have to be converted to factors.
- Original numeric variables corresponding to real quantitative concepts are kept as numeric but additional factors should also be created as a discretization of each numeric variable.
- Exploratory Data Analysis for each variables (numeric summary and graphic support).

Data Quality Report:

Per variable, count:

- Number of missing values
- Number of errors (including inconsistencies)
- Number of outliers
- Rank variables according the sum of missing values (and errors).

Per individuals, count:

- number of missing values
- number of errors,
- number of outliers
- Identify individuals considered as multivariant outliers.

Create variable adding the total number missing values, outliers and errors. Describe these variables, to which other variables exist higher associations.

- Compute the correlation with all other variables. Rank these variables according the correlation
- Compute for every group of individuals (group of age, size of town, singles, married, …) the mean of missing/outliers/errors values. Rank the groups according the computed mean.

Imputation:

- Numeric Variables
- Factors

Profiling:

- Target (age)

Car Prices Report

```
rm(list=ls())
load("Sample_raw.Rdata")
par(mfrow=c(1,1))
```

Data preparation

First, the data was imported and sampled. The result is saved as "Sample_raw.Rdata" and imported.

Variable Analysis

On each variable of this data, descriptive analysis is performed, a data quality report made and imputation and profiling accounted for.

variable 1: model

Model is a nominal variable without missing values. However, it has a lot of levels (89) with a few very sparsely populated ones, such that converting it to a factor is not feasable.

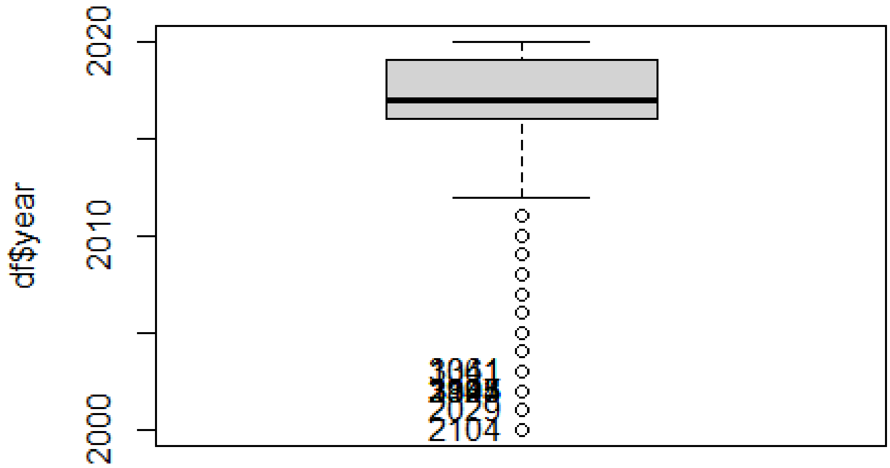
```
summary(df$model)

##    Length     Class      Mode 
##      5000 character character

table(df$model)

## 
##         1 Series         2 Series         3 Series         4 Series 
##              206              141              240               90 
##         5 Series         6 Series         7 Series         8 Series 
##              108               11               12                5 
##          A Class               A1               A3               A4 
##              277              151              208              137 
##               A5               A6               A7               A8 
##               62               86                7               14 
##           Amarok           Arteon          B Class           Beetle 
##               15               24               55                8 
##          C Class      Caddy Life  Caddy Maxi Life       California 
##              354                2               10                1 
##        Caravelle               CC         CL Class        CLA Class 
##               13               13               34                8 
##        CLS Class          E Class              Eos              Fox 
##               37              184                2                1 
##          G Class         GL Class        GLA Class        GLC Class 
##                3               15               84              102 
##        GLE Class        GLS Class             Golf          Golf SV 
##               42                9              497               21 
##               i3               i8            Jetta          M Class 
##                6                3                3                8 
##               M2               M3               M4               M5 
##                3                4               15                5 
##           Passat             Polo               Q2               Q3 
##               80              333               87              135 
##               Q5               Q7               Q8              RS3 
##               95               40                1                1 
##              RS4              RS5              RS6          S Class 
##                3                2                3               23
```

```
##              S3             S4             S5             S8
##               1              1              1              1
##        Scirocco         Sharan        Shuttle       SL CLASS
##              25             33              3             27
##             SLK            SQ5            SQ7        T-Cross
##              13              4              1             20
##           T-Roc         Tiguan Tiguan Allspace        Touareg
##              70            180              9             39
##          Touran             TT             Up        V Class
##              40             34             79             26
##         X-CLASS             X1             X2             X3
##               7             74             29             56
##              X4             X5             X6             X7
##              25             40             12              5
##              Z4
##               6

sum(is.na(df$model))

## [1] 0

df$model[1:5]

## [1] " A3" " A3" " Q5" " A4" " Q3"
```

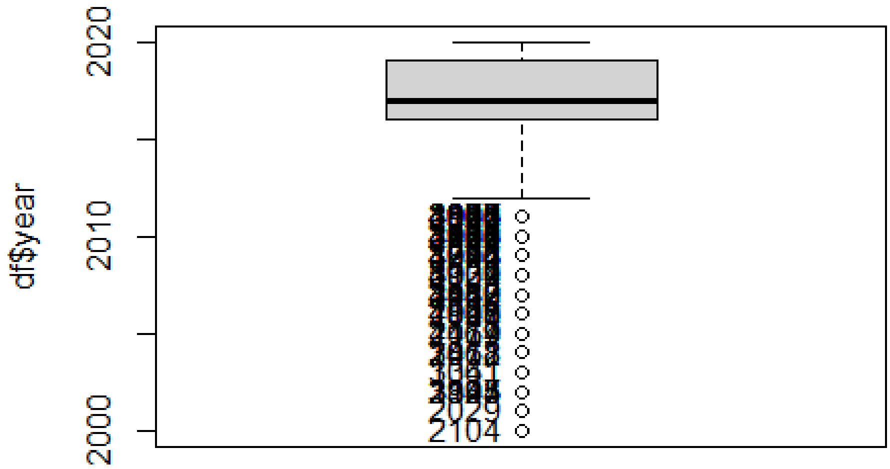
variable 2: year

This is a numeric interval variable. By using a histogram, it is clear that the data set contains mostly recent cars. It contains no missing values thus imputation is not needed. The year variable contains 80 outliers (out of which 25 severe), all on the lower end of the spectrum. This is due to most of the data being from recently build cars. We create two additional variables: a numeric age variable 'n.age' and an age factor "f.age" as discretisation.

```
summary(df$year)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2000    2016    2017    2017    2019    2020

sum(is.na(df$year))

## [1] 0

Boxplot(df$year)
```




```
## [1] 2104 2029 1837 2105 3325 3343 3344 3995 1061 3341
length(Boxplot(df$year, id = list(n=Inf)))
```

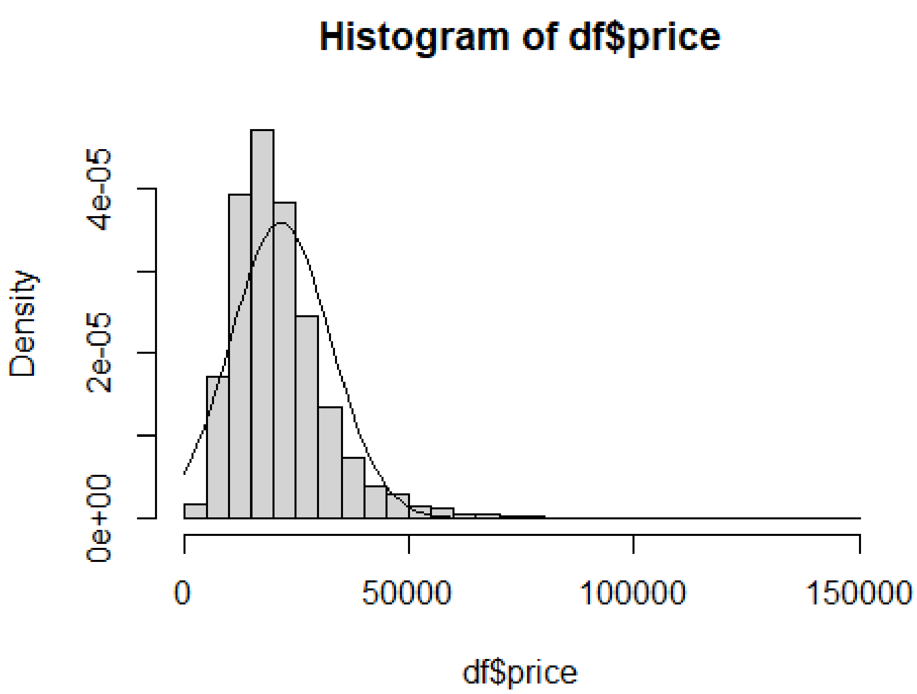

```
## [1] 80
sevout = (quantile(df$year,0.25)-(3*((quantile(df$year,0.75)-quantile(df$year,0.25)))))
length(which(df$year < sevout))
```

```
## [1] 25

df$n.age = max(df$year)-(df$year)

df$f.age <- ifelse(df$n.age <= 1, 1, ifelse(df$n.age > 1 & df$n.age <= 3, 2, ifelse(df$n.a
ge > 3 & df$n.age <= 4, 3, ifelse(df$n.age > 4, 4,0))))
df$f.age <- factor(df$f.age, labels=c("LowAge","LowMidAge","HighMidAge","HighAge"), order
= T, levels=c(1,2,3,4))
table(df$f.age)

##
##     LowAge  LowMidAge HighMidAge    HighAge
##       1938       1426        798        838
```

variable 3: price

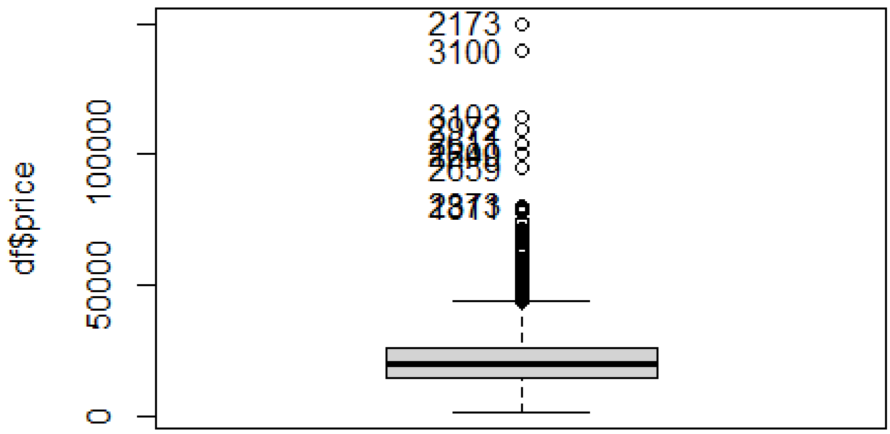
This is a continuous ratio variable. The data is not normally distributed, but this fact is further answered in question 1 in the next section of this document. Again a histogram is used to visualize the data. It contains no missing values thus imputation is not needed. The price variable contains 207 outliers (out of which 108 severe), all on the higher end of the spectrum. We create an additional ordinal price factor "f.price" to create a discretisation according to the quartiles.

```
summary(df$price)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1275   14000   19799   21602   26000  149948

hist(df$price, breaks = 30, freq = F)
curve(dnorm(x, mean(df$price), sd(df$price)), add = T)
```

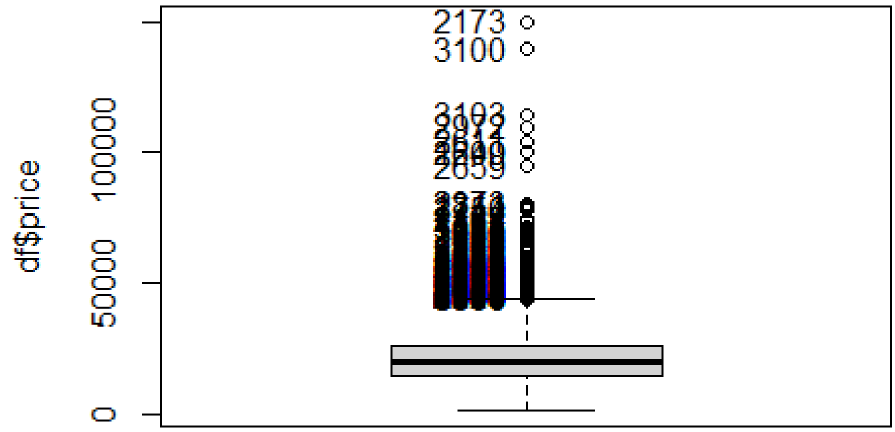




```
shapiro.test(df$price)

##
##  Shapiro-Wilk normality test
```

```
##
## data:  df$price
## W = 0.86882, p-value < 2.2e-16

sum(is.na(df$price))

## [1] 0

Boxplot(df$price)
```




```
##  [1] 2173 3100 3103 2972 2611 1609 2240 2659 2373 1811

length(Boxplot(df$price, id = list(n=Inf)))
```

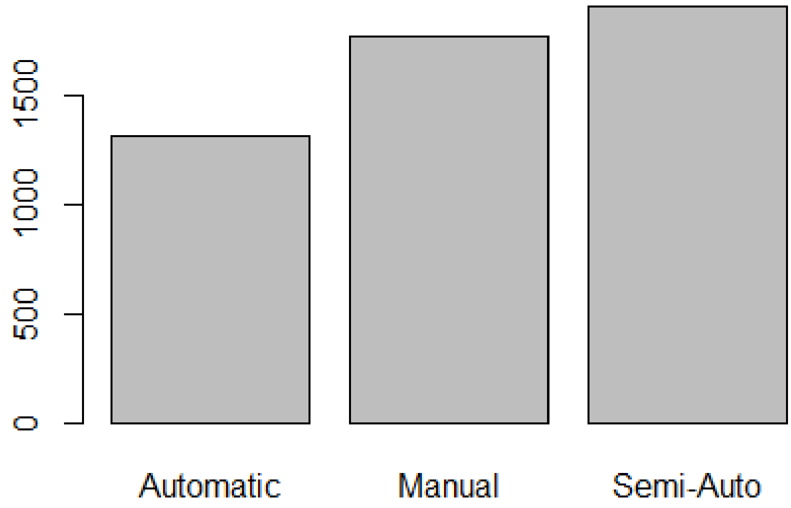



```
## [1] 207

sevout_price = (quantile(df$price,0.25)+(3*((quantile(df$price,0.75)-quantile(df$price,0.2
5)))))
length(which(df$price > sevout_price))

## [1] 108

df$f.price <- ifelse(df$price <= 14000, 1, ifelse(df$price > 14000 & df$price <= 19799, 2,
 ifelse(df$price > 19799 & df$price <= 26000, 3, ifelse(df$price > 26000, 4,0))))
df$f.price <- factor(df$f.price, labels=c("LowPrice","LowMidPrice","HighMidPrice","HighPri
ce"), order = T, levels=c(1,2,3,4))
table(df$f.price)

##
##     LowPrice  LowMidPrice HighMidPrice    HighPrice
##         1257         1244         1258         1241
```

variable 4: transmission

This is a Nominal variable (with three levels) and is thus converted to the factor type. It is visualized by a bar plot, in which it is clear that all levels are well represented. Therefore, no outliers are present. The variable contains no missing values thus imputation is not needed.

```
summary(df$transmission)

##    Length     Class      Mode
##      5000 character character

df$transmission = factor(df$transmission)
plot(df$transmission)
```

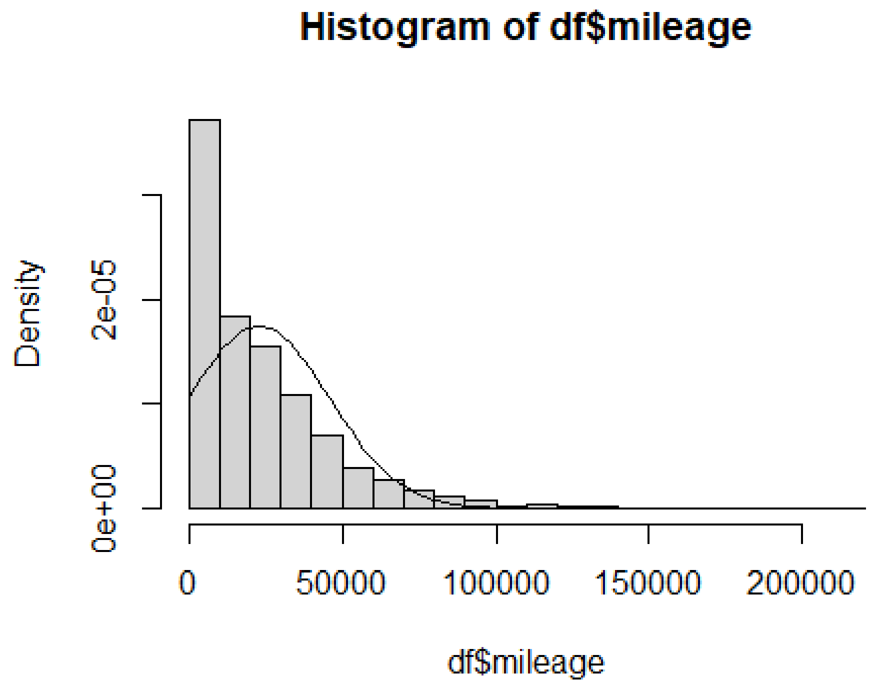



```
sum(is.na(df$transmission))

## [1] 0
```

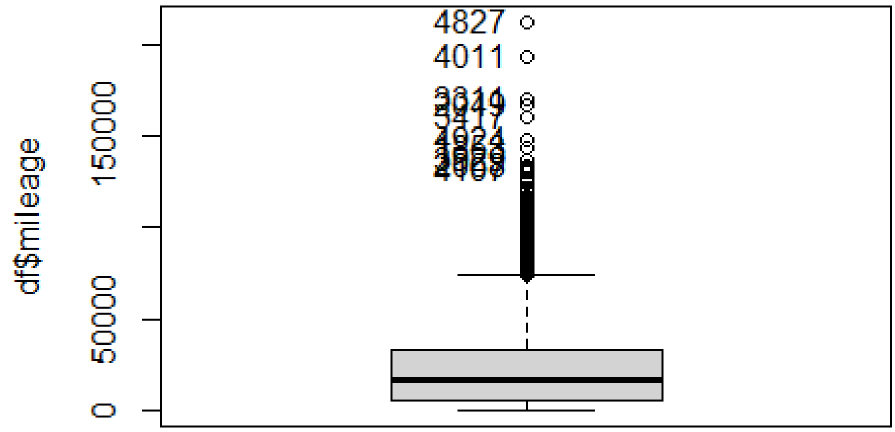

variable 5: mileage

This is a continuous ratio variable. The data does not look normally distributed, which is confirmed by the near-null p-value of the shapiro normallity test. Again a histogram is used to visualize the data. The variable contains no missing values thus imputation is not needed. It contains 188 outliers (out of which 98 severe), all on the higher end of the spectrum. We create an additional ordinal mileage factor "f.mileage" to create a discretisation according to the quartiles.

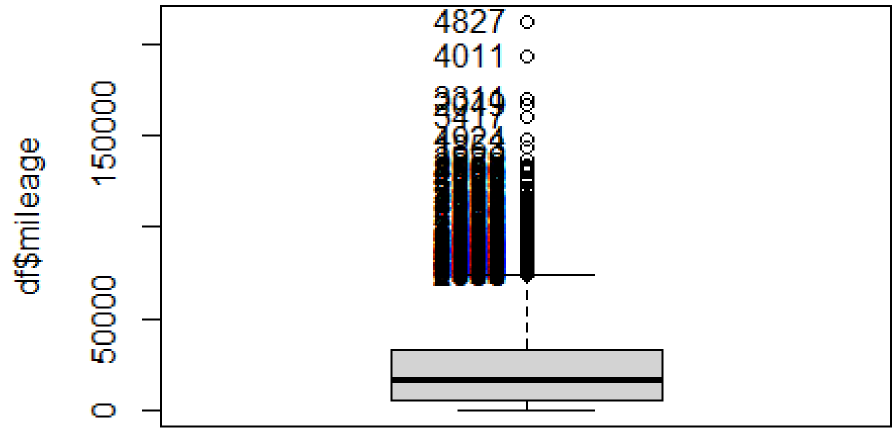
```
summary(df$mileage)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1    5728   16395   23042   33102  212000

hist(df$mileage, breaks = 30, freq = F)
curve(dnorm(x, mean(df$mileage), sd(df$mileage)), add = T)
```




```
shapiro.test(df$mileage)

##
##  Shapiro-Wilk normality test
##
## data:  df$mileage
## W = 0.83769, p-value < 2.2e-16

sum(is.na(df$mileage))

## [1] 0

Boxplot(df$mileage)
```




```
## [1] 4827 4011 3311 2049 3417 4924 1853 3929 2068 4107
length(Boxplot(df$mileage, id = list(n=Inf)))
```

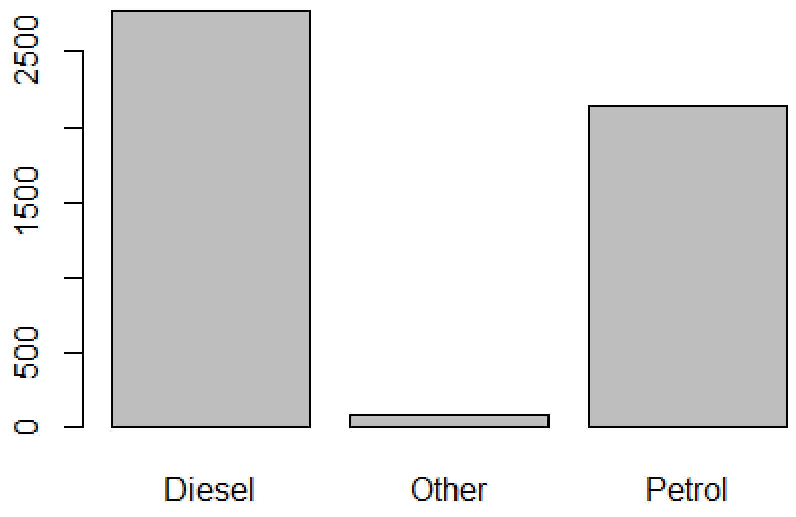

```
## [1] 188
sevout_mileage = (quantile(df$mileage,0.25)+(3*((quantile(df$mileage,0.75)-quantile(df$mil
eage,0.25)))))
length(which(df$mileage > sevout_mileage))
```

```
## [1] 98

df$f.mileage <- ifelse(df$mileage <= 5728, 1, ifelse(df$mileage > 5728 & df$mileage <= 163
95, 2, ifelse(df$mileage > 16395 & df$mileage <= 33102, 3, ifelse(df$mileage > 33102, 4,0)
)))
df$f.mileage <- factor(df$f.mileage,labels=c("LowMileage","LowMidMileage","HighMidMileage"
,"HighMileage"), order = T, levels=c(1,2,3,4))
table(df$f.mileage)

##
##     LowMileage  LowMidMileage HighMidMileage    HighMileage
##           1250           1250           1250           1250
```

variable 6: fuelType

This is a nominal variable with 5 levels in which 'electric', 'hybrid' and 'other' only combine for less then 2% of the instances combined, such that these are all collapsed into the 'other' level. The variable contains no missing values thus imputation is not needed. A bar plot is used to plot the variable.

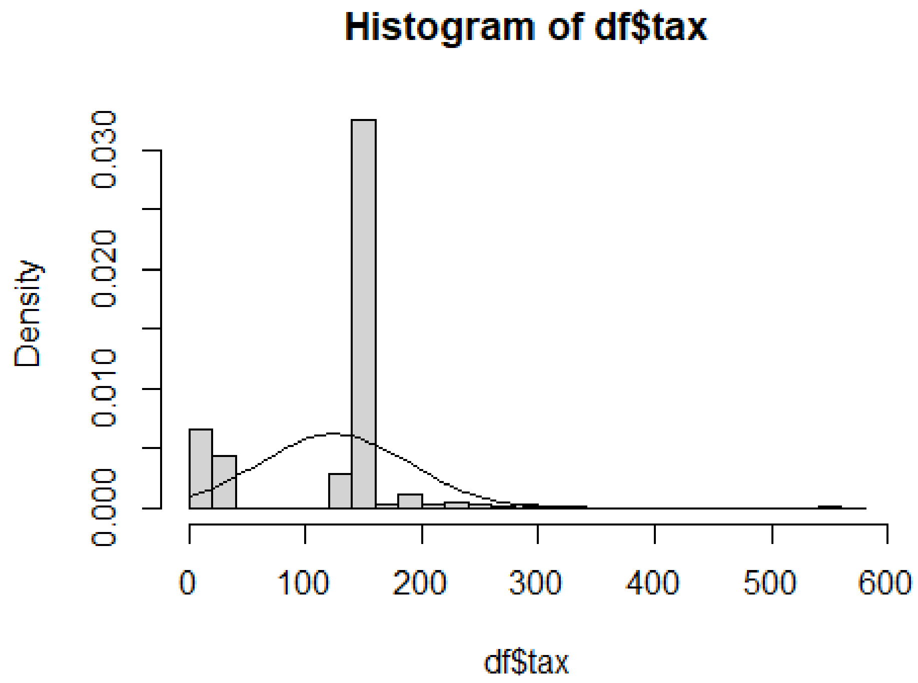
```
summary(df$fuelType)

##    Length     Class      Mode
##      5000 character character

elec_idx <- which(df$fuelType == 'Electric')
prop.table(table(df$fuelType))

##
##   Diesel Electric   Hybrid    Other   Petrol
##   0.5548   0.0002   0.0122   0.0036   0.4292

df$fuelType[which(df$fuelType == 'Hybrid')] = 'Other'
df$fuelType[which(df$fuelType == 'Electric')] = 'Other'
df$fuelType = factor(df$fuelType)

plot(df$fuelType)
```



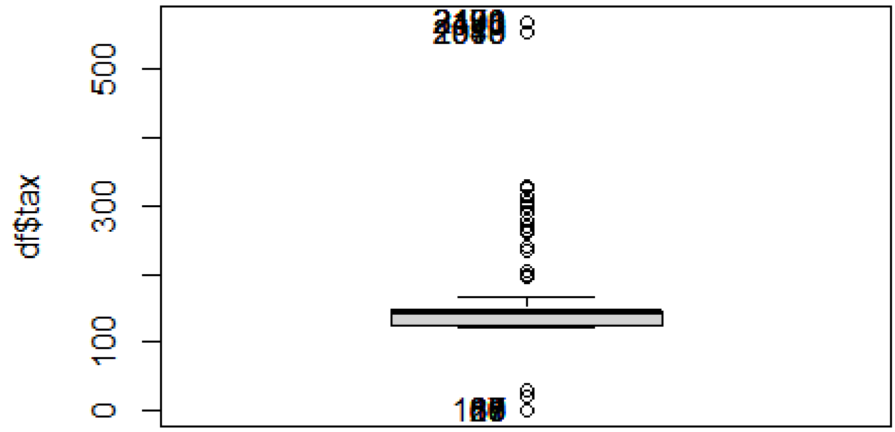
variable 7: tax

This is a continuous ratio variable. The data does not look normally distributed, which is confirmed by the near-null p-value of the shapiro normallity test. Again a histogram is used to visualize the data. The variable contains no missing values thus imputation is not needed. It contains 1422 outliers (out of which all severe), on both sides of the spectrum. We create an additional ordinal tax factor "f.tax" to create a discretisation according to the quartiles.

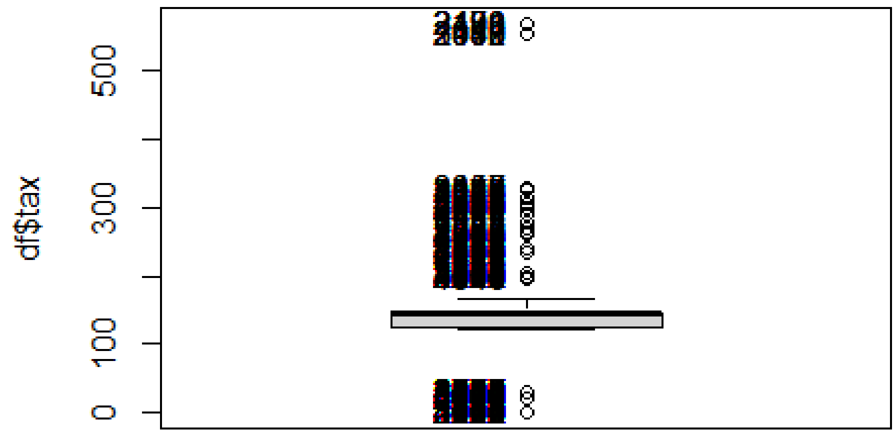
```
summary(df$tax)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##     0.0   125.0   145.0   124.9   145.0   570.0

hist(df$tax, breaks = 30, freq = F)
curve(dnorm(x, mean(df$tax), sd(df$tax)), add = T)
```




```
shapiro.test(df$tax)

##
##  Shapiro-Wilk normality test
##
## data:  df$tax
## W = 0.72815, p-value < 2.2e-16

sum(is.na(df$tax))

## [1] 0

Boxplot(df$tax)
```



```
## [1]    7    9   17   21   66   67   85   88   89  136 2173 2180 2191 2198 3430
## [16] 3431  979 1543 2010 2088

length(Boxplot(df$tax, id = list(n=Inf)))
```

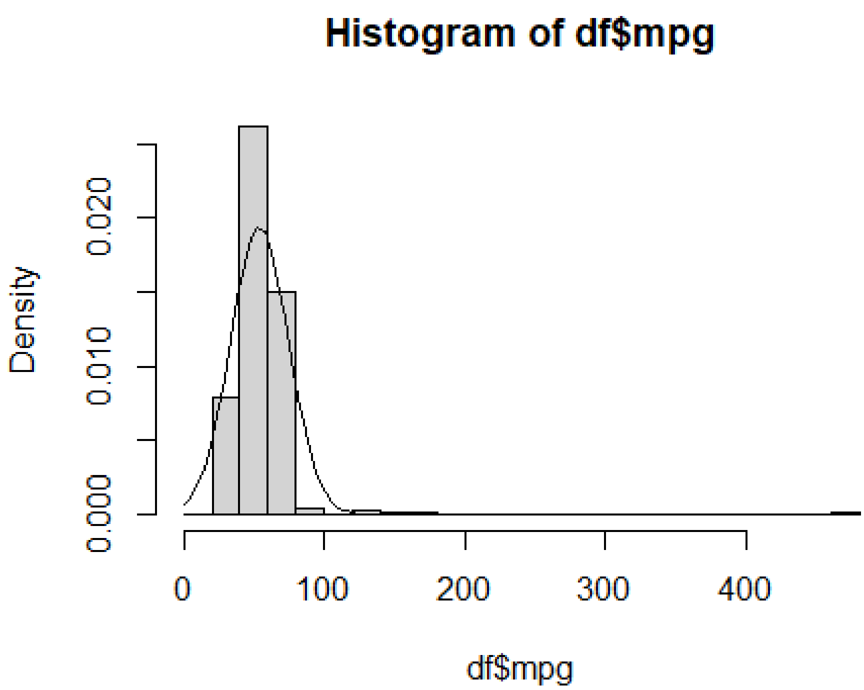
```
## [1] 1422

sevout_tax_upp = (quantile(df$tax,0.25)+(3*((quantile(df$tax,0.75)-quantile(df$tax,0.25)))
))
```

```
sevout_tax_low = (quantile(df$tax,0.25)-(3*((quantile(df$tax,0.75)-quantile(df$tax,0.25)))
))
length(which(df$tax > sevout_tax_upp))+length(which(df$tax < sevout_tax_low))

## [1] 1422

df$f.tax <- ifelse(df$tax <= 125, 1, ifelse(df$tax > 125 & df$tax < 145, 2, ifelse(df$tax
== 145, 3, ifelse(df$tax > 145, 4,0))))
df$f.tax <- factor(df$f.tax,labels=c("Lowtax","LowMidtax","HighMidtax","Hightax"), order =
 T, levels=c(1,2,3,4))
table(df$f.tax)

##
##     Lowtax  LowMidtax HighMidtax    Hightax
##       1349         34       2610       1007
```

variable 8: mpg

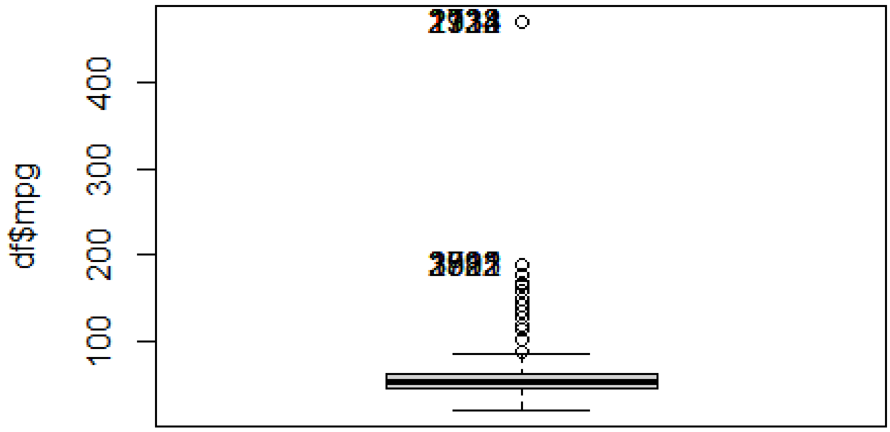
This is a continuous ratio variable. The data does not look normally distributed, which is confirmed by the near-null p-value of the shapiro normallity test. Again a histogram is used to visualize the data. The variable contains no missing values thus imputation is not needed. It contains 56 outliers (out of which 53 severe), all on the high side of the spectrum. We create an additional ordinal mpg factor "f.mpg" to create a discretisation according to the quartiles.

```
summary(df$mpg)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   19.50   44.10   52.80   53.96   61.40  470.80

hist(df$mpg, breaks = 30, freq = F)
curve(dnorm(x, mean(df$mpg), sd(df$mpg)), add = T)
```

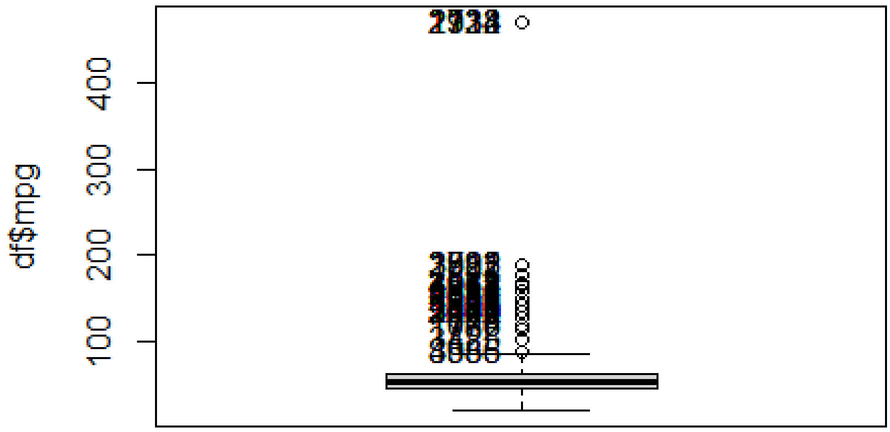

```
shapiro.test(df$mpg)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df$mpg
## W = 0.52394, p-value < 2.2e-16

sum(is.na(df$mpg))

## [1] 0

Boxplot(df$mpg)
```



18

```
##  [1] 1133 1324 1732 1934 2114 2138 1582 2995 3722 3811

length(Boxplot(df$mpg, id = list(n=Inf)))
```



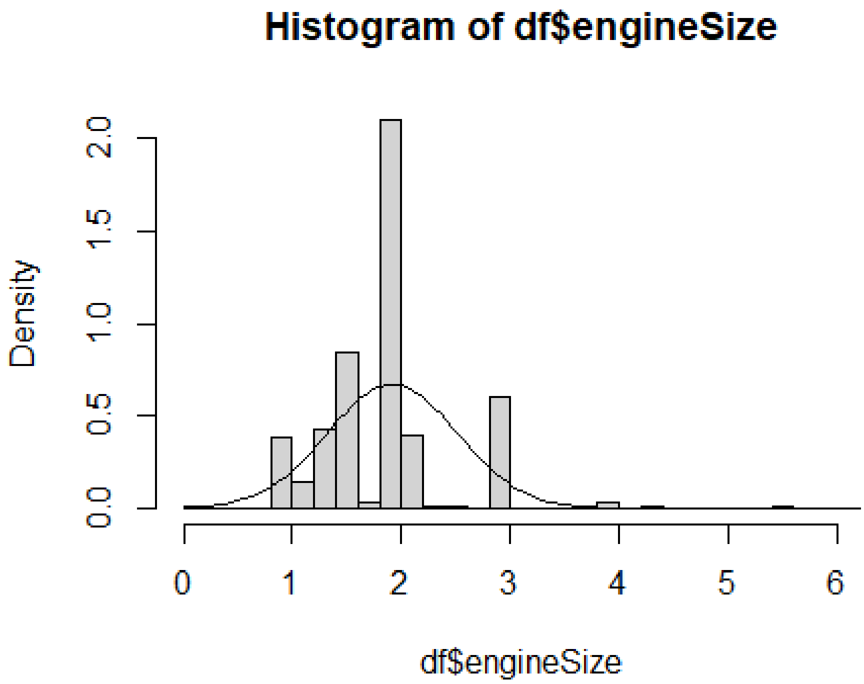

```
## [1] 56

sevout_mpg = (quantile(df$mpg,0.25)+(3*((quantile(df$mpg,0.75)-quantile(df$mpg,0.25)))))
length(which(df$mpg > sevout_mpg))

## [1] 53

df$f.mpg <- ifelse(df$mpg <= 44.1, 1, ifelse(df$mpg > 44.1 & df$mpg <= 52.8, 2, ifelse(df$
mpg > 52.8 & df$mpg <= 61.4, 3, ifelse(df$mpg > 61.4, 4,0))))
df$f.mpg <- factor(df$f.mpg,labels=c("Lowmpg","LowMidmpg","HighMidmpg","Highmpg"), order =
 T, levels=c(1,2,3,4))
table(df$f.mpg)

##
##     Lowmpg  LowMidmpg HighMidmpg    Highmpg
##       1257       1243       1342       1158
```


variable 9: engineSize

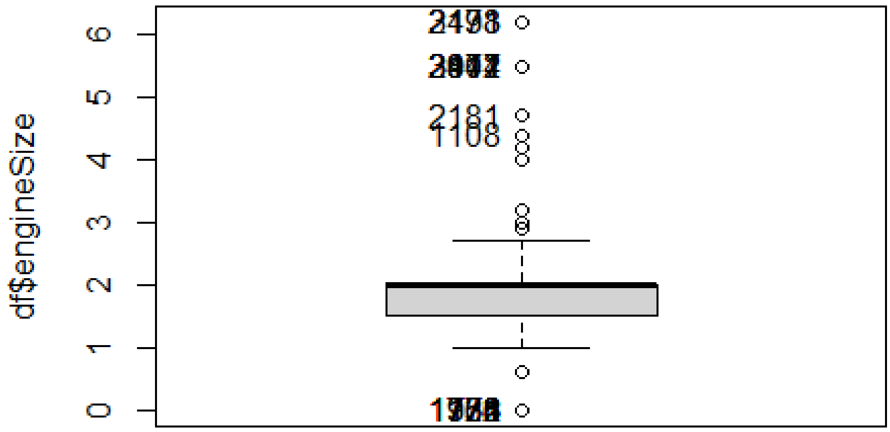
This is an interval variable. It contains 673 outliers, out of which 55 severe. There are 15 instances of cars without enginesize which seems like missing values. As it was stated in the data description, these instance could denote electric fuelTypes, however, after inspecting each case more closely it appeared that only 1 of these truely denotes an electric engine and the others are missing values at random (MAR). As the hybrid and electric fuel types were previously set to other, all engine sizes which have values equal to 0 are set to NA and then imputed using the MICE algorithm (an algorithm using chained equations using k-Nearest-Neighbour and regression techniques), except for the electric fuel type. We create an additional ordinal enginesize factor “f.engineSize” to create a discretisation according to the quartiles.

```
summary(df$engineSize)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.500   2.000   1.918   2.000   6.200
```

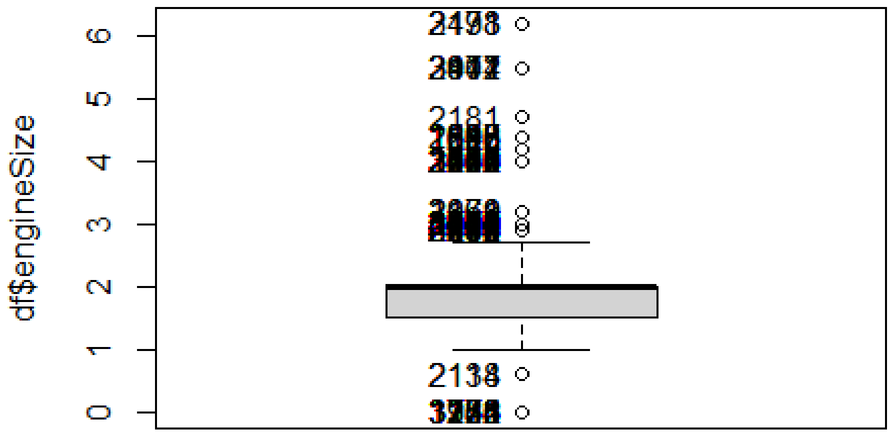
```
hist(df$engineSize, breaks = 30, freq = F)
curve(dnorm(x, mean(df$engineSize), sd(df$engineSize)), add = T)
```




```
shapiro.test(df$engineSize)

##
##  Shapiro-Wilk normality test
##
## data:  df$engineSize
## W = 0.86113, p-value < 2.2e-16

sum(is.na(df$engineSize))

## [1] 0

Boxplot(df$engineSize)
```



```
## [1]  758  768  771  772  773  776 1133 1324 1732 1934 2173 2198 3431 2812 2944
## [16] 3077 3111 3401 2181 1108

length(Boxplot(df$engineSize, id = list(n=Inf)))
```

```
## [1] 673

sevout_engineSize_upp = (quantile(df$engineSize,0.25)+(3*((quantile(df$engineSize,0.75)-qu
antile(df$engineSize,0.25)))))
```

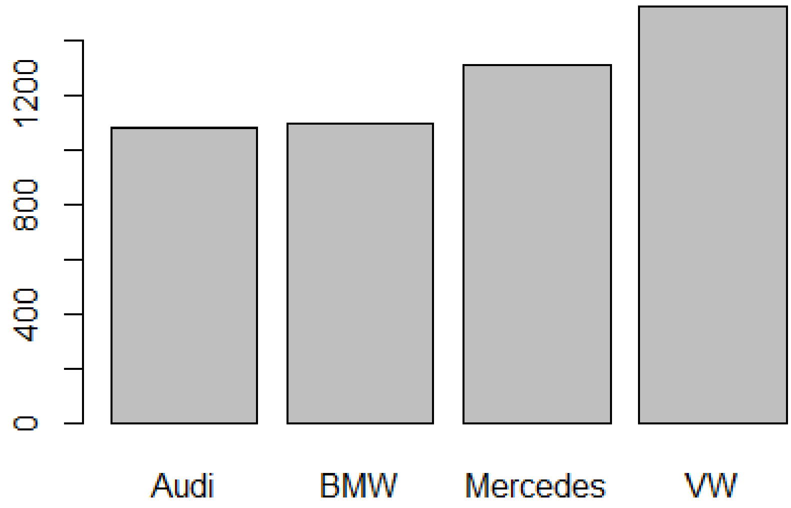
```
sevout_engineSize_down = (quantile(df$engineSize,0.25)-(3*((quantile(df$engineSize,0.75)-q
uantile(df$engineSize,0.25)))))
length(which(df$engineSize > sevout_engineSize_upp | df$engineSize < sevout_engineSize_dow
n))

## [1] 55

df$engineSize[which(df$engineSize == 0)] = NA
df$engineSize[elec_idx] = 0
require(mice)

## Loading required package: mice

## Warning: package 'mice' was built under R version 4.0.5

##
## Attaching package: 'mice'

## The following object is masked from 'package:stats':
##
##     filter

## The following objects are masked from 'package:base':
##
##     cbind, rbind

options(contrasts = c("contr.treatment", "contr.treatment")) ##set options to contr.treatm
ent as mice can only use lm when options is set to this.
imputation = mice(df, method = 'pmm')

##
##  iter imp variable
##   1   1  engineSize*
##   1   2  engineSize*
##   1   3  engineSize*
##   1   4  engineSize*
##   1   5  engineSize*
##   2   1  engineSize*
##   2   2  engineSize*
##   2   3  engineSize*
##   2   4  engineSize*
##   2   5  engineSize*
##   3   1  engineSize*
##   3   2  engineSize*
##   3   3  engineSize*
##   3   4  engineSize*
##   3   5  engineSize*
##   4   1  engineSize*
##   4   2  engineSize*
##   4   3  engineSize*
##   4   4  engineSize*
##   4   5  engineSize*
##   5   1  engineSize*
##   5   2  engineSize*
##   5   3  engineSize*
##   5   4  engineSize*
##   5   5  engineSize*

## Warning: Number of logged events: 28

imputation$imp$engineSize

##        1   2   3   4   5
## 7519  2.0 2.0 1.5 1.5 2.0
## 7599  1.4 1.2 1.4 1.4 1.6
## 7632  1.4 2.0 1.4 2.0 1.6
## 7645  2.5 4.0 3.0 4.0 2.0
```

```
## 7660  2.0 3.0 2.0 2.0 2.0
## 7701  1.6 1.6 2.0 1.6 2.0
## 11290 0.6 1.0 0.6 0.0 0.6
## 13021 0.6 1.0 0.6 0.6 0.0
## 16949 0.6 0.6 0.6 0.6 1.0
## 31068 1.5 2.0 2.0 2.0 3.0
## 31069 1.6 1.4 1.6 1.6 2.0
## 31071 2.0 2.0 2.0 2.0 3.0
## 32125 2.0 2.0 1.5 2.1 2.0
## 32189 2.0 2.0 2.0 2.0 2.0

df = complete(imputation, 5)
summary(df$engineSize)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.500   2.000   1.923   2.000   6.200

df$f.engineSize <- ifelse(df$engineSize <= 1.5, 1, ifelse(df$engineSize > 1.5 & df$engineS
ize < 2, 2, ifelse(df$engineSize >= 2 & df$engineSize <= 2, 3, ifelse(df$engineSize > 2, 4
,0))))
df$f.engineSize <- factor(df$f.engineSize,labels=c("LowengineSize","LowMidengineSize","Hig
hMidengineSize","HighengineSize"), order = T, levels=c(1,2,3,4))
table(df$f.engineSize)

## 
##    LowengineSize  LowMidengineSize HighMidengineSize    HighengineSize 
##             1492               333              2108              1067
```

variable 10: manufacturer

This is a Nominal variable (with four levels) and is thus converted to the factor type. It is visualized by a bar plot, in which it is clear that all levels are well represented. Therefore, no outliers are present. The variable contains no missing values thus imputation is not needed.

```
table(df$manufacturer)

## 
##     Audi      BMW Mercedes       VW 
##     1075     1096     1308     1521

df$manufacturer = factor(df$manufacturer)
plot(df$manufacturer)
```




```
sum(is.na(df$manufacturer))

## [1] 0
```


Data quality report

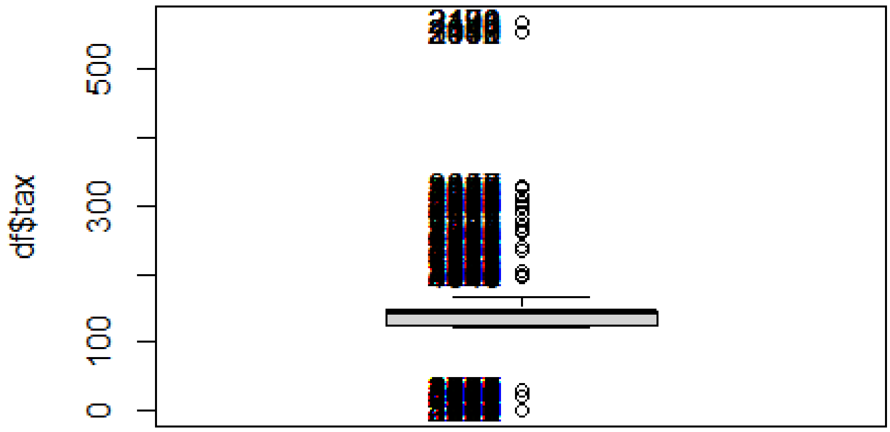
Variables

Now that all the variables have been explored, their general quality can be reported on. In terms of missingness, only the engineSize shows missing data and hence, it ranks last in terms of missingness. If we rank the numeric variables in terms of number of outliers, the following list is obtained, starting with the most outliers: Tax, engine size, price, mileage, year and mpg. Combining these two results, it seems like both tax and engine size are the variables containing the most noise in this data set.

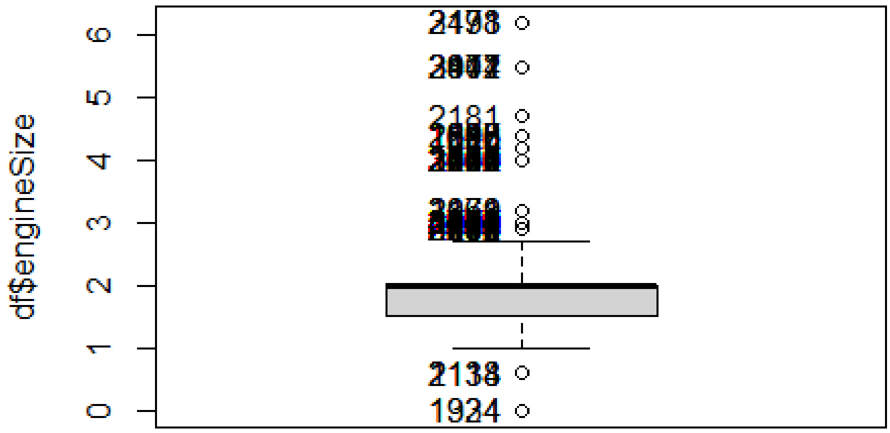
individuals

Now the individuals are investigated. First the number of univariate outliers per individual are counted and added in a new variable called 'univ_outl_count'. Looking at the 8 individuals with the most univariate outliers (4) it can be concluded that they are all old, highly taxed, low mpg, high engine size and most with a low price and high mileage. A correlation matrix confirms this as it shows a significant negative correlation to the year (so the older the car, the more univ outliers) and a significant positive correlation to both mileage and engine size.

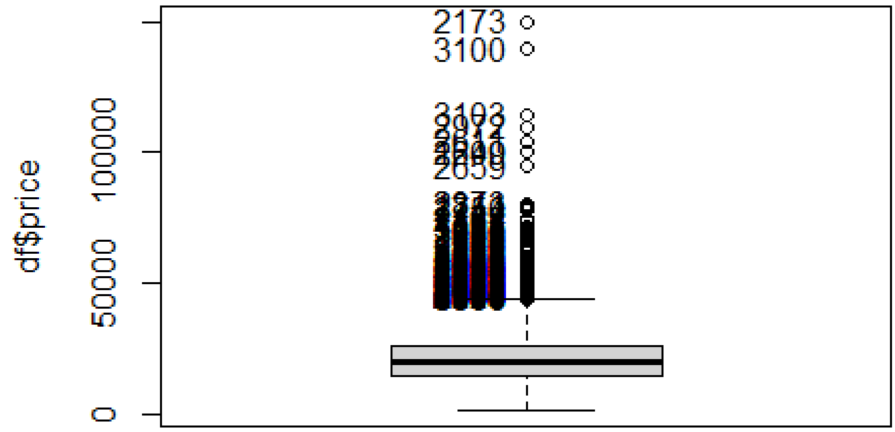
```
df$univ_outl_count <- 0
df$univ_outl_count[Boxplot(df$tax, id = list(n=Inf))] = df$univ_outl_count[Boxplot(df$tax,
 id = list(n=Inf))] + 1
```



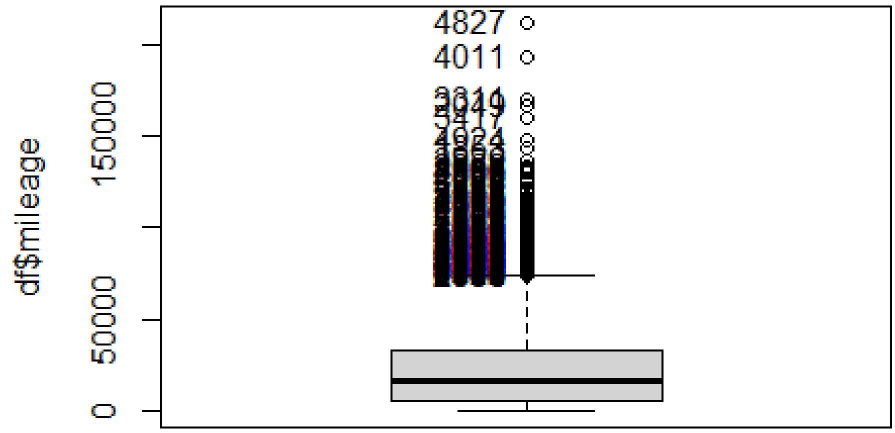

```
df$univ_outl_count[Boxplot(df$engineSize, id = list(n=Inf))] = df$univ_outl_count[Boxplot(
df$engineSize, id = list(n=Inf))] + 1
```

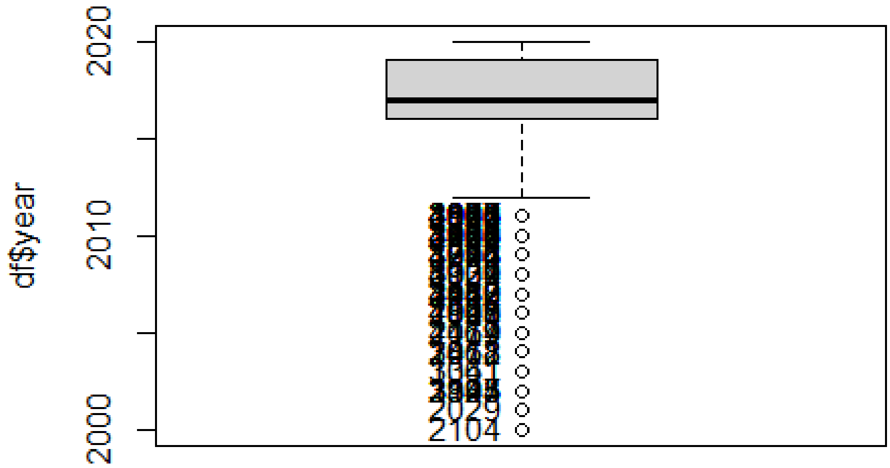

```
df$univ_outl_count[Boxplot(df$price, id = list(n=Inf))] = df$univ_outl_count[Boxplot(df$pr
ice, id = list(n=Inf))] + 1
```



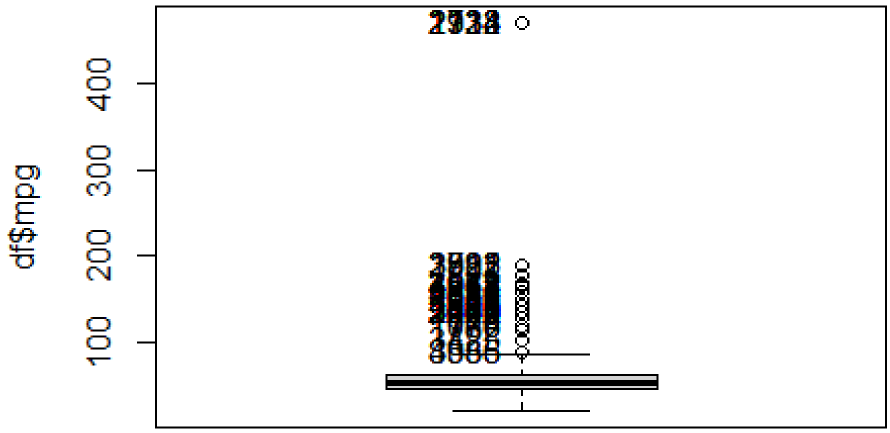

```r
df$univ_outl_count[Boxplot(df$mileage, id = list(n=Inf))] = df$univ_outl_count[Boxplot(df$mileage, id = list(n=Inf))] + 1
```


```r
df$univ_outl_count[Boxplot(df$year, id = list(n=Inf))] = df$univ_outl_count[Boxplot(df$year, id = list(n=Inf))] + 1
```

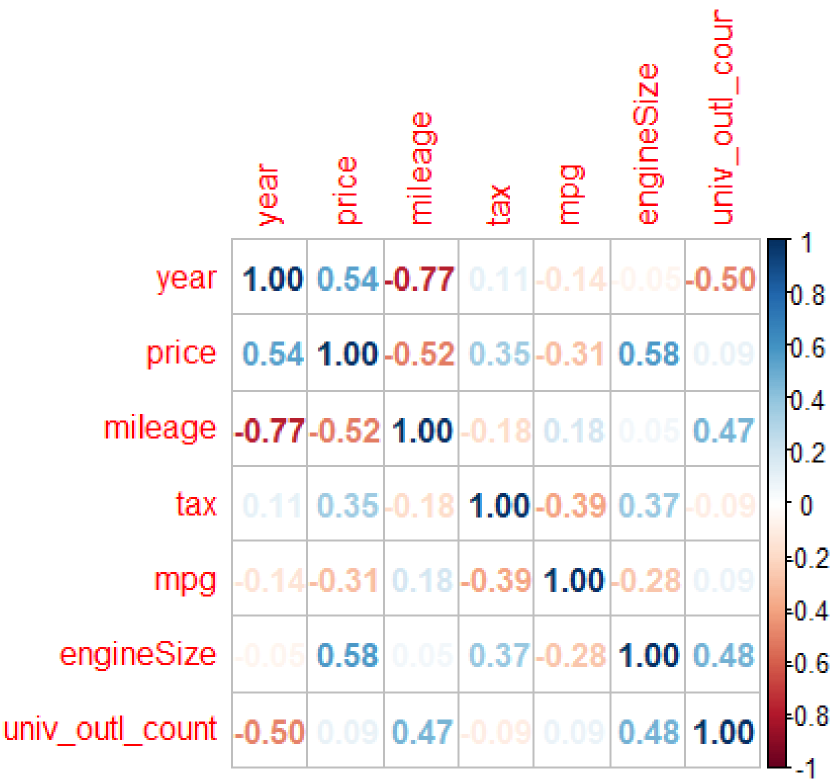



```
df$univ_outl_count[Boxplot(df$mpg, id = list(n=Inf))] = df$univ_outl_count[Boxplot(df$mpg,
 id = list(n=Inf))] + 1
```


```
max(df$univ_outl_count)

## [1] 4

df[which(df$univ_outl_count == 4),]
```

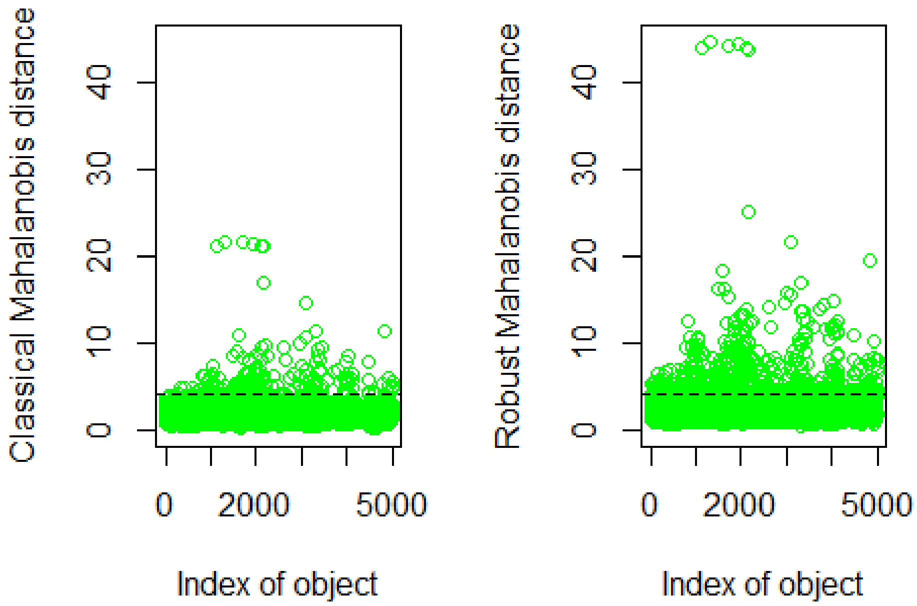
```
##          model year  price transmission mileage fuelType tax  mpg engineSize
## 1014        A4 2005   4990       Manual   87990   Diesel 325 36.7        3.0
## 1023        A8 2006   5595    Automatic  104000   Diesel 325 33.6        3.0
## 2010  6 Series 2006   4999    Automatic  126054   Petrol 555 29.7        3.0
## 2029  3 Series 2001   3050    Automatic   90000   Petrol 325 27.7        3.0
## 2130  3 Series 2008   8790       Manual   85000   Petrol 555 28.5        3.0
## 2169        M3 2005  10999       Manual  115000   Petrol 315 20.8        3.2
## 2173  SL CLASS 2011 149948    Automatic    3000   Petrol 570 21.4        6.2
## 3296   E Class 2009   3995    Automatic  131711   Diesel 300 27.7        3.0
##      manufacturer n.age   f.age   f.price   f.mileage   f.tax  f.mpg
## 1014         Audi    15 HighAge  LowPrice HighMileage Hightax Lowmpg
## 1023         Audi    14 HighAge  LowPrice HighMileage Hightax Lowmpg
## 2010          BMW    14 HighAge  LowPrice HighMileage Hightax Lowmpg
## 2029          BMW    19 HighAge  LowPrice HighMileage Hightax Lowmpg
## 2130          BMW    12 HighAge  LowPrice HighMileage Hightax Lowmpg
## 2169          BMW    15 HighAge  LowPrice HighMileage Hightax Lowmpg
## 2173     Mercedes     9 HighAge HighPrice  LowMileage Hightax Lowmpg
## 3296     Mercedes    11 HighAge  LowPrice HighMileage Hightax Lowmpg
##        f.engineSize univ_outl_count
## 1014 HighengineSize               4
## 1023 HighengineSize               4
## 2010 HighengineSize               4
## 2029 HighengineSize               4
## 2130 HighengineSize               4
## 2169 HighengineSize               4
## 2173 HighengineSize               4
## 3296 HighengineSize               4

df_of_interest = df[,c(2,3,5,7,8,9,18)]
cor_outl = cor(df_of_interest)
require(corrplot)

## Loading required package: corrplot

## corrplot 0.92 loaded

par(mfrow=c(1,1))
corrplot(cor_outl, method = 'number')
```

Multivariate Outliers

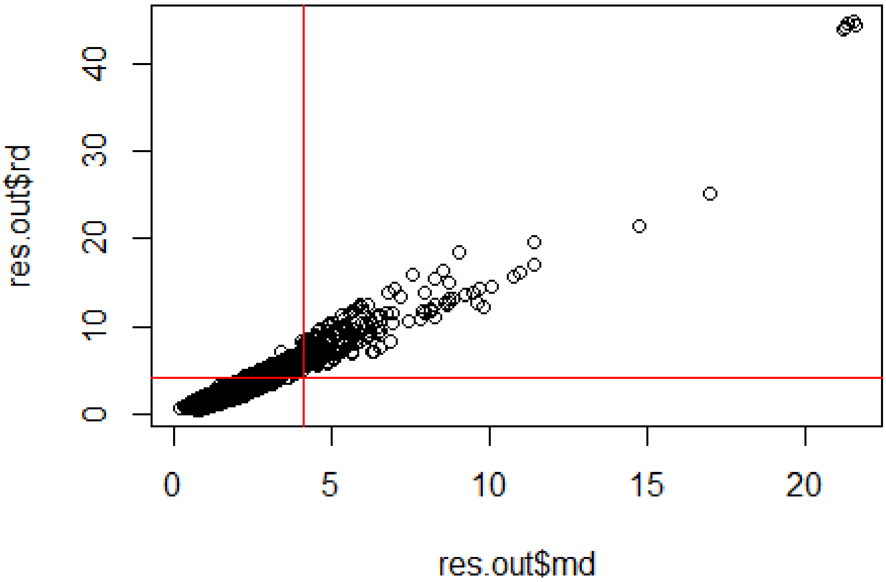
Moutlier is applied on the numerical variables to find multivariate outliers. With the tax variable included, however, the method returns a singular matrix. Therefore this variable is excluded from the calculation. A very mild threshold of 0.5 % is chosen as signficance level because is already returns a significant amount of outliers; This makes up around 4% of the total amount of instances. It is chosen to delete these outliers from the data set for the rest of the project.

```
require(chemometrics)

## Loading required package: chemometrics

## Loading required package: rpart

res.out = Moutlier(df[,c(2,3,5,8,9)], quantile = 0.995, col="green")
```




```
which(res.out$md > res.out$cutoff)

##   [1]  268  306  409  603  740  776  825  848  888  897  928  929  969  976  979
##  [16]  992  997 1000 1004 1005 1007 1014 1023 1032 1038 1058 1061 1108 1122 1133
##  [31] 1141 1154 1217 1258 1299 1324 1445 1508 1523 1543 1549 1554 1582 1609 1638
##  [46] 1640 1642 1648 1683 1684 1687 1689 1697 1714 1732 1741 1749 1750 1768 1806
##  [61] 1811 1835 1837 1839 1845 1846 1847 1850 1853 1861 1881 1892 1906 1934 1939
##  [76] 1954 1956 1981 1986 1993 2009 2010 2029 2035 2045 2048 2049 2058 2068 2078
##  [91] 2082 2088 2096 2103 2104 2105 2106 2114 2130 2138 2167 2169 2172 2173 2180
## [106] 2181 2182 2191 2198 2233 2240 2262 2276 2352 2373 2467 2510 2562 2609 2611
## [121] 2633 2649 2659 2744 2771 2812 2944 2952 2972 2995 3013 3077 3087 3100 3103
## [136] 3111 3148 3198 3204 3213 3238 3265 3284 3296 3311 3325 3328 3329 3331 3332
## [151] 3341 3343 3344 3401 3412 3417 3425 3430 3431 3466 3578 3722 3811 3831 3926
## [166] 3929 3947 3971 3978 3982 3995 4011 4034 4046 4048 4064 4100 4101 4107 4109
## [181] 4110 4113 4456 4462 4479 4628 4633 4794 4827 4842 4922 4924 4929 4966 4995
## [196] 4998 5000

length(which(res.out$md > res.out$cutoff))/5000
```

```
## [1] 0.0394

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




```
summary(df[which(res.out$md > res.out$cutoff),])

##     model                year          price           transmission
##  Length:197         Min.   :2000   Min.   :  1275   Automatic:97
##  Class :character   1st Qu.:2010   1st Qu.:  7000   Manual   :53
##  Mode  :character   Median :2016   Median : 17950   Semi-Auto:47
##                     Mean   :2014   Mean   : 28136
##                     3rd Qu.:2017   3rd Qu.: 44450
##                     Max.   :2020   Max.   :149948
##     mileage         fuelType       tax              mpg           engineSize
##  Min.   :    16   Diesel:68   Min.   :  0.0   Min.   : 19.5   Min.   :0.000
##  1st Qu.:  9377   Other :50   1st Qu.:125.0   1st Qu.: 31.0   1st Qu.:2.000
##  Median : 43000   Petrol:79   Median :145.0   Median : 45.6   Median :2.000
##  Mean   : 57579               Mean   :176.3   Mean   : 77.8   Mean   :2.523
##  3rd Qu.: 99217               3rd Qu.:260.0   3rd Qu.:117.7   3rd Qu.:3.000
##  Max.   :212000               Max.   :570.0   Max.   :470.8   Max.   :6.200
##    manufacturer      n.age            f.age             f.price
##  Audi     :27    Min.   : 0.000   LowAge     :38   LowPrice    :86
##  BMW      :75    1st Qu.: 3.000   LowMidAge  :42   LowMidPrice :27
##  Mercedes:58     Median : 4.000   HighMidAge:26    HighMidPrice:21
##  VW       :37    Mean   : 6.061   HighAge    :91   HighPrice   :63
##                  3rd Qu.:10.000
##                  Max.   :20.000
##           f.mileage          f.tax           f.mpg               f.engineSize
##  LowMileage    : 36   Lowtax     :53   Lowmpg     :96   LowengineSize     :32
##  LowMidMileage : 22   LowMidtax  :25   LowMidmpg  :15   LowMidengineSize  :11
##  HighMidMileage: 28   HighMidtax:44    HighMidmpg:19    HighMidengineSize:69
##  HighMileage   :111   Hightax    :75   Highmpg    :67   HighengineSize    :85
##
##
##  univ_outl_count
```

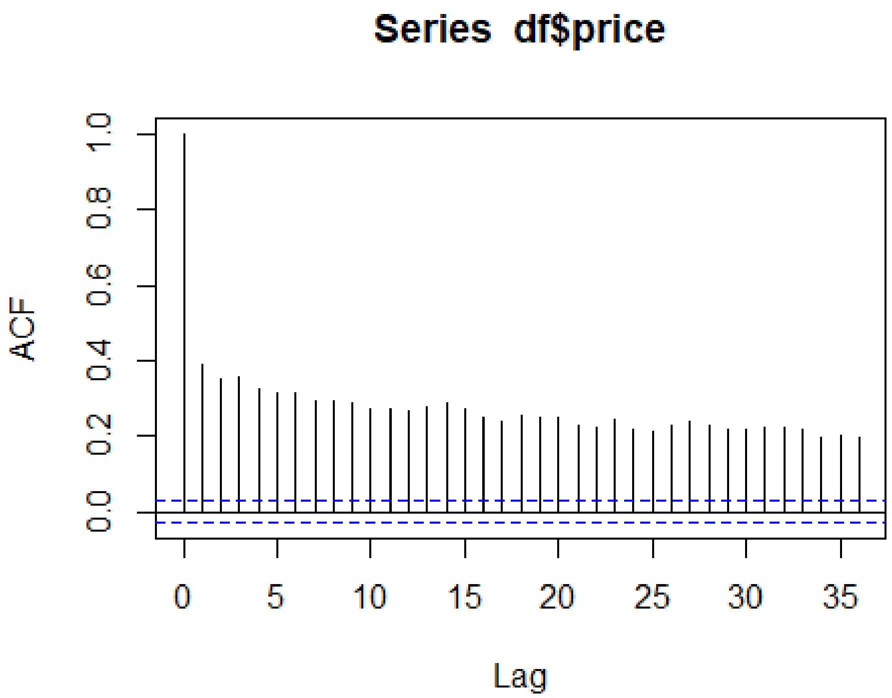
```
##  Min.   :0.000  
##  1st Qu.:2.000  
##  Median :2.000  
##  Mean   :2.102  
##  3rd Qu.:3.000  
##  Max.   :4.000  

summary(df)

##     model                year          price           transmission 
##  Length:5000        Min.   :2000   Min.   :  1275   Automatic:1317  
##  Class :character   1st Qu.:2016   1st Qu.: 14000   Manual   :1775  
##  Mode  :character   Median :2017   Median : 19799   Semi-Auto:1908  
##                     Mean   :2017   Mean   : 21602                   
##                     3rd Qu.:2019   3rd Qu.: 26000                   
##                     Max.   :2020   Max.   :149948                   
##     mileage          fuelType          tax              mpg        
##  Min.   :     1   Diesel:2774   Min.   :  0.0   Min.   : 19.50  
##  1st Qu.:  5728   Other :  80   1st Qu.:125.0   1st Qu.: 44.10  
##  Median : 16395   Petrol:2146   Median :145.0   Median : 52.80  
##  Mean   : 23042                 Mean   :124.9   Mean   : 53.96  
##  3rd Qu.: 33102                 3rd Qu.:145.0   3rd Qu.: 61.40  
##  Max.   :212000                 Max.   :570.0   Max.   :470.80  
##    engineSize      manufacturer      n.age                 f.age     
##  Min.   :0.000   Audi    :1075   Min.   : 0.000   LowAge     :1938  
##  1st Qu.:1.500   BMW     :1096   1st Qu.: 1.000   LowMidAge  :1426  
##  Median :2.000   Mercedes:1308   Median : 3.000   HighMidAge: 798  
##  Mean   :1.923   VW      :1521   Mean   : 2.788   HighAge    : 838  
##  3rd Qu.:2.000                   3rd Qu.: 4.000                     
##  Max.   :6.200                   Max.   :20.000                     
##          f.price              f.mileage              f.tax              f.mpg     
##  LowPrice    :1257   LowMileage    :1250   Lowtax    :1349   Lowmpg    :1257  
##  LowMidPrice :1244   LowMidMileage :1250   LowMidtax :  34   LowMidmpg :1243  
##  HighMidPrice:1258   HighMidMileage:1250   HighMidtax:2610   HighMidmpg:1342  
##  HighPrice   :1241   HighMileage   :1250   Hightax   :1007   Highmpg   :1158  
##                                                                                  
##                                                                                  
##             f.engineSize  univ_outl_count 
##  LowengineSize    :1492   Min.   :0.0000  
##  LowMidengineSize : 333   1st Qu.:0.0000  
##  HighMidengineSize:2108   Median :0.0000  
##  HighengineSize   :1067   Mean   :0.5232  
##                           3rd Qu.:1.0000  
##                           Max.   :4.0000  

df = df[-which(res.out$md > res.out$cutoff),]
```

Profiling

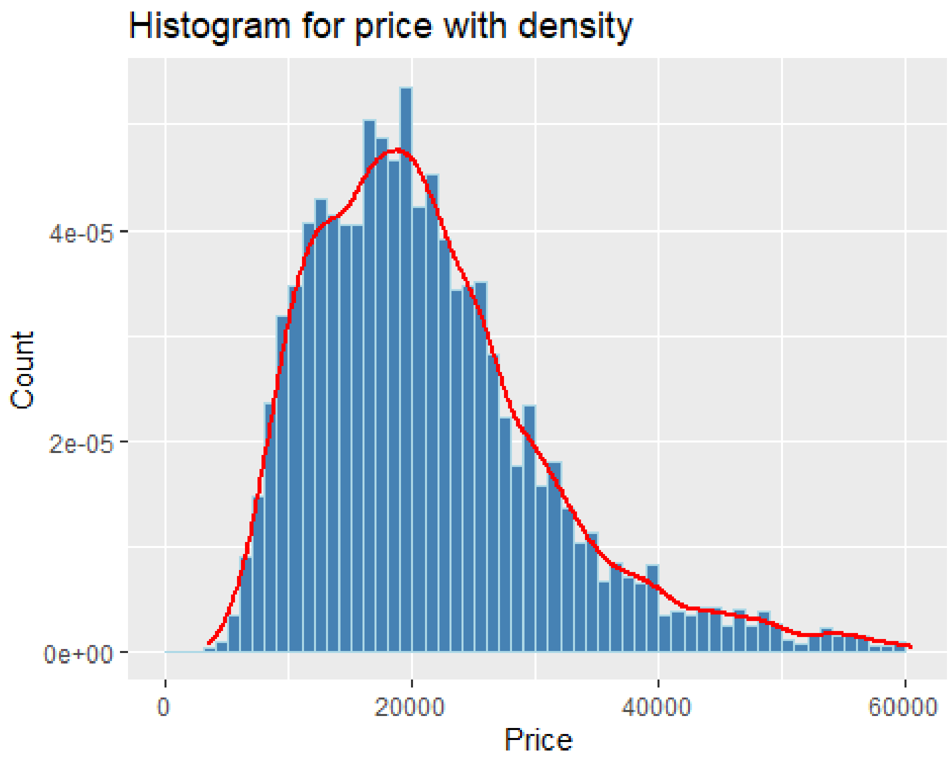
Determine if the response variable (price) has an acceptably normal distribution. Address test to discard serial correlation.

To test for autocorrelation the acf() function is used, of which the result can be seen below.

```
acf(df$price)
```




```
shapiro.test(df$price)

##
##  Shapiro-Wilk normality test
##
## data:  df$price
## W = 0.93366, p-value < 2.2e-16

ggplot(data=df, aes(price, y = ..density..)) +
  geom_histogram(breaks=seq(0, max(df$price), by=1000),
                 col='lightblue',
                 fill='steelblue') +
  geom_density(lwd=1,
               col='red') +
  labs(title="Histogram for price with density", x="Price", y="Count")
```



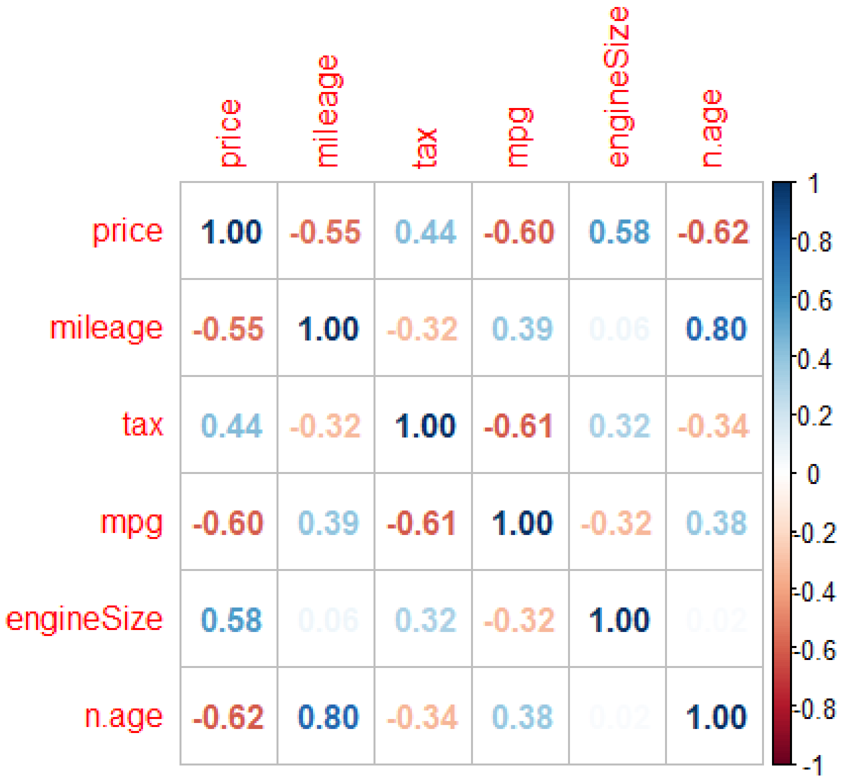
Indicate by exploration of the data which are apparently the variables most associated with the response variable (use only the indicated variables).

To do this, the condes function of the FactoMiner package is used, which for the numeric response variable 'price' calculates the correlation of each of the quantitative variables and the coefficient of determination (R^2) for the qualitative variables, together with a p-value for significance.

For the quantitative variables it is clear both engineSize and year are highly significant positively correlated (r > 0.50, p = 0) to the price. This seems logical as the higher the newer the car is and the bigger the engine, the more it would cost. The mileage and miles per gallon (mpg) are highly significant negatively correlated to the price (r < -0.50 p = 0) which also to be expected: the more a car has driven, the less its value and small motors are more efficient (lower mpg). Tax has a less but also significant positive correlation to the price (r = 0.44, p ~ 0), which makes sense semantically again. To further illustrate the correlations, a correlation matrix is plotted as well.

For the qualitative variables it is clear that the model explains the most variance in the price variable (R^2 = 0.488, p = 0) (only the price factor scores higher which is to be expected as it is build on the numeric value). This is to be expected as a specific model subsumes a whole series of other variables. The influence of the other qualitative variables are in order (highest R^2): (price, model), age, mileage, mpg, tax, transmission, engineSize, manufacturer and fuelType. Manufacturer and fuel type are poorly associated as they have R^2-values under 10%.

```
require(FactoMineR)

## Loading required package: FactoMineR

## Warning: package 'FactoMineR' was built under R version 4.0.5

require(corrplot)

res.con = condes(df,3)
res.con$quanti

##                 correlation       p.value
## year             0.61625532  0.000000e+00
## engineSize       0.57715694  0.000000e+00
## tax              0.43982997 1.929722e-226
## univ_outl_count  0.05998582  3.183838e-05
## mileage         -0.55367677  0.000000e+00
## mpg             -0.60490687  0.000000e+00
## n.age           -0.61625532  0.000000e+00

res.con$quali

##                       R2       p.value
## model        0.487739508  0.000000e+00
## f.age        0.381784788  0.000000e+00
## f.price      0.813844246  0.000000e+00
## f.mileage    0.337295393  0.000000e+00
## f.tax        0.276519319  0.000000e+00
## f.mpg        0.355074061  0.000000e+00
## transmission 0.248188295 4.589851e-298
## f.engineSize 0.225745103 6.262761e-266
## manufacturer 0.094372257 8.545144e-103
## fuelType     0.008387365  1.663122e-09

df_num = df[,c(3,5,7,8,9,11)]
cor_num = cor(df_num)
corrplot(cor_num, method = 'number')
```

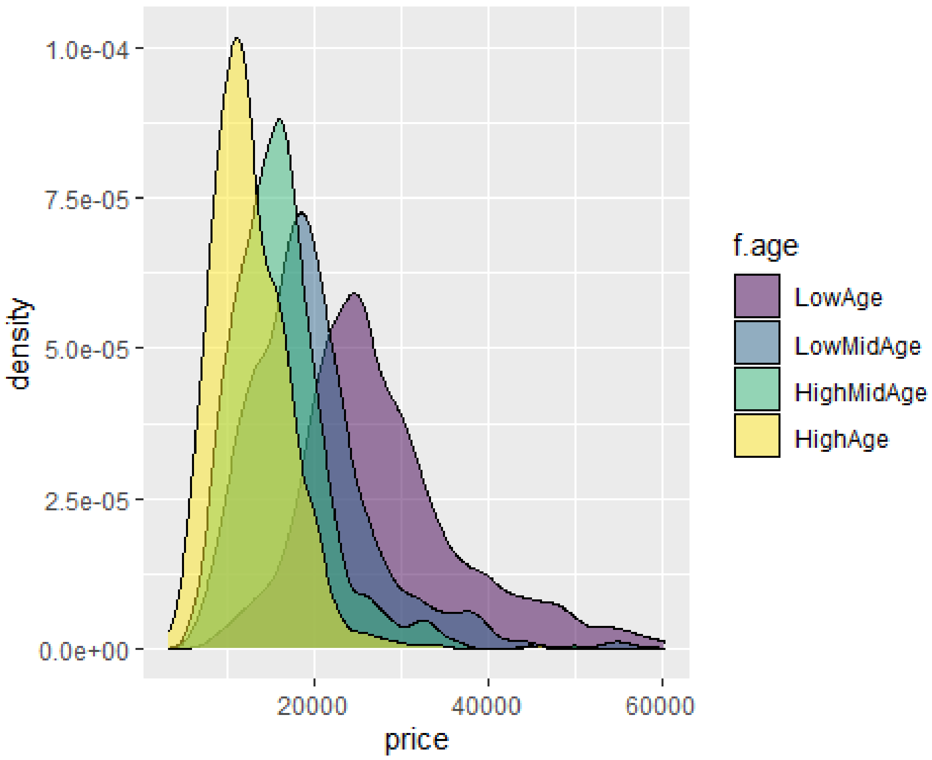


34

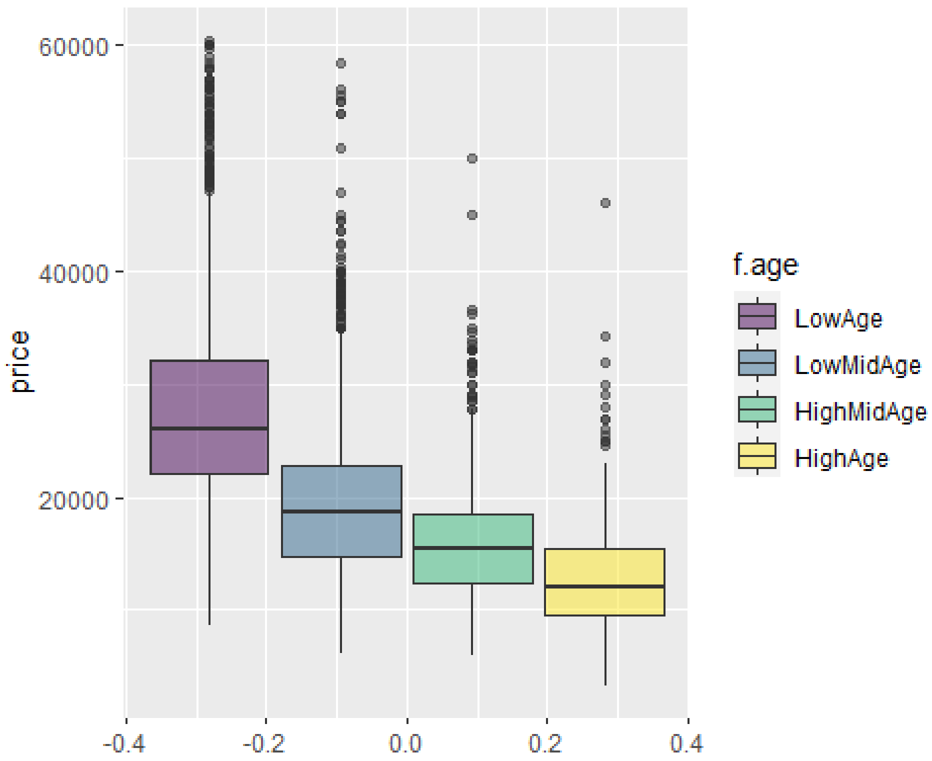
Define a polytomic factor f.age for the covariate car age according to its quartiles and argue if the average price depends on the level of age. Statistically justify the answer.

Given the evidence below, we argue that the price is dependent on the level of age. Firstly, in the boxplots it can be observed that the older the car (Q1) the lower the average price of the car. This is also seen clearly in the distribution plot below. Secondly, the wilcoxon test shows that the means of these levels are not equal meaning that some relation exists between the factor and the response variable. Lastly, using Kolmogorov-Smirnov we see that the price distributions of these levels are also not the same, further strenghtening our argument.

```
ggplot(df, aes(x=price, fill=f.age)) +
  geom_density(alpha=.5)
```




```
ggplot(df, aes(y=price, fill=f.age)) +
  geom_boxplot(alpha=0.5)
```



We perform pairwise wilcoxon test to check for similar means. Looking at the boxplot and distribution plot we can already expect that these are not going to be similar. The result of the wilcoxon test indicates our hypothesis was right that there is a clear difference between the means of the different quartiles.


```
pairwise.wilcox.test(df$price, df$f.age)

##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  df$price and df$f.age
##
##            LowAge LowMidAge HighMidAge
## LowMidAge  <2e-16 -         -
## HighMidAge <2e-16 <2e-16    -
## HighAge    <2e-16 <2e-16    <2e-16
##
## P value adjustment method: holm
```

We perform kolmogorov-Smirnov test whether the distributions of 2 quartiles are the same. From the plot above we can already assume that is likely these test are all going to reject the Null hypothesis, but we will check anyway.

AgeQ1 vs AgeQ2: distributions are not similar.

```
ks.test(df$price[df$f.age=="LowAge"], df$price[df$f.age=="LowMidAge"])

## Warning in ks.test(df$price[df$f.age == "LowAge"], df$price[df$f.age == : p-
## value will be approximate in the presence of ties

##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  df$price[df$f.age == "LowAge"] and df$price[df$f.age == "LowMidAge"]
```

```
## D = 0.47792, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

AgeQ1 vs AgeQ3: distributions are not similar.

```
ks.test(df$price[df$f.age=="LowAge"], df$price[df$f.age=="HighMidAge"])

## Warning in ks.test(df$price[df$f.age == "LowAge"], df$price[df$f.age == : p-
## value will be approximate in the presence of ties

##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  df$price[df$f.age == "LowAge"] and df$price[df$f.age == "HighMidAge"]
## D = 0.68707, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

AgeQ1 vs AgeQ4: distributions are not similar.

```
ks.test(df$price[df$f.age=="LowAge"], df$price[df$f.age=="HighAge"])

## Warning in ks.test(df$price[df$f.age == "LowAge"], df$price[df$f.age == : p-
## value will be approximate in the presence of ties

##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  df$price[df$f.age == "LowAge"] and df$price[df$f.age == "HighAge"]
## D = 0.81035, p-value < 2.2e-16
## alternative hypothesis: two-sided
```


AgeQ3 vs AgeQ2: distributions are not similar.

```
ks.test(df$price[df$f.age=="HighMidAge"], df$price[df$f.age=="LowMidAge"])

## Warning in ks.test(df$price[df$f.age == "HighMidAge"], df$price[df$f.age == : p-
## value will be approximate in the presence of ties

##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  df$price[df$f.age == "HighMidAge"] and df$price[df$f.age == "LowMidAge"]
## D = 0.27428, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

AgeQ4 vs AgeQ2: distributions are not similar.

```
ks.test(df$price[df$f.age=="HighAge"], df$price[df$f.age=="LowMidAge"])

## Warning in ks.test(df$price[df$f.age == "HighAge"], df$price[df$f.age == : p-
## value will be approximate in the presence of ties

##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  df$price[df$f.age == "HighAge"] and df$price[df$f.age == "LowMidAge"]
## D = 0.49092, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

AgeQ4 vs AgeQ3: distributions are not similar.

```
ks.test(df$price[df$f.age=="HighAge"], df$price[df$f.age=="HighMidAge"])

## Warning in ks.test(df$price[df$f.age == "HighAge"], df$price[df$f.age == : p-
## value will be approximate in the presence of ties
```

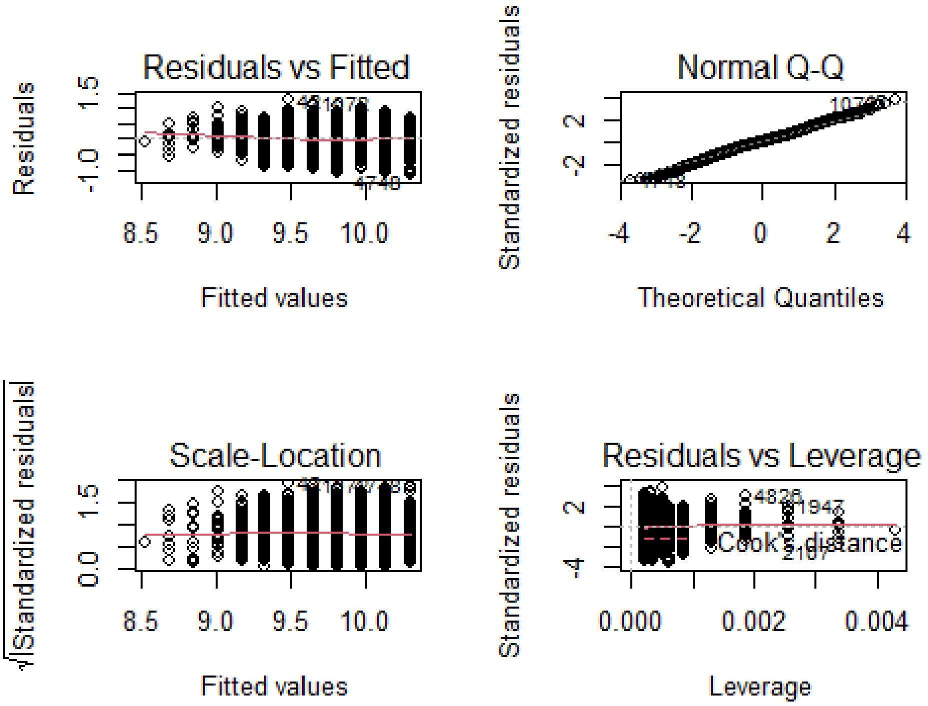
```
##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  df$price[df$f.age == "HighAge"] and df$price[df$f.age == "HighMidAge"]
## D = 0.31515, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

Price Modelling

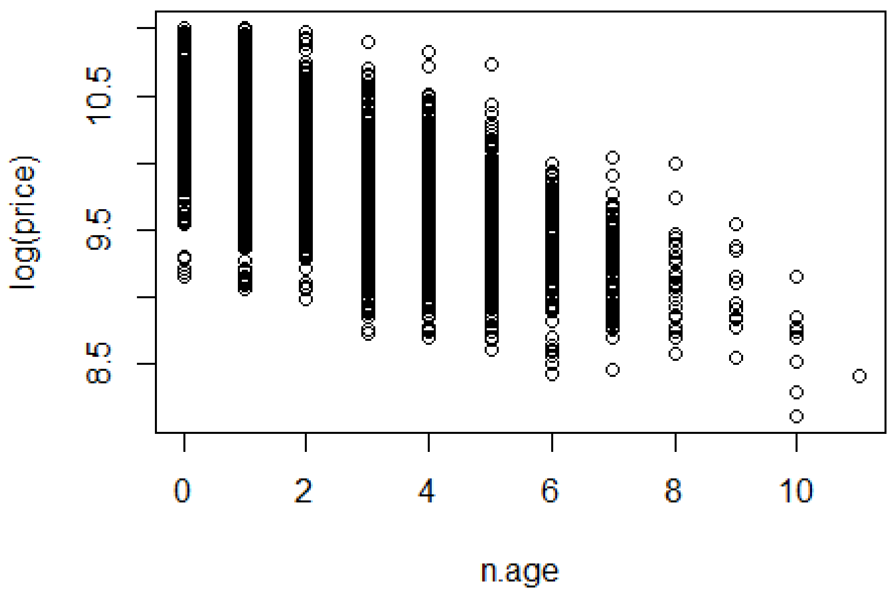
Calculate the linear regression model that explains the price from the age: interpret the regression line and assess its quality.

A linear model using the logarithmic price and n.age variable is constructed. It yields an R-sq of 46%, which is insufficient for accurate predictions. However, all diagnostic plots show solid results. The residuals vs fitted plot yields a more or less straight line which confirms linearity. The Q-Q plot indicates more or less normally distributed standard errors in the model. The scale-location plot gives a straight line which indicates that homoscedastisity is satisfied. The Residuals vs. leverage plot shows that there are some high leverage points in the model. However, they more or less lie in a straight line which indicates that there are no severe contradicting high-leverage instances in our simple model. An influence plot confirms this behaviour.

```
require(MASS)

## Loading required package: MASS

lmAgeLog = lm(log(price)~n.age, data = df)
par(mfrow=c(2,2))
summary(lmAgeLog)

##
## Call:
## lm(formula = log(price) ~ n.age, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.14028 -0.19935  0.00052  0.19819  1.24347
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 10.298277   0.008146 1264.19   <2e-16 ***
## n.age       -0.161069   0.002502  -64.36   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3269 on 4801 degrees of freedom
## Multiple R-squared:  0.4632, Adjusted R-squared:  0.4631
## F-statistic:  4143 on 1 and 4801 DF,  p-value: < 2.2e-16

plot(lmAgeLog)
```



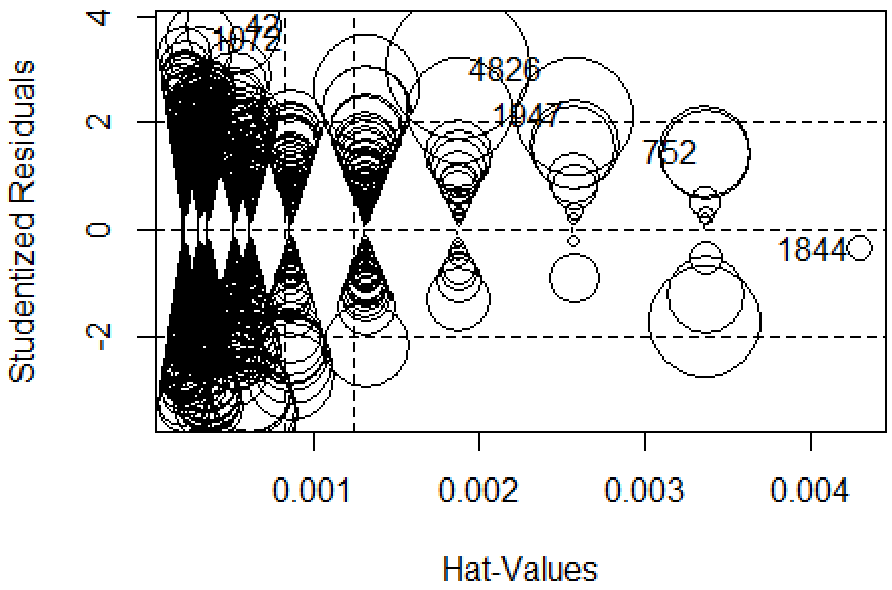

```
par(mfrow=c(1,1))
plot(log(price) ~ n.age, data = df)
```


```
influencePlot(lmAgeLog)
```

```
##         StudRes          Hat        CookD
## 42    3.8101204 0.0005307294 0.0038435282
## 752   1.4431868 0.0033704976 0.0035210927
## 1072  3.5704214 0.0003143748 0.0019995480
```

```
## 1844 -0.3515391 0.0042900503 0.0002662724
## 1947  2.1392082 0.0025681446 0.0058869312
## 4826  3.0303112 0.0018829912 0.0086471251

abline(lmAgeLog, col="red")
```



What is the percentage of the price variability that is explained by the age of the car?

As explained in question 6: for the log(price) ~ n.age model about 46% of the price variability is explained. This is insufficient for accurate prediction, such that additional explanatory variables are expected to be useful.

##. Do you think it is necessary to introduce a quadratic term in the equation that relates the price to its age?

To see if a polynomial transformation on n.age might be beneficial, the boxTidwell function is applied on the linear model (in which a constant term that doesnt influence results is added as the function can't handle zero values on the age). This function returns a lambda value of 0.87. This value seems to indicate that introducing a polynomial term to the age is unnecessary. However, to further confirm this believe, a transformation in the order of the square root is applied and evaluated.

The square root term does not significantly improve the R-sq value (0.005 improvement). The anova test indicates that the models are different, however with a borderline p-value of 0.023. The diagnostic plots show that introducing the square root further increases the leverage of already high-lev points. This impacts the model negatively as it is more prone for overfitting.

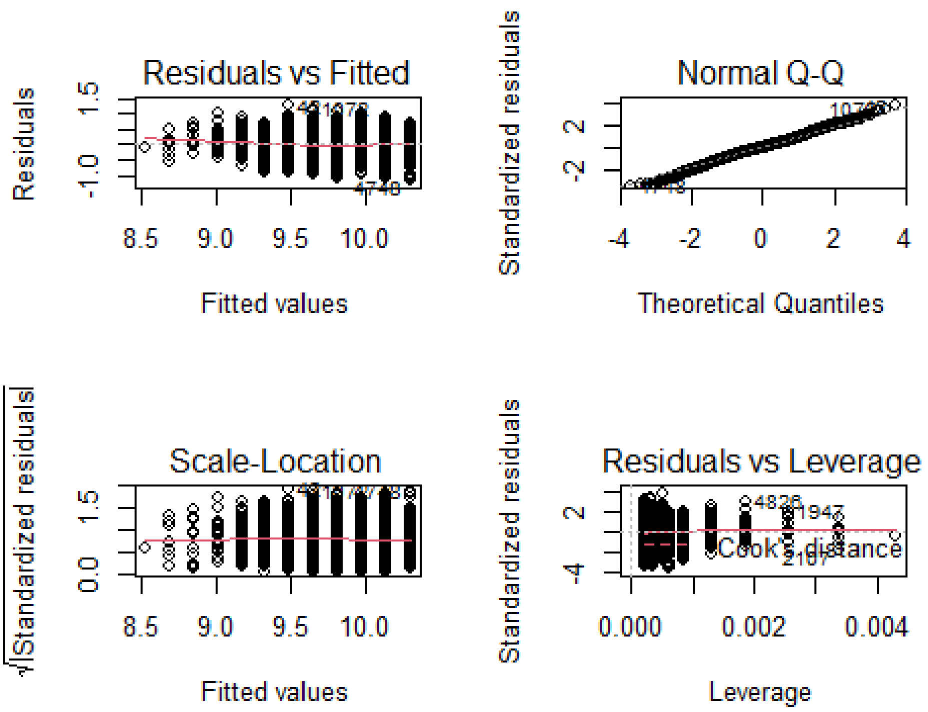
Lastly, looking at AIC, the addition of this polynomial term improves the criterion marginally. However, this insignificant improvement does not justify the addition of an extra degree of freedom that again allows extra room for overfitting.

Concluding, as the benefit of adding a polynomial term to the model, in terms of explainability, is very small, and as the quadratic models might be more biased towards values with high leverage, it is not necessary to add a quadratic term.

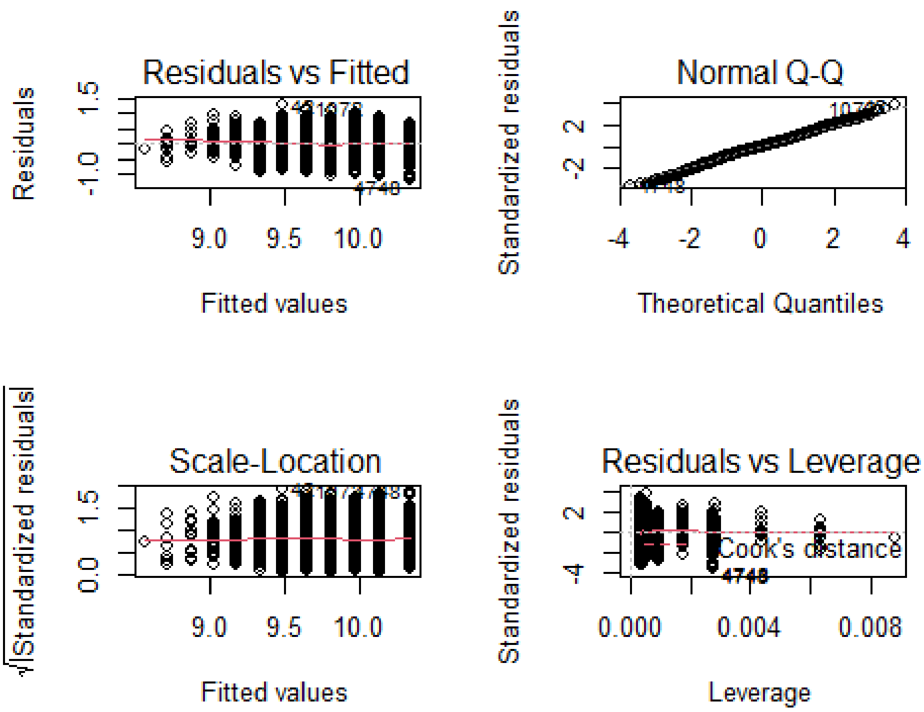
```
boxTidwell(log(price)~I(n.age+0.001), data = df)

##  MLE of lambda Score Statistic (z)  Pr(>|z|)
##        0.87201              4.7739 1.807e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## iterations =  4

lmAgeLogSqrt = lm(log(price)~(sqrt(n.age)+n.age), data=df)

summary(lmAgeLog)

##
## Call:
## lm(formula = log(price) ~ n.age, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.14028 -0.19935  0.00052  0.19819  1.24347
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 10.298277   0.008146 1264.19   <2e-16 ***
## n.age       -0.161069   0.002502  -64.36   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3269 on 4801 degrees of freedom
## Multiple R-squared:  0.4632, Adjusted R-squared:  0.4631
## F-statistic:  4143 on 1 and 4801 DF,  p-value: < 2.2e-16

summary(lmAgeLogSqrt)

##
## Call:
## lm(formula = log(price) ~ (sqrt(n.age) + n.age), data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.17479 -0.19829 -0.00186  0.20164  1.24201
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 10.332780   0.017220 600.051   <2e-16 ***
## sqrt(n.age) -0.054983   0.024179  -2.274    0.023 *
## n.age       -0.143090   0.008293 -17.255   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3268 on 4800 degrees of freedom
## Multiple R-squared:  0.4638, Adjusted R-squared:  0.4636
## F-statistic:  2076 on 2 and 4800 DF,  p-value: < 2.2e-16

anova(lmAgeLog, lmAgeLogSqrt)
```

41

```
## Analysis of Variance Table
##
## Model 1: log(price) ~ n.age
## Model 2: log(price) ~ (sqrt(n.age) + n.age)
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1   4801 513.07
## 2   4800 512.51  1   0.55212 5.1709 0.02301 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

par(mfrow=c(2,2))

plot(lmAgeLog)
```




```
plot(lmAgeLogSqrt)
```




```
AIC(lmAgeLog, lmAgeLogSqrt)

##              df      AIC
## lmAgeLog      3 2893.977
## lmAgeLogSqrt  4 2890.805

rm(lmAgeLogSqrt)
```

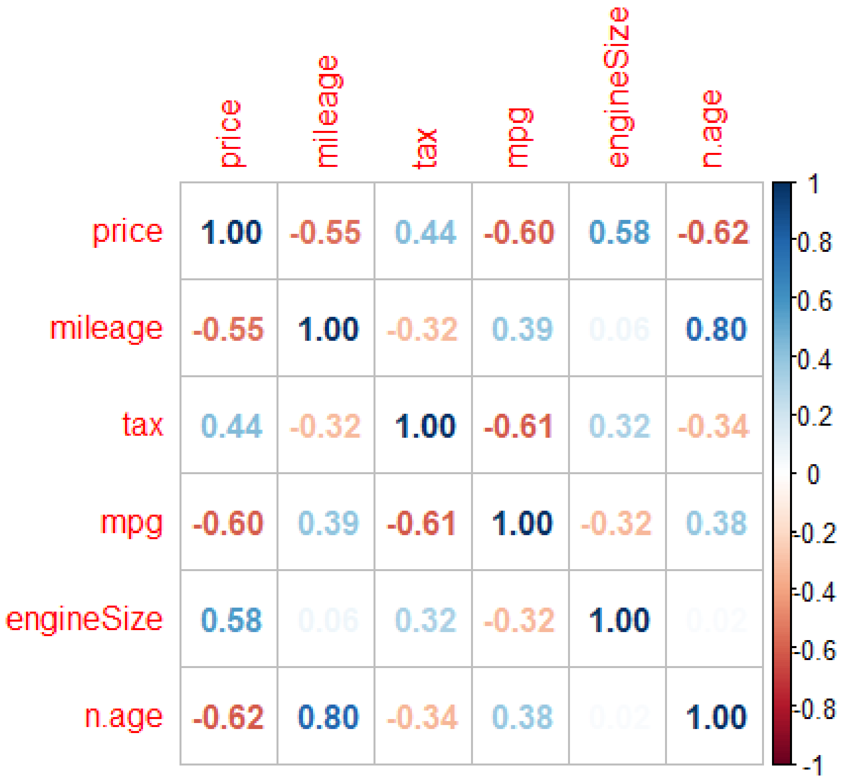

Are there any additional explanatory numeric variables needed to the car price? Study collinearity effects.

First, all remaining numeric variables are naively additively added to the model. This yields a signficantly better prediction model with an R-sq of about 83%. However, to simplify our model, collinearity is investigated to see if there are variables that are redundant in our model.

First a correlation matrix is plotted. This hints that possible candidates for linearity are: mileage & age (rho = 0.80) and tax & mpg (rho = -0.61). Next, the variance inflation factor is calculated for the numeric variables. This indicates whether or not a variable correlates too much with other predictors such that it becomes redundant in the model. In general, a VIF-value larger than 1/(1-R_sq) is considered as showing too much collinear behaviour. The result for every variable is always significantly below this threshold such that no severe collinearity is detected in the model. To further confirm this hypothesis, models are build by alternately removing the highly correlated variables from the logarithmic model. Then, ANOVA is applied to test whether or not the models are significantly predicting something else and AIC to see what model is considered the best. These tests show that the model with all numeric variables performs the best and that no severe collinearity is present in our model.

Therefore, the model of choice for the continuation of the project will be the one with all numeric variables.

```
lmNumLog = lm(log(price) ~ n.age+mileage+tax+mpg+engineSize, data=df)
t = summary(lmNumLog)

df_num = df[,c(3,5,7,8,9,11)]
cor_num = cor(df_num)
par(mfrow=c(1,1))
corrplot(cor_num, method = 'number')
```




```
vif(lmNumLog)

##      n.age    mileage        tax        mpg engineSize
##   2.787247   2.853458   1.699123   1.820579   1.217203

lmNumLog2 = lm(log(price) ~ n.age+tax+mpg+engineSize, data=df)
lmNumLog3 = lm(log(price) ~ n.age+mileage+mpg+engineSize, data=df)
lmNumLog4 = lm(log(price) ~ n.age+tax+mileage+engineSize, data=df)

anova(lmNumLog, lmNumLog2)

## Analysis of Variance Table
##
## Model 1: log(price) ~ n.age + mileage + tax + mpg + engineSize
## Model 2: log(price) ~ n.age + tax + mpg + engineSize
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1   4797 164.80
## 2   4798 180.51 -1   -15.704 457.12 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

anova(lmNumLog, lmNumLog3)

## Analysis of Variance Table
##
## Model 1: log(price) ~ n.age + mileage + tax + mpg + engineSize
```

44

```
## Model 2: log(price) ~ n.age + mileage + mpg + engineSize
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)    
## 1   4797 164.8                                  
## 2   4798 165.2 -1  -0.39664 11.545 0.0006848 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

anova(lmNumLog, lmNumLog4)

## Analysis of Variance Table
## 
## Model 1: log(price) ~ n.age + mileage + tax + mpg + engineSize
## Model 2: log(price) ~ n.age + tax + mileage + engineSize
##   Res.Df    RSS Df Sum of Sq     F    Pr(>F)    
## 1   4797 164.80                                 
## 2   4798 176.94 -1   -12.141 353.4 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

AIC(lmNumLog, lmNumLog2, lmNumLog3, lmNumLog4)

##           df       AIC
## lmNumLog   7 -2552.599
## lmNumLog2  6 -2117.421
## lmNumLog3  6 -2543.053
## lmNumLog4  6 -2213.180
```

After controlling by numerical variables, indicate whether the additive effect of the available factors on the price are statistically significant.

All the remaining available factors (except the ones defined on the numeric variables like fe. f.age) are now added one by one to the current model. This yields an improvement of R-sq of about 6%. An anova test indicates significant difference in prediction. The vif function indicates no significant collinearity by introducing the factors. Finally, the AIC function shows that the model with the factors is significantly better than the one without. Therefore, it can be concluded that the effect of the available factors is statistically significant and positive in its prediction capabilities.

```
lmNumLog = lm(log(price) ~ n.age+tax+mileage+engineSize+mpg, data=df)
lmFactLog = lm(log(price) ~ n.age+tax+mileage+engineSize+mpg+transmission+fuelType+manufac
turer, data=df)

summary(lmNumLog)

## 
## Call:
## lm(formula = log(price) ~ n.age + tax + mileage + engineSize + 
##     mpg, data = df)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -0.6920 -0.1168  0.0071  0.1260  0.8611 
## 
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  9.742e+00  2.455e-02 396.871  < 2e-16 ***
## n.age       -1.069e-01  2.369e-03 -45.132  < 2e-16 ***
## tax          2.040e-04  6.004e-05   3.398 0.000685 ***
## mileage     -4.888e-06  2.286e-07 -21.380  < 2e-16 ***
## engineSize   4.281e-01  5.488e-03  78.005  < 2e-16 ***
## mpg         -6.027e-03  3.206e-04 -18.799  < 2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1854 on 4797 degrees of freedom
## Multiple R-squared:  0.8276, Adjusted R-squared:  0.8274
## F-statistic:  4605 on 5 and 4797 DF,  p-value: < 2.2e-16

summary(lmFactLog)

##
## Call:
## lm(formula = log(price) ~ n.age + tax + mileage + engineSize +
##     mpg + transmission + fuelType + manufacturer, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.59016 -0.09418  0.00520  0.10045  0.77456
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             1.044e+01  3.099e-02 336.865  < 2e-16 ***
## n.age                  -9.842e-02  1.987e-03 -49.521  < 2e-16 ***
## tax                    -9.454e-05  5.038e-05  -1.877   0.0606 .
## mileage                -4.681e-06  1.898e-07 -24.659  < 2e-16 ***
## engineSize              2.791e-01  6.562e-03  42.533  < 2e-16 ***
## mpg                    -1.091e-02  3.502e-04 -31.144  < 2e-16 ***
## transmissionManual     -9.303e-02  6.718e-03 -13.848  < 2e-16 ***
## transmissionSemi-Auto   1.059e-02  5.689e-03   1.862   0.0627 .
## fuelTypeOther           1.507e-01  2.828e-02   5.329 1.03e-07 ***
## fuelTypePetrol         -1.117e-01  6.647e-03 -16.806  < 2e-16 ***
## manufacturerBMW        -8.470e-02  7.048e-03 -12.018  < 2e-16 ***
## manufacturerMercedes    1.750e-02  6.840e-03   2.558   0.0105 *
## manufacturerVW         -1.784e-01  6.336e-03 -28.153  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1528 on 4790 degrees of freedom
## Multiple R-squared:  0.8831, Adjusted R-squared:  0.8828
## F-statistic:  3014 on 12 and 4790 DF,  p-value: < 2.2e-16

anova(lmFactLog, lmNumLog)

## Analysis of Variance Table
##
## Model 1: log(price) ~ n.age + tax + mileage + engineSize + mpg + transmission +
##     fuelType + manufacturer
## Model 2: log(price) ~ n.age + tax + mileage + engineSize + mpg
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1   4790 111.77
## 2   4797 164.80 -7   -53.029 324.65 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

vif(lmFactLog)

##                  GVIF Df GVIF^(1/(2*Df))
## n.age        2.888413  1        1.699533
## tax          1.761263  1        1.327126
## mileage      2.896111  1        1.701797
## engineSize   2.561845  1        1.600576
## mpg          3.199012  1        1.788578
## transmission 1.606988  2        1.125909
## fuelType     2.254649  2        1.225377
## manufacturer 1.513418  3        1.071502

AIC(lmFactLog, lmNumLog)
```

```
##           df       AIC
## lmFactLog 14 -4403.483
## lmNumLog   7 -2552.599

summary(lmFactLog)

##
## Call:
## lm(formula = log(price) ~ n.age + tax + mileage + engineSize +
##     mpg + transmission + fuelType + manufacturer, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.59016 -0.09418  0.00520  0.10045  0.77456
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)            1.044e+01  3.099e-02 336.865  < 2e-16 ***
## n.age                 -9.842e-02  1.987e-03 -49.521  < 2e-16 ***
## tax                   -9.454e-05  5.038e-05  -1.877   0.0606 .
## mileage               -4.681e-06  1.898e-07 -24.659  < 2e-16 ***
## engineSize             2.791e-01  6.562e-03  42.533  < 2e-16 ***
## mpg                   -1.091e-02  3.502e-04 -31.144  < 2e-16 ***
## transmissionManual    -9.303e-02  6.718e-03 -13.848  < 2e-16 ***
## transmissionSemi-Auto  1.059e-02  5.689e-03   1.862   0.0627 .
## fuelTypeOther          1.507e-01  2.828e-02   5.329 1.03e-07 ***
## fuelTypePetrol        -1.117e-01  6.647e-03 -16.806  < 2e-16 ***
## manufacturerBMW       -8.470e-02  7.048e-03 -12.018  < 2e-16 ***
## manufacturerMercedes   1.750e-02  6.840e-03   2.558   0.0105 *
## manufacturerVW        -1.784e-01  6.336e-03 -28.153  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1528 on 4790 degrees of freedom
## Multiple R-squared:  0.8831, Adjusted R-squared:  0.8828
## F-statistic:  3014 on 12 and 4790 DF,  p-value: < 2.2e-16
```


Select the best model available so far. Interpret the equations that relate the explanatory variables to the answer (rate).

From the AIC we find that the model with all factors, all numeric variables and a logarithmic transformation has the best criterion.

In order to interpret the rates of the model for each of the variables, we use the contrasts option "sum" to be able to derive the coefficients for the factors that are subsumed by the intercept.

Because we are using contr.sum, the general equation will be Yhat = mu + alpha + beta + gamma + error, where alpha describes the transmission factor, beta describes the fuelType factor, and gamme describes the manufacturer factor. For alpha, beta, gamma we have: sum(alpha) = 0, sum(beta) = 0, and sum(gamme) = 0. Most of the rates of the factors are given by the summary fucntion. However, cases transmission = Automatic, fuelType = Diesel and manufacturer = Audi are contained in the intercept of the equation. These rates are calculated using the fact that the coefficients sum to zero.

We thus find coefficients: transmissionAutomatic = - (2.755e-02 - 6.539e-02) = 3.784e-02 fuelTypeDiesel = - (1.374e-01 - 0) = - 1.374e-01 (0 because the p-value is above 0.05) manufacutererAudi = - (6.122e-02 - 2.367e-02 + 7.930e-02) = - 1.1685e-01

The rest of the rates can be found in the summary.

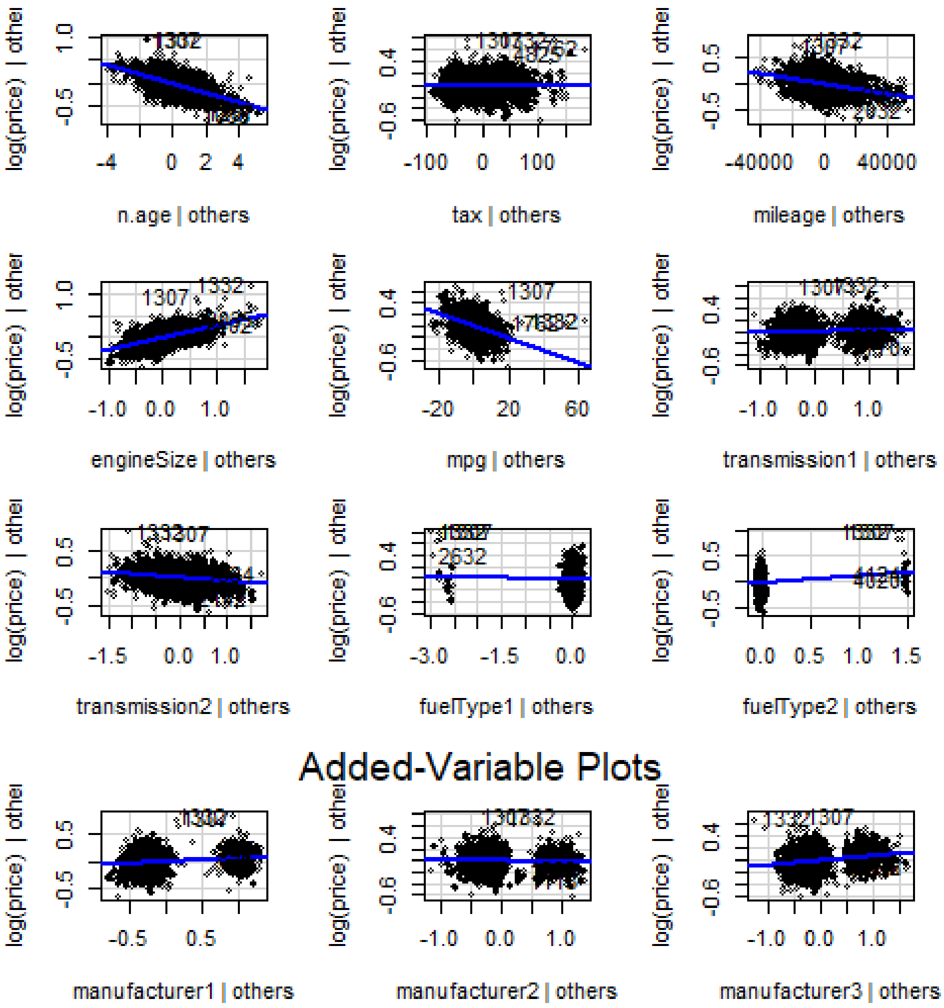
Now, some graphical representations of the behaviour of the variables in the model are investigated and looked in to. First, an added variable plot is obtained using the avPlots-function. This plot represents for each predictor in the model, the actual behaviour of the response variable by keeping the influence of the other explanatory variables constant. This scatterplot matrix clearly represents the linear relationships between the explanatory variables and response variable. However, to further look for monotone linear relationsships that might benefit from a transformation, the crPlots funtion is used, which is a Partial-Residual plot. These are partial regressions and are used to distinguish between monotone linearity (which might benefit from a transformation) and non-monotone linearity (which don't). From these plots however, no clear cases of monotone linearity are found. However, from the av-plot showing mpg vs the prices and petrol vs prices, we see that this variable is heavily influenced by a few values with high leverage. This will be discussed in question 14, after which the av-plot will be shown again.

```
AIC(lmAgeLog, lmFactLog, lmNumLog, lmNumLog2, lmNumLog3, lmNumLog4)

##             df       AIC
## lmAgeLog     3  2893.977
## lmFactLog   14 -4403.483
## lmNumLog     7 -2552.599
## lmNumLog2    6 -2117.421
## lmNumLog3    6 -2543.053
## lmNumLog4    6 -2213.180

options(contrasts = c("contr.sum", "contr.sum"))
lmBest= lm(log(price) ~ n.age+tax+mileage+engineSize+mpg+transmission+fuelType+manufacture
r, data=df)
summary(lmBest)

##
## Call:
## lm(formula = log(price) ~ n.age + tax + mileage + engineSize +
##     mpg + transmission + fuelType + manufacturer, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.59016 -0.09418  0.00520  0.10045  0.77456
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     1.036e+01  3.074e-02 337.046  < 2e-16 ***
## n.age          -9.842e-02  1.987e-03 -49.521  < 2e-16 ***
## tax            -9.454e-05  5.038e-05  -1.877   0.0606 .
## mileage        -4.681e-06  1.898e-07 -24.659  < 2e-16 ***
## engineSize      2.791e-01  6.562e-03  42.533  < 2e-16 ***
## mpg            -1.091e-02  3.502e-04 -31.144  < 2e-16 ***
## transmission1   2.748e-02  3.602e-03   7.628 2.85e-14 ***
## transmission2  -6.555e-02  3.863e-03 -16.968  < 2e-16 ***
## fuelType1      -1.300e-02  9.844e-03  -1.320   0.1868
## fuelType2       1.377e-01  1.882e-02   7.318 2.94e-13 ***
```

```
## manufacturer1  6.140e-02  4.125e-03  14.885  < 2e-16 ***
## manufacturer2 -2.331e-02  4.282e-03  -5.443 5.51e-08 ***
## manufacturer3  7.890e-02  4.092e-03  19.281  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1528 on 4790 degrees of freedom
## Multiple R-squared:  0.8831, Adjusted R-squared:  0.8828
## F-statistic:  3014 on 12 and 4790 DF,  p-value: < 2.2e-16

boxTidwell(log(price)~ mileage, ~n.age+tax+engineSize+mpg+transmission+fuelType+manufactur
er, data=df, max.iter = 70)

##  MLE of lambda Score Statistic (z) Pr(>|z|)
##         1.1987             -2.6163  0.00889 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## iterations =  4

boxTidwell(log(price)~ engineSize, ~n.age+tax+mileage+mpg+transmission+fuelType+manufactur
er, data=df, max.iter = 70)

##  MLE of lambda Score Statistic (z)  Pr(>|z|)
##       0.077326             -11.209 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## iterations =  3

boxTidwell(log(price)~ mpg, ~n.age+tax+mileage+engineSize+transmission+fuelType+manufactur
er, data=df, max.iter = 70)

##  MLE of lambda Score Statistic (z)  Pr(>|z|)
##      -0.054497              9.4464 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## iterations =  6

boxTidwell(log(price)~ I(tax+0.01), ~n.age+tax+mileage+engineSize+transmission+fuelType+ma
nufacturer, data=df, max.iter = 70)

##  MLE of lambda Score Statistic (z) Pr(>|z|)
##        0.26626             -1.8716  0.06127 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## iterations =  12

lmBest2 = lm(log(price) ~ n.age+tax+engineSize+mpg+transmission+fuelType+manufacturer, dat
a=df)
lmBest3 = lm(log(price) ~ n.age+tax+log(engineSize)+mpg+transmission+fuelType+manufacturer
, data=df)
lmBest4 = lm(log(price) ~ n.age+tax+engineSize+log(mpg)+transmission+fuelType+manufacturer
, data=df)
summary(lmBest)

##
## Call:
## lm(formula = log(price) ~ n.age + tax + mileage + engineSize +
##     mpg + transmission + fuelType + manufacturer, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.59016 -0.09418  0.00520  0.10045  0.77456
```

```
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     1.036e+01  3.074e-02 337.046  < 2e-16 ***
## n.age          -9.842e-02  1.987e-03 -49.521  < 2e-16 ***
## tax            -9.454e-05  5.038e-05  -1.877   0.0606 .
## mileage        -4.681e-06  1.898e-07 -24.659  < 2e-16 ***
## engineSize      2.791e-01  6.562e-03  42.533  < 2e-16 ***
## mpg            -1.091e-02  3.502e-04 -31.144  < 2e-16 ***
## transmission1   2.748e-02  3.602e-03   7.628 2.85e-14 ***
## transmission2  -6.555e-02  3.863e-03 -16.968  < 2e-16 ***
## fuelType1      -1.300e-02  9.844e-03  -1.320   0.1868
## fuelType2       1.377e-01  1.882e-02   7.318 2.94e-13 ***
## manufacturer1   6.140e-02  4.125e-03  14.885  < 2e-16 ***
## manufacturer2  -2.331e-02  4.282e-03  -5.443 5.51e-08 ***
## manufacturer3   7.890e-02  4.092e-03  19.281  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1528 on 4790 degrees of freedom
## Multiple R-squared:  0.8831, Adjusted R-squared:  0.8828
## F-statistic:  3014 on 12 and 4790 DF,  p-value: < 2.2e-16

summary(lmBest2)

##
## Call:
## lm(formula = log(price) ~ n.age + tax + engineSize + mpg + transmission +
##     fuelType + manufacturer, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.68470 -0.10190  0.00573  0.10740  0.81999
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     1.042e+01  3.253e-02 320.377  < 2e-16 ***
## n.age          -1.343e-01  1.439e-03 -93.332  < 2e-16 ***
## tax            -6.574e-05  5.346e-05  -1.230   0.2189
## engineSize      2.668e-01  6.945e-03  38.417  < 2e-16 ***
## mpg            -1.174e-02  3.700e-04 -31.741  < 2e-16 ***
## transmission1   2.628e-02  3.824e-03   6.874 7.02e-12 ***
## transmission2  -6.744e-02  4.100e-03 -16.450  < 2e-16 ***
## fuelType1      -2.081e-02  1.044e-02  -1.993   0.0463 *
## fuelType2       1.410e-01  1.997e-02   7.058 1.93e-12 ***
## manufacturer1   5.727e-02  4.375e-03  13.092  < 2e-16 ***
## manufacturer2  -2.520e-02  4.544e-03  -5.545 3.09e-08 ***
## manufacturer3   8.663e-02  4.331e-03  20.003  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1621 on 4791 degrees of freedom
## Multiple R-squared:  0.8682, Adjusted R-squared:  0.8679
## F-statistic:  2869 on 11 and 4791 DF,  p-value: < 2.2e-16

summary(lmBest3)

##
## Call:
## lm(formula = log(price) ~ n.age + tax + log(engineSize) + mpg +
##     transmission + fuelType + manufacturer, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.66502 -0.10344  0.00315  0.10699  0.79504
##
```

```
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)    
## (Intercept)       1.057e+01  2.941e-02 359.318  < 2e-16 ***
## n.age            -1.351e-01  1.427e-03 -94.714  < 2e-16 ***
## tax              -1.784e-05  5.293e-05  -0.337 0.736119    
## log(engineSize)   5.507e-01  1.375e-02  40.052  < 2e-16 ***
## mpg              -1.123e-02  3.700e-04 -30.353  < 2e-16 ***
## transmission1     2.425e-02  3.789e-03   6.398 1.72e-10 ***
## transmission2    -6.084e-02  4.091e-03 -14.870  < 2e-16 ***
## fuelType1        -3.471e-02  1.040e-02  -3.338 0.000849 ***
## fuelType2         1.376e-01  1.977e-02   6.960 3.87e-12 ***
## manufacturer1     5.628e-02  4.328e-03  13.004  < 2e-16 ***
## manufacturer2    -2.960e-02  4.519e-03  -6.551 6.34e-11 ***
## manufacturer3     8.334e-02  4.291e-03  19.420  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.1605 on 4791 degrees of freedom
## Multiple R-squared:  0.8709, Adjusted R-squared:  0.8706 
## F-statistic:  2937 on 11 and 4791 DF,  p-value: < 2.2e-16

summary(lmBest4)

## 
## Call:
## lm(formula = log(price) ~ n.age + tax + engineSize + log(mpg) + 
##     transmission + fuelType + manufacturer, data = df)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -0.68796 -0.10045  0.00486  0.10519  0.64091 
## 
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)    
## (Intercept)     1.222e+01  8.417e-02 145.186  < 2e-16 ***
## n.age          -1.326e-01  1.443e-03 -91.910  < 2e-16 ***
## tax             2.554e-05  5.165e-05   0.494    0.621    
## engineSize      2.516e-01  7.106e-03  35.411  < 2e-16 ***
## log(mpg)       -6.125e-01  1.868e-02 -32.785  < 2e-16 ***
## transmission1   2.491e-02  3.805e-03   6.548 6.45e-11 ***
## transmission2  -6.786e-02  4.074e-03 -16.658  < 2e-16 ***
## fuelType1      -8.476e-03  1.044e-02  -0.812    0.417    
## fuelType2       1.240e-01  1.983e-02   6.254 4.36e-10 ***
## manufacturer1   5.633e-02  4.351e-03  12.947  < 2e-16 ***
## manufacturer2  -2.372e-02  4.523e-03  -5.244 1.64e-07 ***
## manufacturer3   8.568e-02  4.297e-03  19.938  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.1612 on 4791 degrees of freedom
## Multiple R-squared:  0.8697, Adjusted R-squared:  0.8694 
## F-statistic:  2908 on 11 and 4791 DF,  p-value: < 2.2e-16

AIC(lmBest, lmBest2, lmBest3, lmBest4)

##         df       AIC
## lmBest  14 -4403.483
## lmBest2 13 -3831.464
## lmBest3 13 -3928.789
## lmBest4 13 -3886.930

avPlots(lmBest)
```



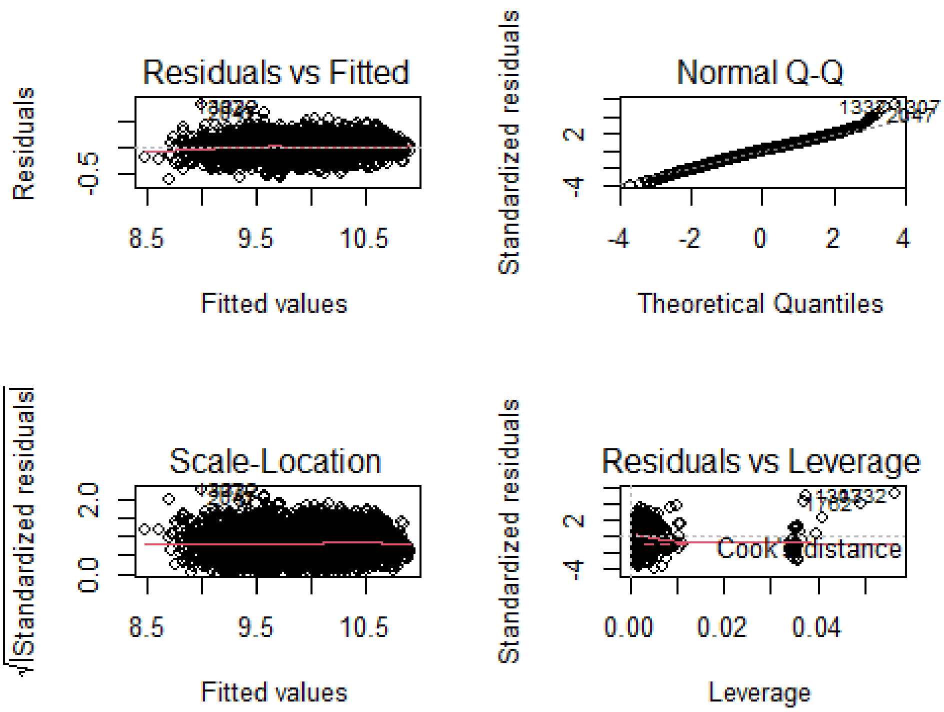
Graphically assess the best model obtained so far.

The residuals vs Fitted plot shows that the residuals follow a good linear pattern, which meets the regression assumptions very well. The Normal Q-Q plot shows that the standard errors are mostly normally distributed with a small amount of deviating prediction at the upper end of the tail. The scale-location plot shows that homoscedasticity is satisfied as a straight line is obtained. The residuals vs leverage plot shows that our model includes some high-leverage (so highly influential in our model) points that deviate significantly (more than 4 standardized residuals away) and asymmetrically from the prediction. An influence plot further confirms this believe.

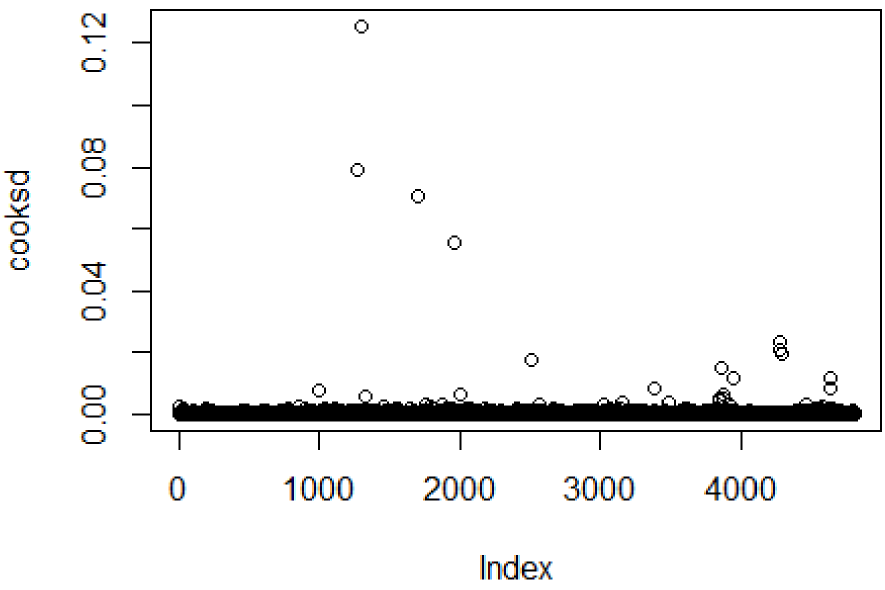
To quantify their influence, the model is reconstructed without these noteworthy points. The result is that the errors are now even more closely normally distributed around the predictions. Moreover, the new high-leverage points are more closely and symmetrically distributed around the predictions. This new model also yields a marginal R-squared improvement of about 0.3%. It can be concluded that removing the highly influential points results in more favorable diagnostic plots. This thus concludes the new best model.

As discussed in 11. the av-plot is shown again in order to demonstrate that the high levearge points in mpg have been removed and no longer influence its line.

```
options(contrasts = c("contr.treatment", "contr.treatment"))
summary(lmBest)

##
## Call:
## lm(formula = log(price) ~ n.age + tax + mileage + engineSize +
##     mpg + transmission + fuelType + manufacturer, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.59016 -0.09418  0.00520  0.10045  0.77456
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.036e+01  3.074e-02 337.046  < 2e-16 ***
## n.age         -9.842e-02  1.987e-03 -49.521  < 2e-16 ***
## tax           -9.454e-05  5.038e-05  -1.877   0.0606 .
## mileage       -4.681e-06  1.898e-07 -24.659  < 2e-16 ***
## engineSize     2.791e-01  6.562e-03  42.533  < 2e-16 ***
## mpg           -1.091e-02  3.502e-04 -31.144  < 2e-16 ***
## transmission1  2.748e-02  3.602e-03   7.628 2.85e-14 ***
## transmission2 -6.555e-02  3.863e-03 -16.968  < 2e-16 ***
## fuelType1     -1.300e-02  9.844e-03  -1.320   0.1868
## fuelType2      1.377e-01  1.882e-02   7.318 2.94e-13 ***
## manufacturer1  6.140e-02  4.125e-03  14.885  < 2e-16 ***
## manufacturer2 -2.331e-02  4.282e-03  -5.443 5.51e-08 ***
## manufacturer3  7.890e-02  4.092e-03  19.281  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1528 on 4790 degrees of freedom
## Multiple R-squared:  0.8831, Adjusted R-squared:  0.8828
## F-statistic:  3014 on 12 and 4790 DF,  p-value: < 2.2e-16

par(mfrow=c(2,2))
plot(lmBest)
```

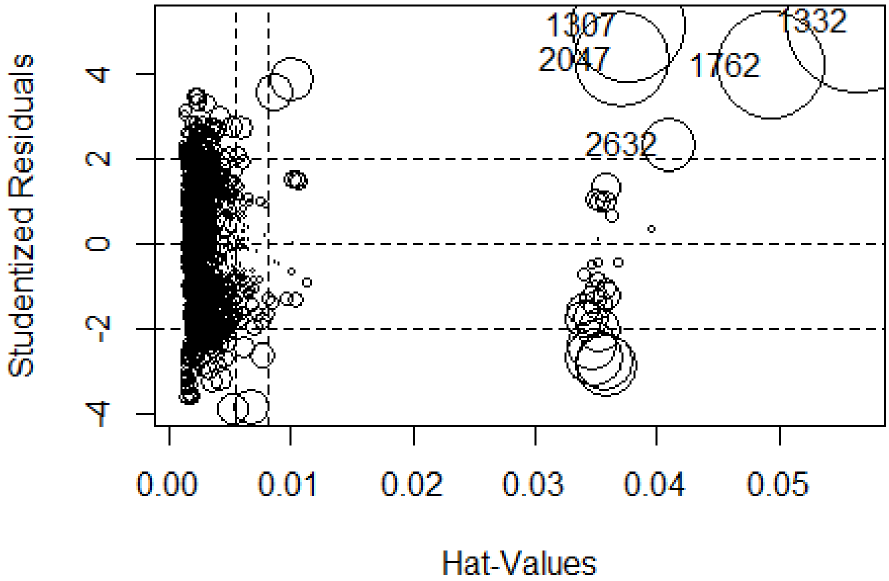



```
par(mfrow=c(1,1))
cooksd <- cooks.distance(lmBest)
plot(cooksd)
```

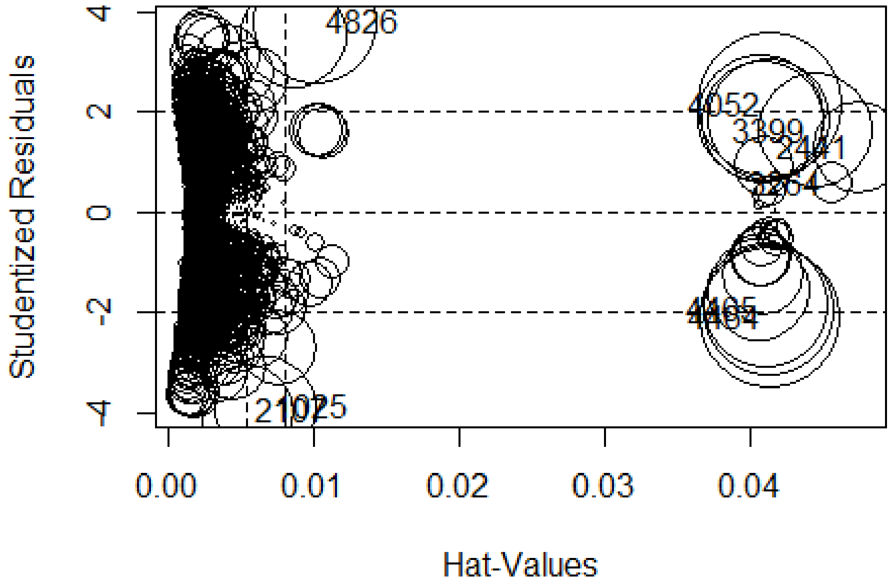




```
influencePlot(lmBest, id=list(n=3, method="noteworthy"))

##        StudRes        Hat      CookD
## 1307 5.145781 0.03766905 0.07930771
## 1332 5.234571 0.05652106 0.12557693
```

```
## 1762 4.201172 0.04944766 0.07038179
## 2047 4.326689 0.03716402 0.05537767
## 2632 2.331151 0.04088376 0.01780226

high_infl = rownames(as.data.frame(influencePlot(lmBest, id=list(n=3, method="noteworthy")
)))
```

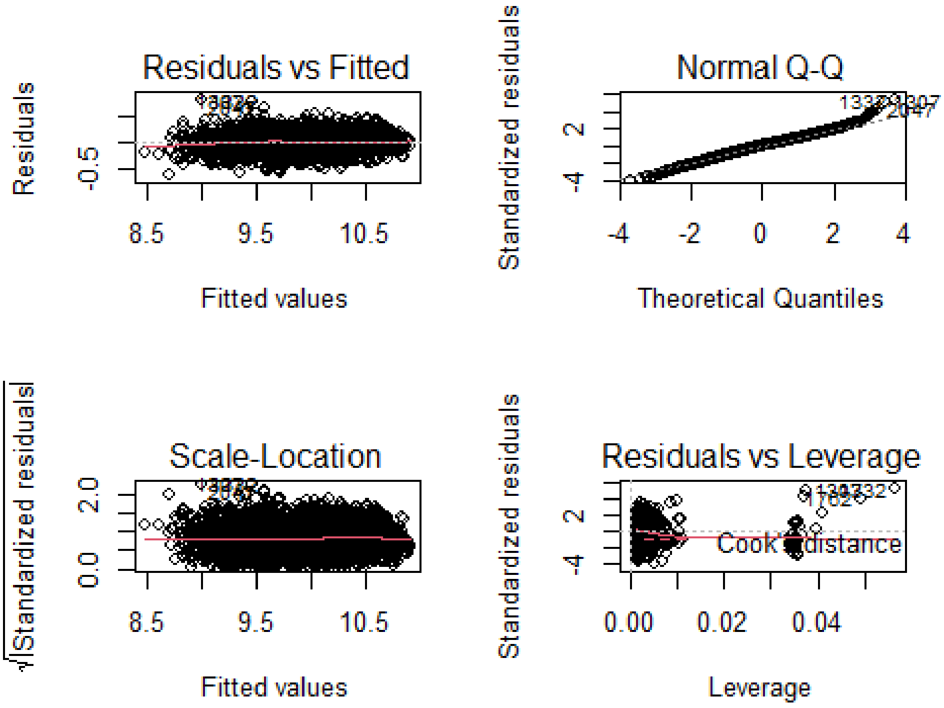



```
df_no_high_infl = df[!(rownames(df) %in% high_infl),]
lmBest_no_high_infl = lm(log(price) ~ n.age+tax+mileage+engineSize+mpg+transmission+fuelTy
pe+manufacturer, data=df_no_high_infl)
influencePlot(lmBest_no_high_infl, id=list(n=3, method="noteworthy"))
```

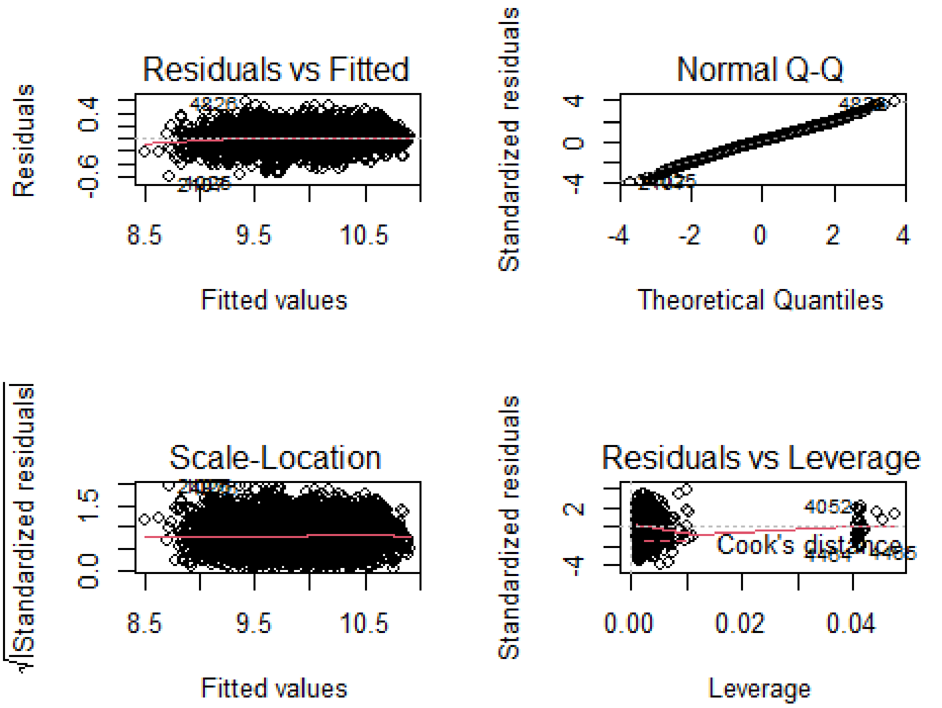



```
##          StudRes         Hat       CookD
## 1025 -3.8884860 0.006737703 0.007866580
## 2107 -3.9611022 0.005390858 0.006521732
## 2441  1.2922595 0.047552932 0.006412565
## 3264  0.5869794 0.045578061 0.001265836
## 3399  1.6377138 0.044466329 0.009597658
## 4052  2.1582652 0.041394204 0.015460867
## 4464 -2.0897835 0.041401007 0.014498653
## 4465 -1.9546346 0.041329574 0.012662617
## 4826  3.8021627 0.010185029 0.011410552

summary(lmBest)

##
## Call:
## lm(formula = log(price) ~ n.age + tax + mileage + engineSize +
##     mpg + transmission + fuelType + manufacturer, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.59016 -0.09418  0.00520  0.10045  0.77456
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.036e+01  3.074e-02 337.046  < 2e-16 ***
## n.age         -9.842e-02  1.987e-03 -49.521  < 2e-16 ***
## tax           -9.454e-05  5.038e-05  -1.877   0.0606 .
## mileage       -4.681e-06  1.898e-07 -24.659  < 2e-16 ***
## engineSize     2.791e-01  6.562e-03  42.533  < 2e-16 ***
## mpg           -1.091e-02  3.502e-04 -31.144  < 2e-16 ***
## transmission1  2.748e-02  3.602e-03   7.628 2.85e-14 ***
## transmission2 -6.555e-02  3.863e-03 -16.968  < 2e-16 ***
## fuelType1     -1.300e-02  9.844e-03  -1.320   0.1868
## fuelType2      1.377e-01  1.882e-02   7.318 2.94e-13 ***
## manufacturer1  6.140e-02  4.125e-03  14.885  < 2e-16 ***
## manufacturer2 -2.331e-02  4.282e-03  -5.443 5.51e-08 ***
## manufacturer3  7.890e-02  4.092e-03  19.281  < 2e-16 ***
## ---
```

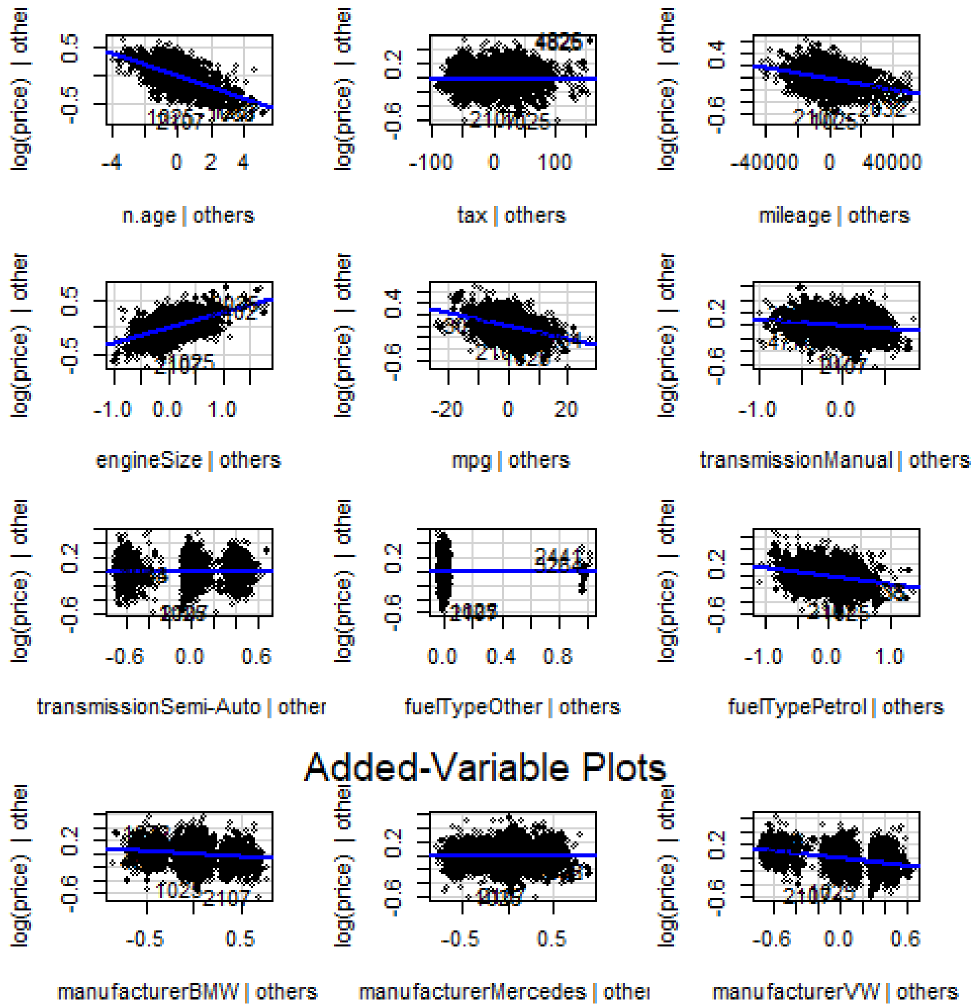
```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1528 on 4790 degrees of freedom
## Multiple R-squared:  0.8831, Adjusted R-squared:  0.8828
## F-statistic:  3014 on 12 and 4790 DF,  p-value: < 2.2e-16

summary(lmBest_no_high_infl)

##
## Call:
## lm(formula = log(price) ~ n.age + tax + mileage + engineSize +
##     mpg + transmission + fuelType + manufacturer, data = df_no_high_infl)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.59566 -0.09277  0.00537  0.10074  0.57046
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)            1.050e+01  3.123e-02 336.125  < 2e-16 ***
## n.age                 -9.778e-02  1.966e-03 -49.740  < 2e-16 ***
## tax                   -1.229e-04  4.998e-05  -2.458  0.01399 *
## mileage               -4.659e-06  1.878e-07 -24.807  < 2e-16 ***
## engineSize             2.710e-01  6.541e-03  41.440  < 2e-16 ***
## mpg                   -1.167e-02  3.551e-04 -32.869  < 2e-16 ***
## transmissionManual    -9.015e-02  6.651e-03 -13.554  < 2e-16 ***
## transmissionSemi-Auto  1.222e-02  5.629e-03   2.172  0.02994 *
## fuelTypeOther          1.740e-02  3.071e-02   0.567  0.57102
## fuelTypePetrol        -1.218e-01  6.647e-03 -18.325  < 2e-16 ***
## manufacturerBMW       -8.382e-02  6.971e-03 -12.023  < 2e-16 ***
## manufacturerMercedes   2.069e-02  6.771e-03   3.055  0.00226 **
## manufacturerVW        -1.768e-01  6.266e-03 -28.218  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.151 on 4785 degrees of freedom
## Multiple R-squared:  0.8857, Adjusted R-squared:  0.8854
## F-statistic:  3091 on 12 and 4785 DF,  p-value: < 2.2e-16

par(mfrow=c(2,2))
plot(lmBest)
```




```
plot(lmBest_no_high_infl)
```


```
lmBest = lm(log(price) ~ n.age+tax+mileage+engineSize+mpg+transmission+fuelType+manufactur
er, data=df_no_high_infl)

avPlots(lmBest)
```



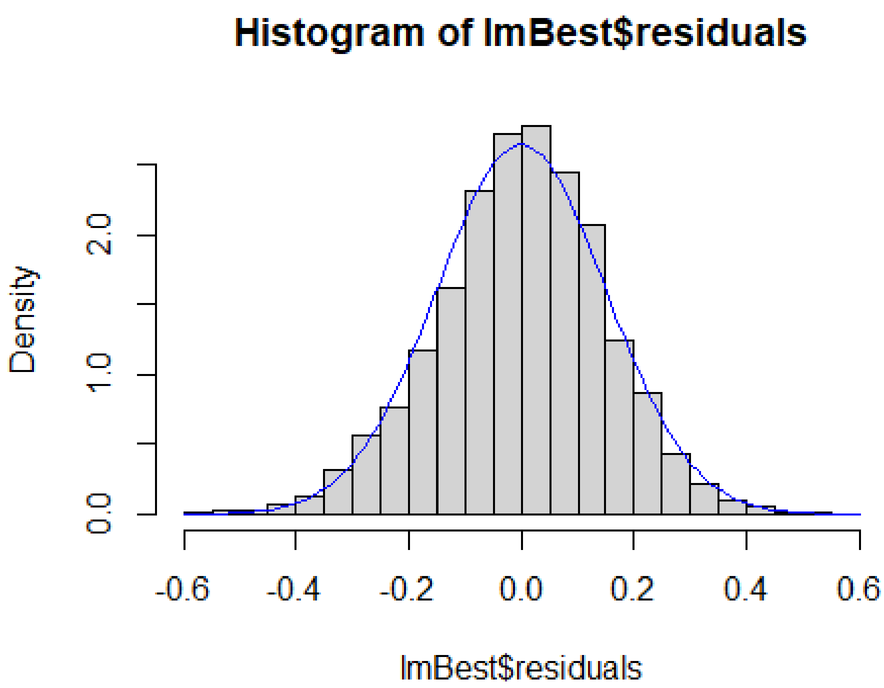
Assess the presence of outliers in the studentized residuals at a 99% confidence level. Indicate what those observations are.

As values with high leverage were already removed in the last question, the outliers found here might not coincide with the outliers of the original model.

First, a histogram is plotted to make sure the residuals follow a nice and smooth normal distribution, which they do. Then the studentized residuals at the 99% CI are calculated. The boxplot and plot of the residuals show which values are considered outliers. Interestingly, the plot of residuals shows many residuals grouped tightly together in the right bottom of the plot. Out of curiosity, these will be inspected more in depth a little later. First, the cooks distance is plotted, together with the outliers crossed out. As can be seen, most observations with a high leverage also turn up as residual outliers in the model.

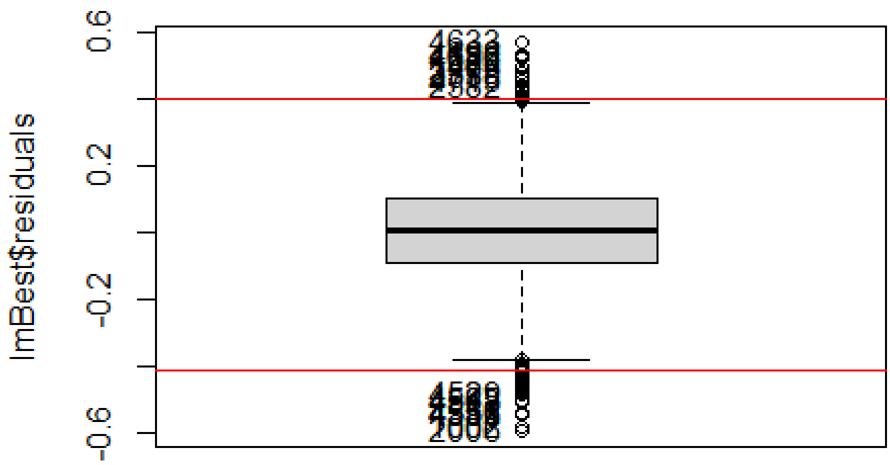
From the summary, it can be observed that half of the outliers are Volkswagens. Using a boxplot the prices of the residual outliers vs the manufacturer are plotted to see if Volkswagen deviates from the others, which apart from the 4 outliers from Mercedes, is not the case. Lastly, all prices are plotted, together with the prices of the outliers per manufacturer (Volkswagen=blue, Audi=red, Mercedes=green, and BMW=orange). From this it seems that the earlier observed clustered group are all volkswagens. When manually looking at these observations it appears that these all belong to a specific model line named "Up". Weirdly, it appears that not all "Up" models are found to be outliers, although many of them have a similar price. From the summary, it seems that these observations are all cars with a low age, low mileage, and a low price, which is in constrats with most other cars.

```
par(mfrow=c(1,1))
hist(lmBest$residuals, freq=FALSE, breaks=20)
curve(dnorm(x, mean(lmBest$residuals), sd(lmBest$residuals)), col="blue", add = T)
```

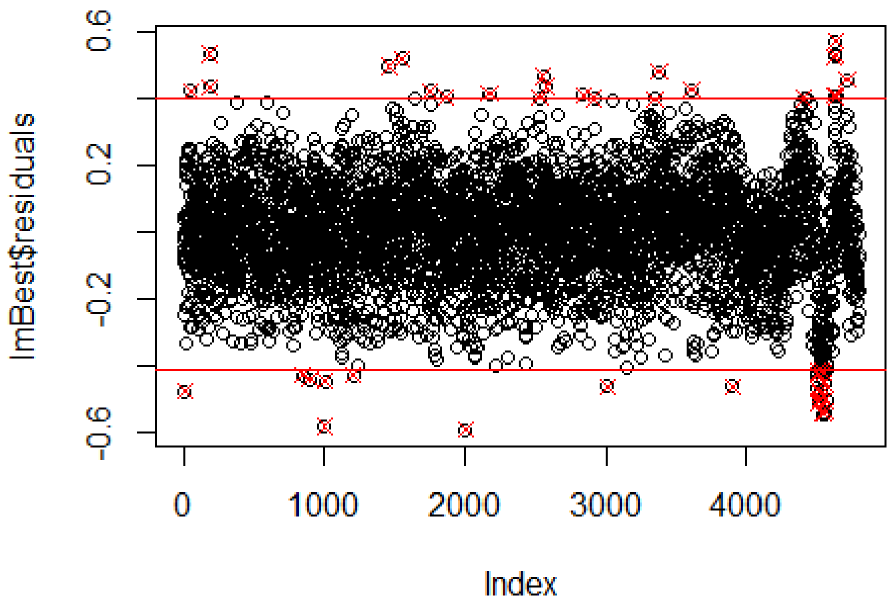




```
res.lower_bound <- quantile(lmBest$residuals, 0.005)
res.upper_bound <- quantile(lmBest$residuals, 0.995)
res.outl <- unname(which(lmBest$residuals > res.upper_bound | lmBest$residuals < res.lower
```

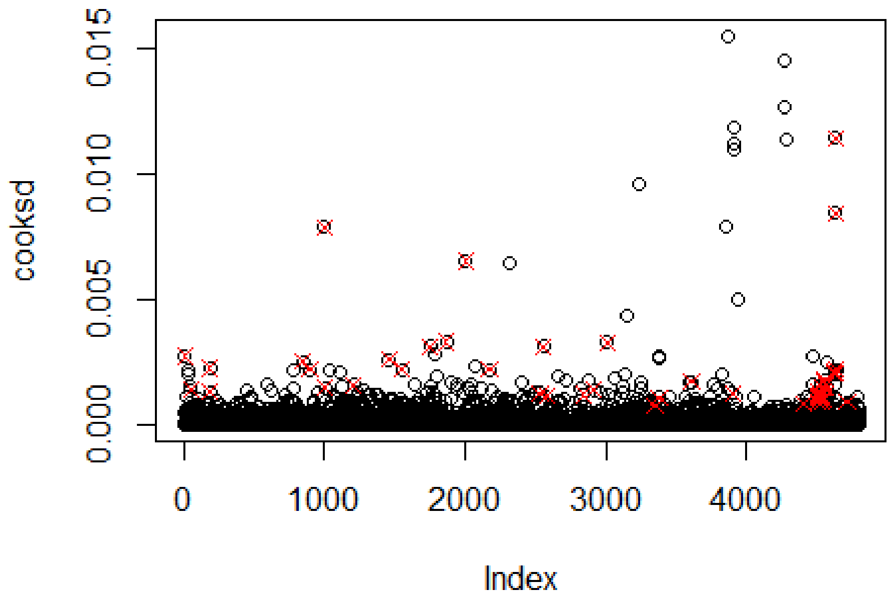
```
_bound))
length(res.outl)

## [1] 48

res.outl

##  [1]   13   56  190  191  852  896 1002 1007 1211 1462 1552 1754 1869 2006 2174
## [16] 2534 2558 2582 2840 2918 3013 3351 3379 3610 3902 4408 4501 4502 4503 4513
## [31] 4523 4525 4529 4536 4543 4544 4550 4551 4552 4556 4565 4626 4627 4631 4632
## [46] 4633 4638 4715

Boxplot(lmBest$residuals)

##  [1] 2006 1002 4551 4543 4552 4556 4513 4565 4525 4529 4633  190 4632 4626 1552
## [16] 1462 3379 2558 4715 2582

abline(h=res.upper_bound, col="red")
abline(h=res.lower_bound, col="red")
```




```
plot(lmBest$residuals)
abline(h=res.upper_bound, col="red")
abline(h=res.lower_bound, col="red")
points(res.outl, lmBest$residuals[res.outl], pch=4, col="red")
```




```
cooksd <- cooks.distance(lmBest)
plot(cooksd)
points(res.outl, cooksd[res.outl], pch=4, col="red")
```


```
res.outl_df <- df[res.outl,]
res.outl_df$orig_idx <- res.outl

names(res.outl_df)
```

```
##  [1] "model"           "year"            "price"           "transmission"
##  [5] "mileage"         "fuelType"        "tax"             "mpg"
##  [9] "engineSize"      "manufacturer"    "n.age"           "f.age"
## [13] "f.price"         "f.mileage"       "f.tax"           "f.mpg"
## [17] "f.engineSize"    "univ_outl_count" "orig_idx"
```

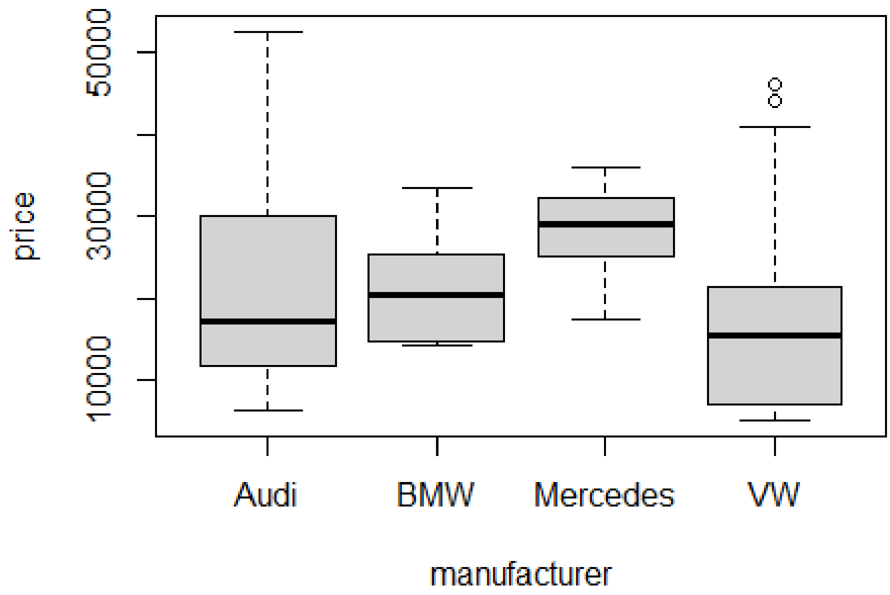
```
summary(res.outl_df[c(1:12)])
```

```
##     model                year          price           transmission
##  Length:48          Min.   :2011   Min.   : 5221   Automatic:14
##  Class :character   1st Qu.:2016   1st Qu.: 8187   Manual   :19
##  Mode  :character   Median :2017   Median :17895   Semi-Auto:15
##                     Mean   :2017   Mean   :19955
##                     3rd Qu.:2019   3rd Qu.:25964
##                     Max.   :2020   Max.   :52500
##     mileage         fuelType        tax              mpg           engineSize
##  Min.   :  100   Diesel:30   Min.   : 20.0   Min.   :32.80   Min.   :1.000
##  1st Qu.: 4969   Other : 0   1st Qu.:125.0   1st Qu.:44.80   1st Qu.:1.000
##  Median :15216   Petrol:18   Median :145.0   Median :55.95   Median :2.000
##  Mean   :23001               Mean   :127.7   Mean   :53.23   Mean   :1.808
##  3rd Qu.:33539               3rd Qu.:145.0   3rd Qu.:62.80   3rd Qu.:2.000
##  Max.   :94700               Max.   :325.0   Max.   :67.30   Max.   :3.000
##    manufacturer     n.age              f.age
##  Audi    : 8    Min.   :0.000   LowAge     :17
##  BMW     : 6    1st Qu.:1.000   LowMidAge  :12
##  Mercedes: 7    Median :3.000   HighMidAge: 8
##  VW      :27    Mean   :3.021   HighAge    :11
##                 3rd Qu.:4.000
##                 Max.   :9.000
```

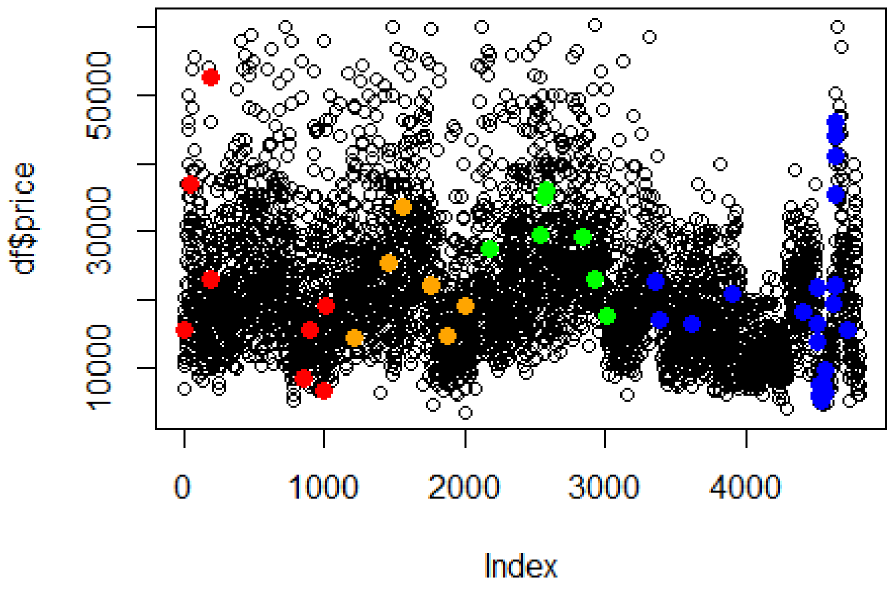
```
summary(df[c(1:12)])
```

```
##     model                year          price           transmission
##  Length:4803        Min.   :2009   Min.   : 3350   Automatic:1220
##  Class :character   1st Qu.:2016   1st Qu.:14298   Manual   :1722
##  Mode  :character   Median :2017   Median :19862   Semi-Auto:1861
##                     Mean   :2017   Mean   :21334
##                     3rd Qu.:2019   3rd Qu.:25995
##                     Max.   :2020   Max.   :60399
##     mileage          fuelType          tax              mpg
##  Min.   :     1   Diesel:2706   Min.   :  0.0   Min.   : 24.80
##  1st Qu.:   5628   Other :  30   1st Qu.:125.0   1st Qu.: 44.80
##  Median : 15865   Petrol:2067   Median :145.0   Median : 53.30
##  Mean   : 21626                 Mean   :122.8   Mean   : 52.98
##  3rd Qu.: 32235                 3rd Qu.:145.0   3rd Qu.: 61.40
##  Max.   :103160                 Max.   :330.0   Max.   :135.50
##    engineSize       manufacturer       n.age                 f.age
##  Min.   :1.000   Audi     :1048   Min.   : 0.000   LowAge     :1900
##  1st Qu.:1.500   BMW      :1021   1st Qu.: 1.000   LowMidAge  :1384
##  Median :2.000   Mercedes:1250    Median : 3.000   HighMidAge: 772
##  Mean   :1.898   VW       :1484   Mean   : 2.654   HighAge    : 747
##  3rd Qu.:2.000                    3rd Qu.: 4.000
##  Max.   :4.000                    Max.   :11.000
```

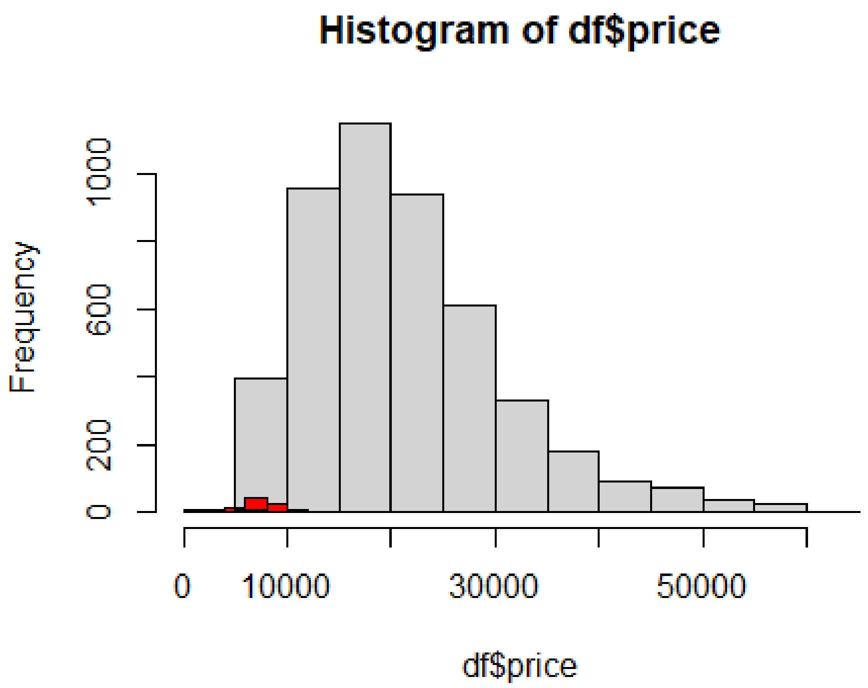
```
boxplot(price~manufacturer, data=res.outl_df)
```




```
plot(df$price)
points(res.outl_df$orig_idx[which(res.outl_df$manufacturer=="VW")], res.outl_df$price[which(res.outl_df$manufacturer=="VW")], pch=19, cex=1.2, col="blue")
points(res.outl_df$orig_idx[which(res.outl_df$manufacturer=="Audi")], res.outl_df$price[which(res.outl_df$manufacturer=="Audi")], pch=19, cex=1.2, col="red")
points(res.outl_df$orig_idx[which(res.outl_df$manufacturer=="Mercedes")], res.outl_df$price[which(res.outl_df$manufacturer=="Mercedes")], pch=19, cex=1.2, col="green")
points(res.outl_df$orig_idx[which(res.outl_df$manufacturer=="BMW")], res.outl_df$price[which(res.outl_df$manufacturer=="BMW")], pch=19, cex=1.2, col="orange")
```

```
hist(df$price)
hist(df$price[which(df$model==" Up")], col='red', add=T)
hist(res.outl_df$price[which(res.outl_df$model==" Up")], col="blue", add=T)
```



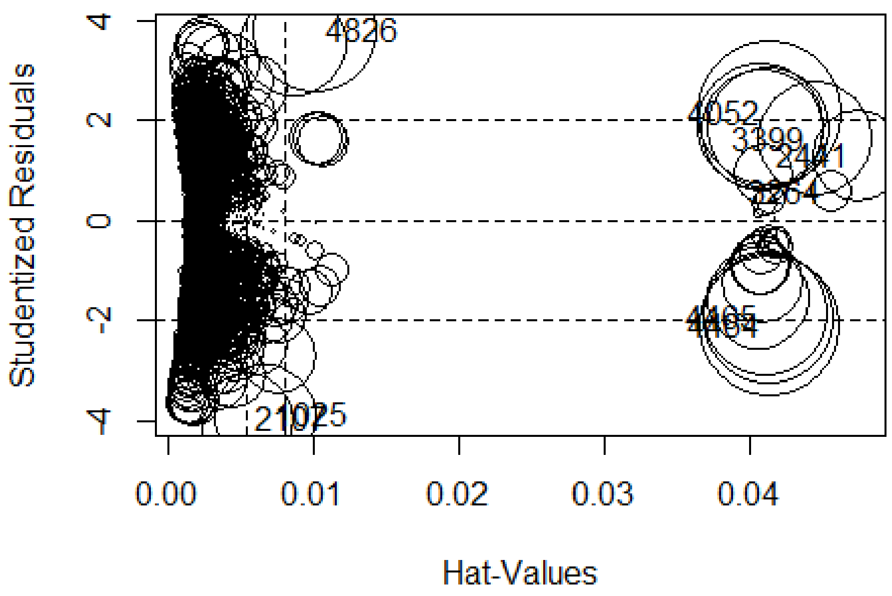

```
summary(df[which(df$model==" Up"), c(1:12)], col='red', add=T)

##     model                year          price          transmission
##  Length:79          Min.   :2013   Min.   : 4591   Automatic: 3   
##  Class :character   1st Qu.:2016   1st Qu.: 6985   Manual   :74   
##  Mode  :character   Median :2017   Median : 7562   Semi-Auto: 2   
##                     Mean   :2017   Mean   : 7917                  
##                     3rd Qu.:2019   3rd Qu.: 8997                  
##                     Max.   :2020   Max.   :15991                  
##     mileage         fuelType       tax               mpg          engineSize
##  Min.   :   10   Diesel: 0   Min.   :  0.0   Min.   :52.3   Min.   :1  
##  1st Qu.: 4426   Other : 0   1st Qu.: 20.0   1st Qu.:54.3   1st Qu.:1  
##  Median :15974   Petrol:79   Median : 20.0   Median :62.8   Median :1  
##  Mean   :18190               Mean   : 81.9   Mean   :60.9   Mean   :1  
##  3rd Qu.:27495               3rd Qu.:145.0   3rd Qu.:64.2   3rd Qu.:1  
##  Max.   :53000               Max.   :150.0   Max.   :68.9   Max.   :1  
##    manufacturer      n.age              f.age   
##  Audi     : 0    Min.   :0.000   LowAge    :21  
##  BMW      : 0    1st Qu.:1.000   LowMidAge :27  
##  Mercedes: 0    Median :3.000   HighMidAge:12  
##  VW       :79    Mean   :3.089   HighAge   :19  
##                  3rd Qu.:4.000                  
##                  Max.   :7.000                  

summary(res.outl_df[which(res.outl_df$model==" Up"), c(1:12)])

##     model                year          price         transmission
##  Length:12          Min.   :2014   Min.   :5221   Automatic: 0   
##  Class :character   1st Qu.:2015   1st Qu.:6148   Manual   :12   
##  Mode  :character   Median :2017   Median :6990   Semi-Auto: 0   
##                     Mean   :2016   Mean   :6952                  
##                     3rd Qu.:2017   3rd Qu.:7496                  
##                     Max.   :2020   Max.   :9700                  
##     mileage         fuelType       tax                mpg           engineSize
```

```
##  Min.   :  100   Diesel: 0   Min.   : 20.00   Min.   :54.30   Min.   :1
##  1st Qu.:14804   Other : 0   1st Qu.: 20.00   1st Qu.:62.12   1st Qu.:1
##  Median :19440   Petrol:12   Median : 82.50   Median :63.50   Median :1
##  Mean   :25577               Mean   : 83.75   Mean   :62.34   Mean   :1
##  3rd Qu.:36103               3rd Qu.:146.25   3rd Qu.:64.20   3rd Qu.:1
##  Max.   :53000               Max.   :150.00   Max.   :64.20   Max.   :1
##    manufacturer      n.age              f.age
##  Audi     : 0     Min.   :0.00   LowAge     :1
##  BMW      : 0     1st Qu.:3.00   LowMidAge  :6
##  Mercedes: 0      Median :3.00   HighMidAge:1
##  VW       :12     Mean   :3.75   HighAge    :4
##                   3rd Qu.:5.25
##                   Max.   :6.00
```

Study the presence of a priori influential data observations, indicating their number according to the criteria studied in class.

This question has mostly already been answered, high leverage observations are removed from the model. However, the proper cut off value was not used in this example. Therefore, this is shown in the following segment. In this case, only 5 a priori values where found, assuming that the dataset is large enough to use hat>3*mean(hat) instead of multiplying with 2.

```
high_lev <- as.data.frame(influencePlot(lmBest, id=list(n=3, method="noteworthy")))
```


```
mean_hat <- mean(high_lev$Hat)

priori <- row.names(high_lev[which(high_lev$hat>3*mean_hat)])
priori

## [1] "1025" "2107" "2441" "3264" "3399" "4052" "4464" "4465" "4826"
```

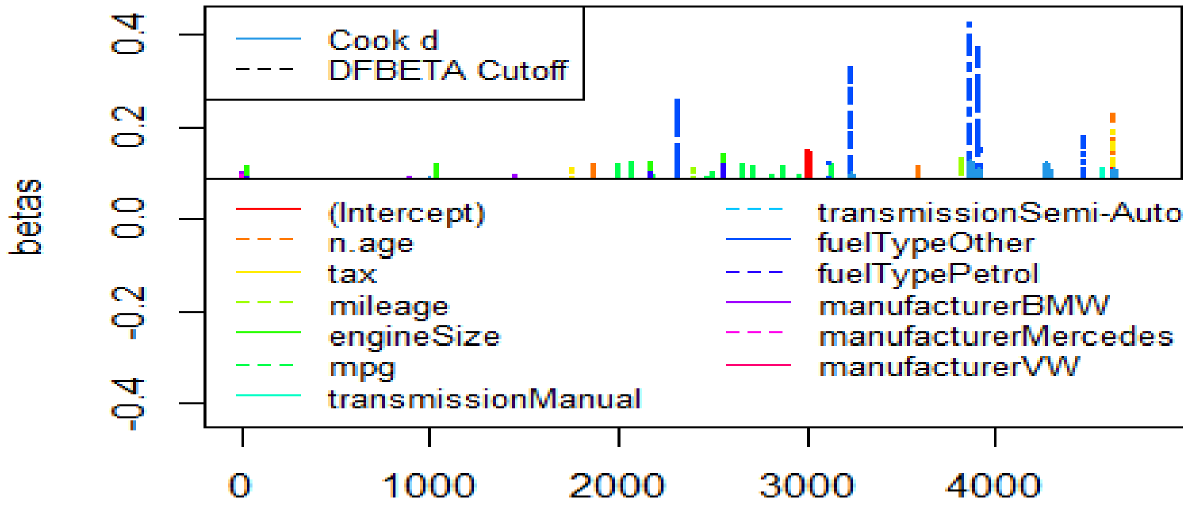
Study the presence of a posteriori influential values, indicating the criteria studied in class and the actual atypical observations.

A posteriori influential values are found using the dfbetas function. These are first plotted against their cut-off value, which is given by 2/sqrt(n). Then for each variable in the model, it is tested which observations are considered as a posteriori influential values, which are then temporarely removed from the data set (which already did not contain high leverage values).

The best model is then reconstructed on this new data set and compared with its original to see the difference. From the summary, it can be observed that the change mostly affected the coefficients of the tax, tranmission levels and manufacturer levels. The R-squared of the model has increased by 6% up to 0.9471.

Graphically, it can be observed that indeed several high leverage observations where taken out, which improves the models price variability coverage drastically. However, this model is build by deleting more then 20% of the data set, all highly influential points, and hence introduces a most likely significant bias in our predictions. Therefore, in the last model, the original best model from question 14 is used.

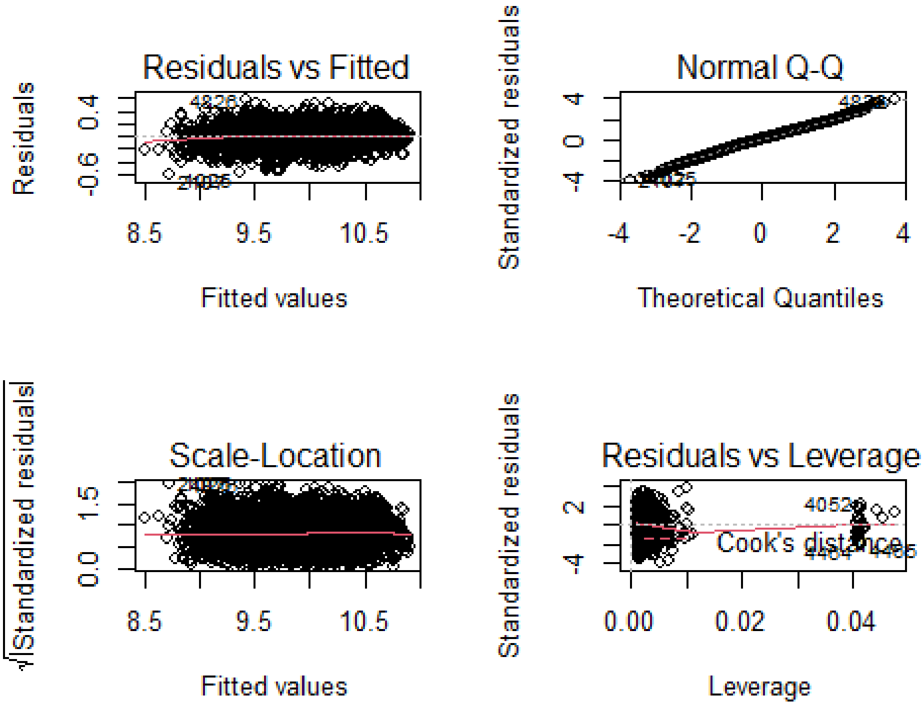
```
betas <- as.data.frame(dfbetas(lmBest))
betas_cutoff = 2 / sqrt(dim(df_no_high_infl)[1])
betas_cutoff

## [1] 0.02887353

par(mfrow=c(1,1))
matplot(betas, type="l", lwd=2, col= rainbow(ncol(betas)))
lines(sqrt(cooks.distance(lmBest)), col=4, lwd=3)
abline(h=betas_cutoff, lty=3,lwd=1,col=1)
abline(h=-betas_cutoff[1], lty=3,lwd=1,col=1)
legend("topleft", legend=c("Cook d", "DFBETA Cutoff"),
       col=c(4, 1), lty=1:2, cex=0.8)

legend("bottomleft", legend=names(coef(lmBest)),
       col=rainbow(ncol(betas)), lty=1:2, cex=0.8, ncol=2)
```




```
idx_list <- c()

idx_list <- append(idx_list, which(betas$n.age>betas_cutoff | betas$n.age< -betas_cutoff))
idx_list <- append(idx_list, which(betas$tax>betas_cutoff | betas$tax< -betas_cutoff))
idx_list <- append(idx_list, which(betas$mileage>betas_cutoff | betas$mileage< -betas_cuto
ff))
idx_list <- append(idx_list, which(betas$engineSize>betas_cutoff | betas$engineSize< -beta
s_cutoff))
idx_list <- append(idx_list, which(betas$transmissionManual>betas_cutoff | betas$transmiss
ionManual< -betas_cutoff))
idx_list <- append(idx_list, which(betas$`transmissionSemi-Auto`>betas_cutoff | betas$`tra
nsmissionSemi-Auto`< -betas_cutoff))
idx_list <- append(idx_list, which(betas$fuelTypeOther>betas_cutoff | betas$fuelTypeOther<
 -betas_cutoff))
idx_list <- append(idx_list, which(betas$fuelTypePetrol>betas_cutoff | betas$fuelTypePetro
l< -betas_cutoff))
idx_list <- append(idx_list, which(betas$manufacturerBMW>betas_cutoff | betas$manufacturer
BMW< -betas_cutoff))
idx_list <- append(idx_list, which(betas$manufacturerVW>betas_cutoff | betas$manufacturerV
W< -betas_cutoff))
idx_list <- append(idx_list, which(betas$`(Intercept)`>betas_cutoff | betas$`(Intercept)`<
 -betas_cutoff))
idx_list = unique(idx_list)

df_no_post <- df_no_high_infl[-idx_list,]

lmBest_no_posteriori <- lm(log(price) ~ n.age+tax+mileage+engineSize+mpg+transmission+fuel
Type+manufacturer, data=df_no_post)

summary(lmBest)

##
## Call:
## lm(formula = log(price) ~ n.age + tax + mileage + engineSize +
##     mpg + transmission + fuelType + manufacturer, data = df_no_high_infl)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

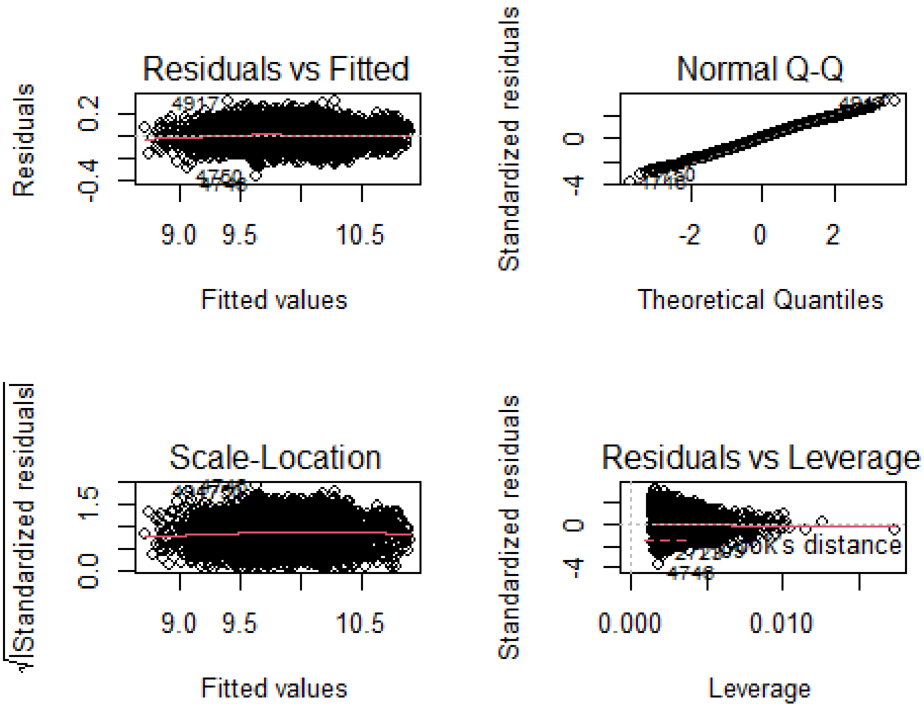
68

```
## -0.59566 -0.09277  0.00537  0.10074  0.57046
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             1.050e+01  3.123e-02 336.125  < 2e-16 ***
## n.age                  -9.778e-02  1.966e-03 -49.740  < 2e-16 ***
## tax                    -1.229e-04  4.998e-05  -2.458  0.01399 *
## mileage                -4.659e-06  1.878e-07 -24.807  < 2e-16 ***
## engineSize              2.710e-01  6.541e-03  41.440  < 2e-16 ***
## mpg                    -1.167e-02  3.551e-04 -32.869  < 2e-16 ***
## transmissionManual     -9.015e-02  6.651e-03 -13.554  < 2e-16 ***
## transmissionSemi-Auto   1.222e-02  5.629e-03   2.172  0.02994 *
## fuelTypeOther           1.740e-02  3.071e-02   0.567  0.57102
## fuelTypePetrol         -1.218e-01  6.647e-03 -18.325  < 2e-16 ***
## manufacturerBMW        -8.382e-02  6.971e-03 -12.023  < 2e-16 ***
## manufacturerMercedes    2.069e-02  6.771e-03   3.055  0.00226 **
## manufacturerVW         -1.768e-01  6.266e-03 -28.218  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.151 on 4785 degrees of freedom
## Multiple R-squared:  0.8857, Adjusted R-squared:  0.8854
## F-statistic:  3091 on 12 and 4785 DF,  p-value: < 2.2e-16

summary(lmBest_no_posteriori)

##
## Call:
## lm(formula = log(price) ~ n.age + tax + mileage + engineSize +
##     mpg + transmission + fuelType + manufacturer, data = df_no_post)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36375 -0.07109  0.00282  0.07094  0.30242
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             1.041e+01  2.639e-02 394.590  < 2e-16 ***
## n.age                  -9.470e-02  1.589e-03 -59.600  < 2e-16 ***
## tax                    -4.027e-05  3.849e-05  -1.046    0.295
## mileage                -4.861e-06  1.508e-07 -32.240  < 2e-16 ***
## engineSize              2.849e-01  5.713e-03  49.869  < 2e-16 ***
## mpg                    -1.100e-02  2.878e-04 -38.218  < 2e-16 ***
## transmissionManual     -8.630e-02  5.035e-03 -17.140  < 2e-16 ***
## transmissionSemi-Auto   1.697e-02  4.248e-03   3.995 6.59e-05 ***
## fuelTypeOther           1.887e-02  9.816e-02   0.192    0.848
## fuelTypePetrol         -1.067e-01  5.538e-03 -19.275  < 2e-16 ***
## manufacturerBMW        -7.399e-02  5.310e-03 -13.933  < 2e-16 ***
## manufacturerMercedes    2.228e-02  5.111e-03   4.359 1.34e-05 ***
## manufacturerVW         -1.682e-01  4.681e-03 -35.932  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09802 on 3670 degrees of freedom
## Multiple R-squared:  0.946,  Adjusted R-squared:  0.9458
## F-statistic:  5353 on 12 and 3670 DF,  p-value: < 2.2e-16

par(mfrow=c(2,2))
plot(lmBest)
```




```
plot(lmBest_no_posteriori)

## Warning: not plotting observations with leverage one:
##   1135
```

70



Given a 5-year old car, the rest of numerical variables on the mean and factors on the reference level, what would be the expected price with a 95% confidence interval?

Reference factors are transmission = Automatic, fuelType = Diesel and manufacturer = Audi. Using the best model, the price is expected to lie in interval [17530.93, 18110.67] with a confidence of 95 %.

```
sample <- data.frame(n.age=5, tax=mean(df_no_high_infl$tax), mileage=mean(df_no_high_infl$
mileage), engineSize=mean(df_no_high_infl$engineSize), mpg=mean(df_no_high_infl$mpg), tran
smission="Automatic", fuelType="Diesel", manufacturer="Audi")

sam.fit <- predict.lm(lmBest, sample, se.fit=TRUE, interval="confidence", level=0.95)

exp(sam.fit$fit)

##        fit      lwr      upr
## 1 17820.13 17532.34 18112.64
```

Summarize what you have learned by working with this interesting real dataset.

This project is a clear example of how a data set that at first doesn't look that complex requires a lot of careful preprocessing, transformation, analysis and constant re-analysis.

The preprocessing part and exploratory analysis are components we were already a bit more familiar with and thus were carried out rather quickly. However, at first, multivariate outliers weren't accounted for, such that are initial results were quite different and more imbalanced than after these were removed from the data set. This showcased the importance of careful preprocessing in the sense that skipping steps can severely influence later results.

The actual body of the project consisted mostly of building a lot of linear models and its accompanying diagnostic plots. This is an intensive process but step by step improves your best model and at the same time reveal a lot of information in a data set. A simple example of this is the need of the logarithmic transformation which drastically improves the variability coverage of your model but at the same time decreases the influence and interaction of some explanatory variables, thus yielding information about the actual influence of attributes in a data set. Furthermore, we believe that balancing between improving the response variables' variabilty coverage and not overfitting or adding to much complexity and/or degrees of freedom is a ever recurring and crucial reality in most data science projects.

To sum up, working with this data set is an important reference of how to deal with typical difficulties as well as an overview of what to expect in data science projects.