Deliverable 1: Data Processing, Description, Validation and Profiling

Soukaina Mahboub Mehboub

2023-10-22

1. Introduction

1.1 Description

This report presents an exploratory analysis of the 100,000 UK used car dataset. The dataset includes information from four major car manufacturers: Audi, BMW, Mercedes, and Volkswagen. The data consists of details such as car model, registration year, price, gearbox type, mileage, engine fuel, road tax, consumption in miles per gallon, and engine size.

To make the analysis manageable and insightful, a random sample of 5,000 records has been selected from this extensive dataset.

Data from: https://www.kaggle.com/adityadesai13/used-car-dataset-ford-and-mercedes

1.2 Dataset Overview

1.2.1 Variables

- Manufacturer: The car's manufacturer (Audi, BMW, Mercedes, or Volkswagen).
- Model: The specific model of the car.
- Year: The registration year of the car.
- Price: The price of the car in £.
- Transmission: The type of gearbox (e.g., Manual, Semi-Auto, Automatic).
- Mileage: The distance the car has been used.
- Fuel Type: The type of engine fuel (e.g., Diesel, Petrol, Hybrid).
- Tax: The road tax for the car.
- MPG: Consumption in miles per gallon.
- Engine Size: The size of the car's engine in liters.

1.3 Data preparation

As our initial step, we'll start by downloading the essential packages and libraries required for our project. It's crucial to ensure that these packages are properly installed to avoid any issues later on. Once that's accomplished, our next task involves creating a subset of our dataset with 5000 specific observations. It's important to note that during this process, we will maintain the complete set of original variables, ensuring that no data is lost.

We'll now upload the data and proceed to create our sample by randomly selecting 5000 records.

Sample overview (Dimension of the dataframe (number of rows and columns), the names of variables and brief statistical summary (including measures such as mean, median, quartiles, and counts for each variable)).

```r
str(df) # Variable types
```

```
## 'data.frame':    5000 obs. of  10 variables:
##  $ model       : chr  " 1 Series" " GLE Class" " Caddy Maxi Life" " Golf" ...
##  $ year        : int  2017 2018 2019 2019 2016 2019 2018 2017 2018 2019 ...
##  $ price       : int  19761 44738 19000 17990 25412 16930 20310 15498 17250 16555 ...
##  $ transmission: chr  "Semi-Auto" "Semi-Auto" "Automatic" "Manual" ...
##  $ mileage     : int  39681 21276 13191 1201 24346 5317 14863 62140 7629 9451 ...
##  $ fuelType    : chr  "Petrol" "Diesel" "Diesel" "Diesel" ...
##  $ tax         : int  200 150 145 145 160 145 145 145 150 145 ...
##  $ mpg         : num  39.8 36.7 44.1 57.7 51.4 49.6 53.3 64.2 56.5 68.9 ...
##  $ engineSize  : num  3 3 2 1.6 3 1.6 1.4 2 1.4 2 ...
##  $ manufacturer: chr  "BMW" "Mercedes" "VW" "VW" ...
```

```r
dim(df) # Displays the sample size
```

```
## [1] 5000   10
```

```r
names(df) # Displays the names of the sample variables
```

```
##  [1] "model"        "year"         "price"        "transmission" "mileage"
##  [6] "fuelType"     "tax"          "mpg"          "engineSize"   "manufacturer"
```

```r
summary(df)
```

```
##     model                year          price         transmission
##  Length:5000        Min.   :1998   Min.   :   899   Length:5000
##  Class :character   1st Qu.:2016   1st Qu.: 13994   Class :character
##  Mode  :character   Median :2017   Median : 19500   Mode  :character
##                     Mean   :2017   Mean   : 21573
##                     3rd Qu.:2019   3rd Qu.: 26499
##                     Max.   :2020   Max.   :154998
##     mileage         fuelType              tax             mpg
##  Min.   :     1   Length:5000        Min.   :  0.0   Min.   : 21.10
##  1st Qu.:  5866   Class :character   1st Qu.:125.0   1st Qu.: 44.10
##  Median : 16698   Mode  :character   Median :145.0   Median : 52.30
##  Mean   : 23309                      Mean   :125.5   Mean   : 53.67
##  3rd Qu.: 33646                      3rd Qu.:145.0   3rd Qu.: 61.40
##  Max.   :323000                      Max.   :580.0   Max.   :470.80
##    engineSize    manufacturer
##  Min.   :0.000   Length:5000
##  1st Qu.:1.500   Class :character
##  Median :2.000   Mode  :character
##  Mean   :1.927
##  3rd Qu.:2.000
##  Max.   :6.200
```

2. Univariate Descriptive Analysis

Prior to examining individual variables, we'll establish counters to track missing values, errors, and outliers within the vectors.

We will also detect all the missing values in the dataframe and store them in two vectors (initial missings for the individuals and for each variable.

```
mis1<-countNA(df)
imis<-mis1$mis_ind
#mis1$mis_col # Number of missings for the current set of variables
jmis<-mis1$mis_col$mis_x

iouts<-rep(0,nrow(df))  # rows - trips
jouts<-rep(0,ncol(df))  # columns - variables


ierrs<-rep(0,nrow(df))  # rows - trips
jerrs<-rep(0,ncol(df))  # columns - variables
```

2.1 Factors: Categorical Variables

Categorical variables should be converted to factors for appropriate analysis to enhance data analysis and enabling effective grouping, summarization, and visualization.

Model (1)

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

```
##  [1] "Audi- A1"           "Audi- A3"           "Audi- A4"
##  [4] "Audi- A5"           "Audi- A6"           "Audi- A7"
##  [7] "Audi- A8"           "Audi- Q2"           "Audi- Q3"
## [10] "Audi- Q5"           "Audi- Q7"           "Audi- Q8"
## [13] "Audi- R8"           "Audi- RS3"          "Audi- RS4"
## [16] "Audi- RS5"          "Audi- RS6"          "Audi- S3"
## [19] "Audi- S4"           "Audi- S8"           "Audi- SQ5"
## [22] "Audi- TT"           "BMW- 1 Series"      "BMW- 2 Series"
## [25] "BMW- 3 Series"      "BMW- 4 Series"      "BMW- 5 Series"
## [28] "BMW- 6 Series"      "BMW- 7 Series"      "BMW- 8 Series"
## [31] "BMW- i3"            "BMW- M2"            "BMW- M3"
## [34] "BMW- M4"            "BMW- M5"            "BMW- M6"
## [37] "BMW- X1"            "BMW- X2"            "BMW- X3"
## [40] "BMW- X4"            "BMW- X5"            "BMW- X6"
## [43] "BMW- X7"            "BMW- Z3"            "BMW- Z4"
## [46] "Mercedes- A Class"  "Mercedes- B Class"  "Mercedes- C Class"
## [49] "Mercedes- CL Class" "Mercedes- CLA Class" "Mercedes- CLS Class"
## [52] "Mercedes- E Class"  "Mercedes- G Class"  "Mercedes- GL Class"
## [55] "Mercedes- GLA Class" "Mercedes- GLB Class" "Mercedes- GLC Class"
## [58] "Mercedes- GLE Class" "Mercedes- GLS Class" "Mercedes- M Class"
## [61] "Mercedes- S Class"  "Mercedes- SL CLASS"  "Mercedes- SLK"
## [64] "Mercedes- V Class"  "Mercedes- X-CLASS"  "VW- Amarok"
## [67] "VW- Arteon"         "VW- Beetle"         "VW- Caddy"
## [70] "VW- Caddy Maxi"     "VW- Caddy Maxi Life" "VW- California"
## [73] "VW- Caravelle"      "VW- CC"             "VW- Fox"
## [76] "VW- Golf"           "VW- Golf SV"        "VW- Passat"
## [79] "VW- Polo"           "VW- Scirocco"       "VW- Sharan"
## [82] "VW- Shuttle"        "VW- T-Cross"        "VW- T-Roc"
## [85] "VW- Tiguan"         "VW- Tiguan Allspace" "VW- Touareg"
## [88] "VW- Touran"         "VW- Up"
```

Transmission (4)

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

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

```r
df$transmission <- factor( df$transmission, levels = c("Manual","Semi-Auto","Automatic"),labels = paste(
```

FueltType (6)

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

Manufacturer (10)

```r
df$manufacturer <- factor( df$manufacturer )
```

2.2 Exploratory Data Analysis and Data Quality

2.2.1 Categorical Variables - Factors

Model (1):

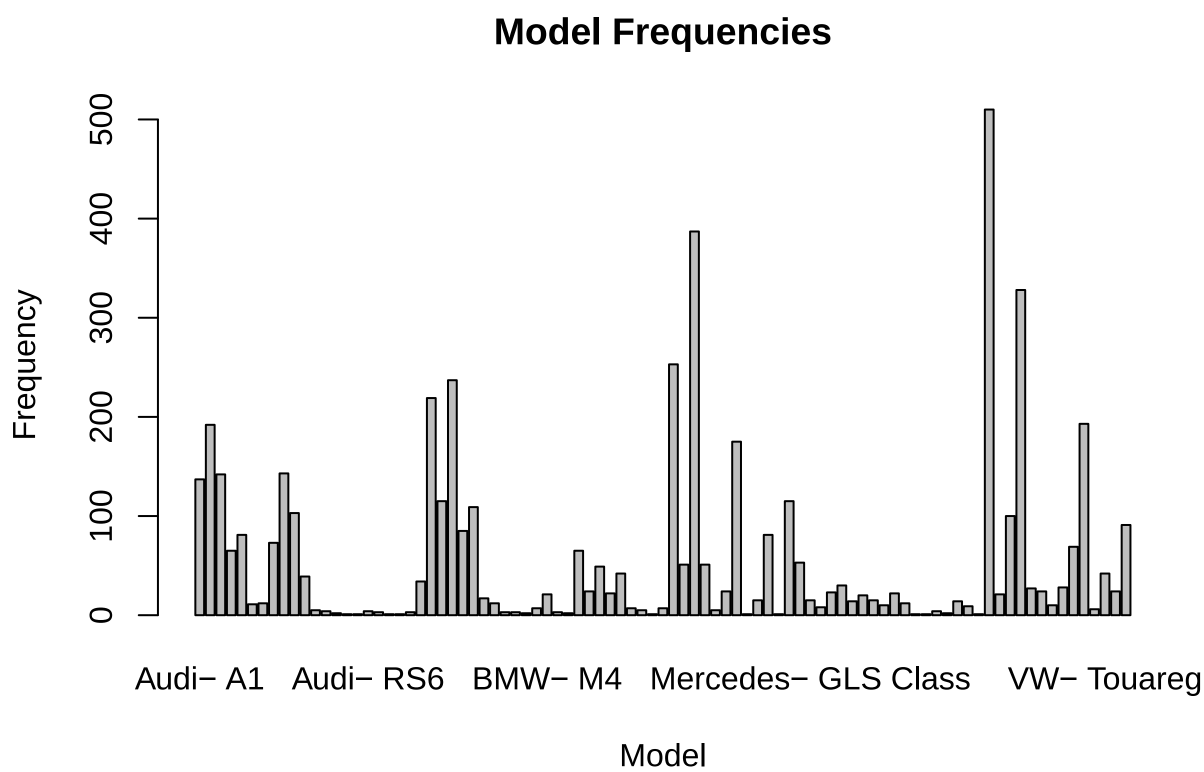
In this variable, the presence of numerous car models makes it challenging to identify missing values through a barplot. To tackle this, we will primarily utilize functions such as table() and is.na() to assess the distribution of cars across each model and employ is.na() for missing value detection.

```r
summary(df$model)
```

```
##            Audi- A1           Audi- A3           Audi- A4           Audi- A5
##                 137                192                142                 65
##            Audi- A6           Audi- A7           Audi- A8           Audi- Q2
##                  81                 11                 12                 73
##            Audi- Q3           Audi- Q5           Audi- Q7           Audi- Q8
##                 143                103                 39                  5
##            Audi- R8          Audi- RS3          Audi- RS4          Audi- RS5
##                   4                  2                  1                  1
##           Audi- RS6           Audi- S3           Audi- S4           Audi- S8
##                   4                  3                  1                  1
##           Audi- SQ5           Audi- TT      BMW- 1 Series      BMW- 2 Series
##                   3                 34                219                115
##       BMW- 3 Series      BMW- 4 Series      BMW- 5 Series      BMW- 6 Series
##                 237                 85                109                 17
##       BMW- 7 Series      BMW- 8 Series           BMW- i3            BMW- M2
##                  12                  3                  3                  2
##             BMW- M3            BMW- M4            BMW- M5            BMW- M6
##                   7                 21                  3                  2
##             BMW- X1            BMW- X2            BMW- X3            BMW- X4
##                  65                 24                 49                 22
##             BMW- X5            BMW- X6            BMW- X7            BMW- Z3
##                  42                  7                  5                  1
##             BMW- Z4  Mercedes- A Class  Mercedes- B Class  Mercedes- C Class
##                   7                253                 51                387
##  Mercedes- CL Class Mercedes- CLA Class Mercedes- CLS Class  Mercedes- E Class
##                  51                  5                 24                175
##   Mercedes- G Class  Mercedes- GL Class Mercedes- GLA Class Mercedes- GLB Class
##                   1                 15                 81                  1
## Mercedes- GLC Class Mercedes- GLE Class Mercedes- GLS Class  Mercedes- M Class
##                 115                 53                 15                  8
```

```
##    Mercedes- S Class  Mercedes- SL CLASS       Mercedes- SLK   Mercedes- V Class
##                   23                  30                  14                  20
##    Mercedes- X-CLASS          VW- Amarok          VW- Arteon          VW- Beetle
##                   15                  10                  22                  12
##            VW- Caddy      VW- Caddy Maxi VW- Caddy Maxi Life      VW- California
##                    1                   1                   4                   2
##        VW- Caravelle              VW- CC             VW- Fox            VW- Golf
##                   14                   9                   1                 510
##         VW- Golf SV          VW- Passat            VW- Polo        VW- Scirocco
##                   21                 100                 328                  27
##           VW- Sharan         VW- Shuttle         VW- T-Cross           VW- T-Roc
##                   24                  10                  28                  69
##           VW- Tiguan VW- Tiguan Allspace         VW- Touareg          VW- Touran
##                  193                   6                  42                  24
##               VW- Up
##                   91
```

```r
barplot(table(df$model), main = "Model Frequencies", xlab = "Model", ylab = "Frequency")
```



Detecting any missing values:

```r
#Detecting any missing values as previous barplot cannot show missing values:
na_values <- is.na(df$model)
any(na_values)
```

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

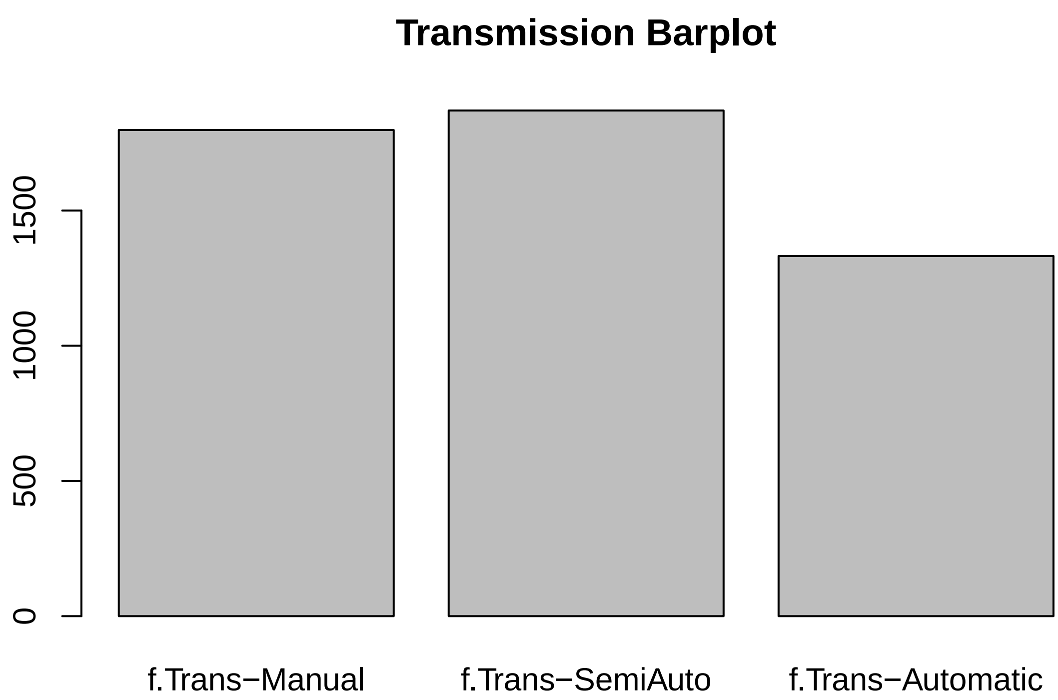
Transmission (2):

Zero missing values, and cars are nearly evenly distributed across three categories. No errors or outliers are present (as these three are the only three possible transmission types in cars).

```
summary(df$transmission)
```

```
##    f.Trans-Manual  f.Trans-SemiAuto f.Trans-Automatic
##              1798              1870              1332
```

```
barplot(summary(df$transmission),main="Transmission Barplot")
```



FuelType (6):

As we can see, the summary reveals that there are 15 NA's in this variable, and very few cars are hybrid

At this stage we will consider missing values as electrical cars if their engine-size are zero (This assumption will help us analyze the "engineSize" variable later).

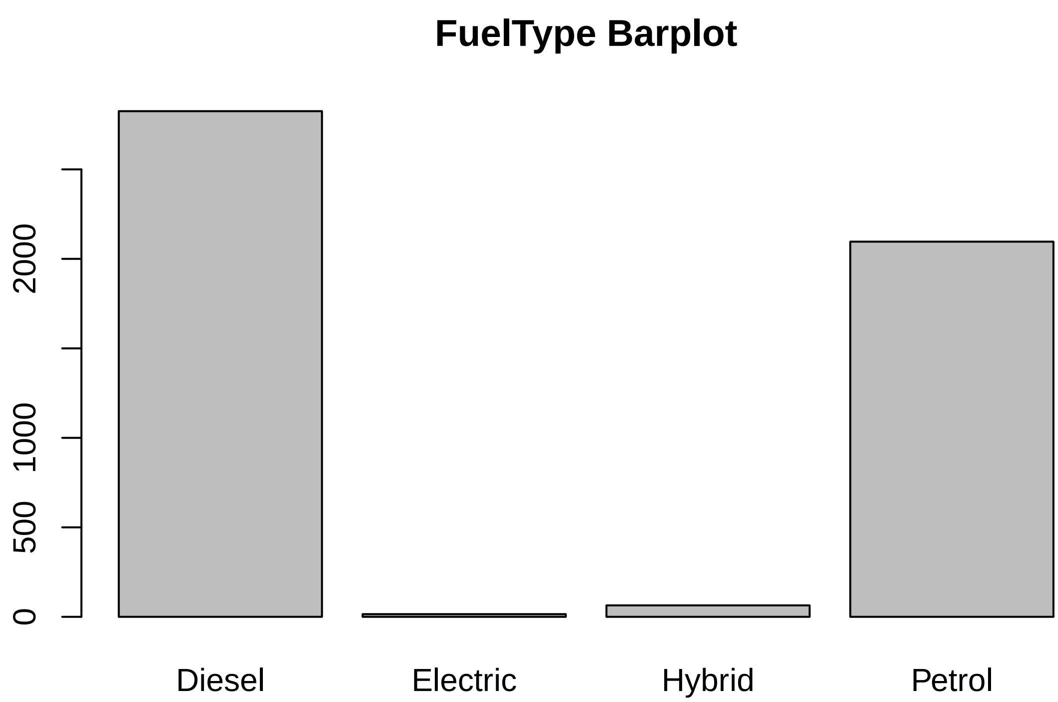
```
summary(df$fuelType)
```

```
## Diesel Hybrid  Other Petrol
##   2825     64     15   2096
```

```
#Mark NA's as Electric car
na_rows <- which(df$fuelType == 'Other')
#convert variable back to character (to avoid warnings)
df$fuelType <- as.character(df$fuelType)
df$fuelType[na_rows] <- 'Electric'
#convert variable back to factor
df$fuelType <- as.factor(df$fuelType)
```

FuelType Distribution:

```r
#Barplot
barplot(summary(df$fuelType),main="FuelType Barplot")
```



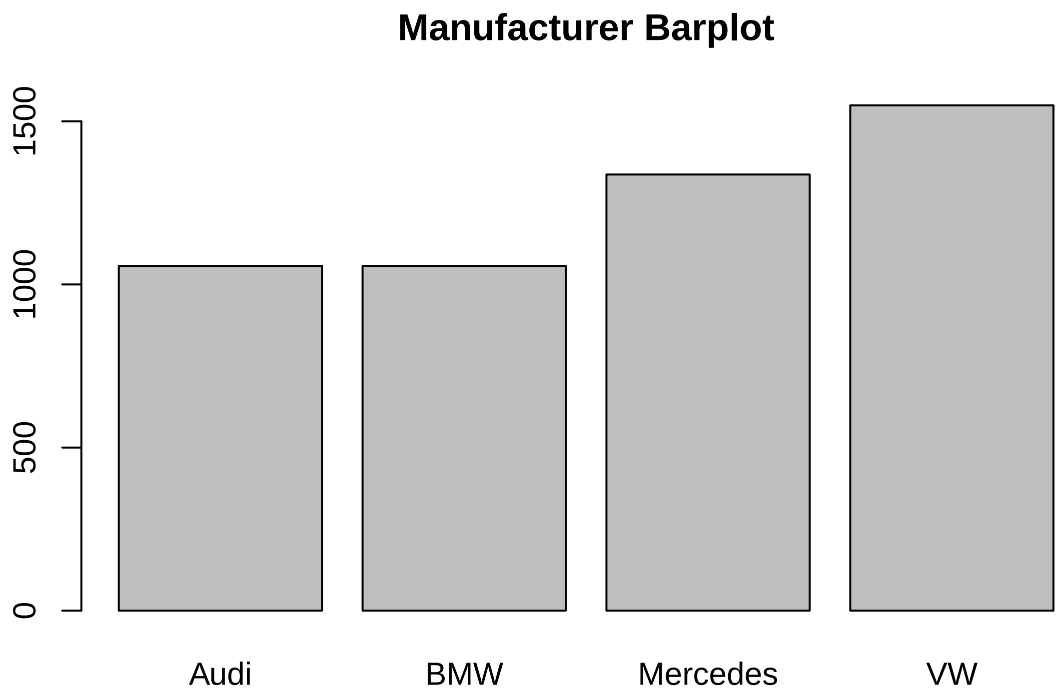
Manufacturer (10):

Every vehicle in our sample is sourced from one of the four manufacturers that contributed to our dataset. We've detected no missing values. Since our sample was selected randomly, we have a slightly higher representation of VW and Mercedes cars compared to Audi and BMW. For this variable, no missing, errors, or outliers data has been identified.

```r
summary(df$manufacturer)
```

```
##     Audi      BMW Mercedes       VW
##     1057     1057     1337     1549
```

```r
barplot(summary(df$manufacturer),main="Manufacturer Barplot")
```



2.2.2 Numerical Variables

We will consistently detect missing outliers in all numerical variables using the same method, which involves identifying both low and high outliers. This approach ensures that the R script remains adaptable to changes in datasets or samples without requiring modifications.

Year (2):

The summary indicates that the 'year' values fall within the valid range of 1998 to 2020, demonstrating the absence of errors or inconsistencies. Given that 'year' is typically represented as an integer, we'll ensure any potential decimal values are rounded to maintain data integrity.

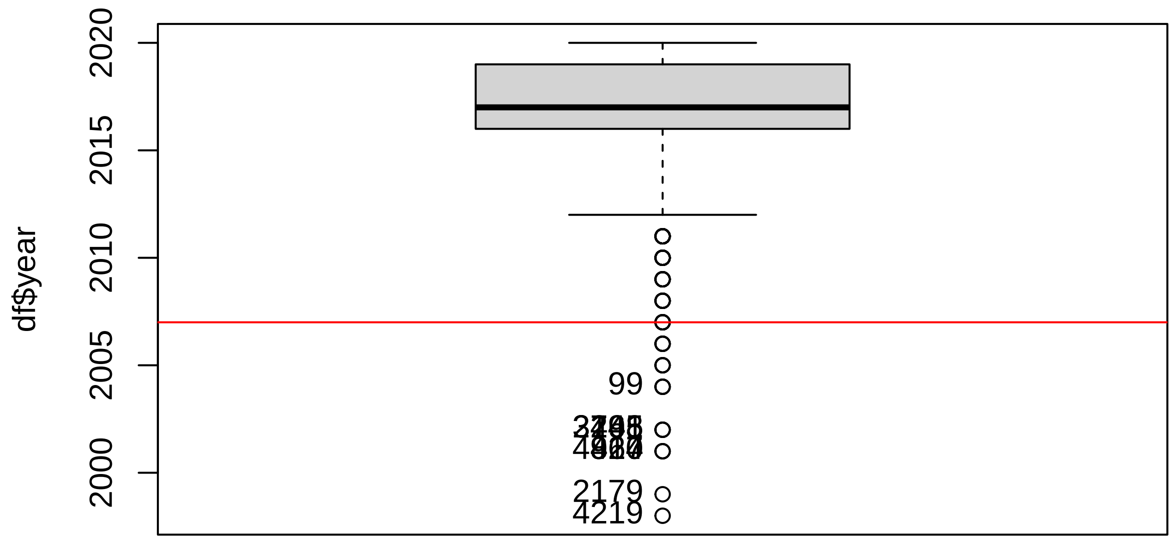
```
summary(df$year)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1998    2016    2017    2017    2019    2020
```

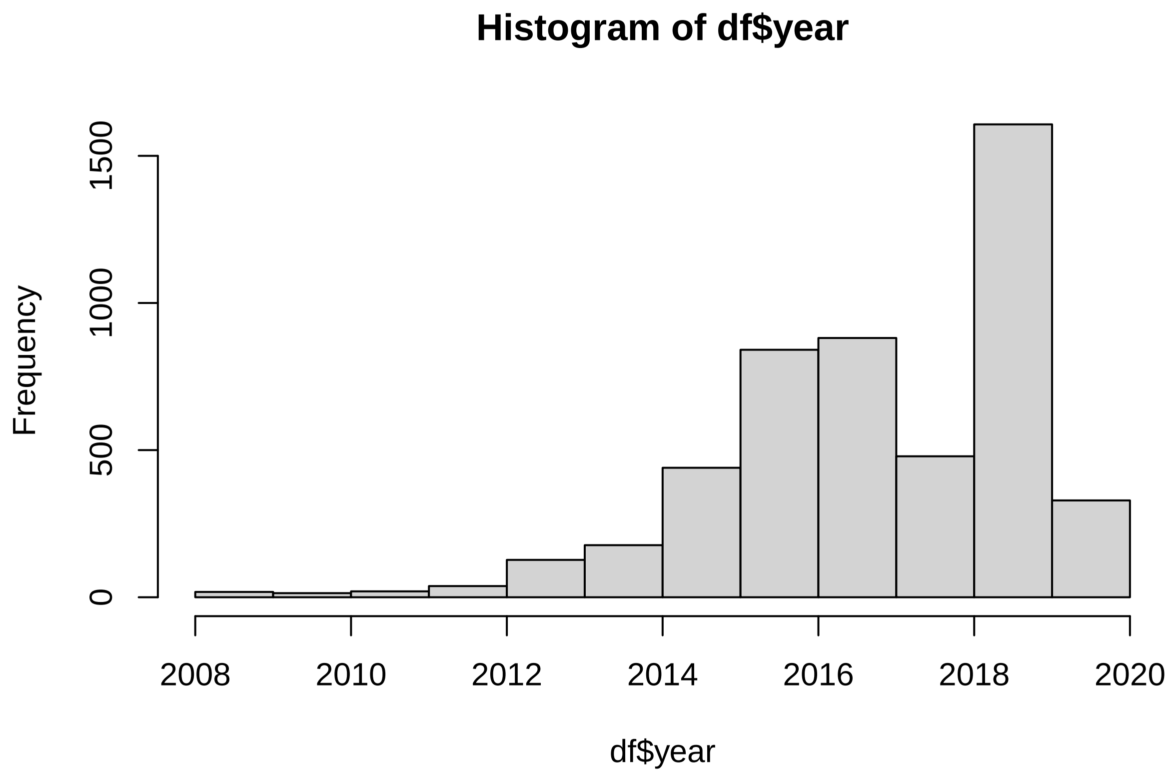
```
df$year <- as.integer(df$year)

# Outlier detection
Boxplot(df$year)
```

```
##  [1] 4219 2179  460  814 4927  248 2495 3165 3741   99
```

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




```r
sel <- which(df$year <= var_out$souti);
iouts[sel]<-iouts[sel]+1
jouts[2]<-jouts[2]+length(sel)
df[sel, "year"] <- NA

sel <- which(df$year >= var_out$souts);
iouts[sel]<-iouts[sel]+1
jouts[2]<-jouts[2]+length(sel)
df[sel, "year"] <- NA

hist(df$year)  #Distribution of "year"
```



Price (3):

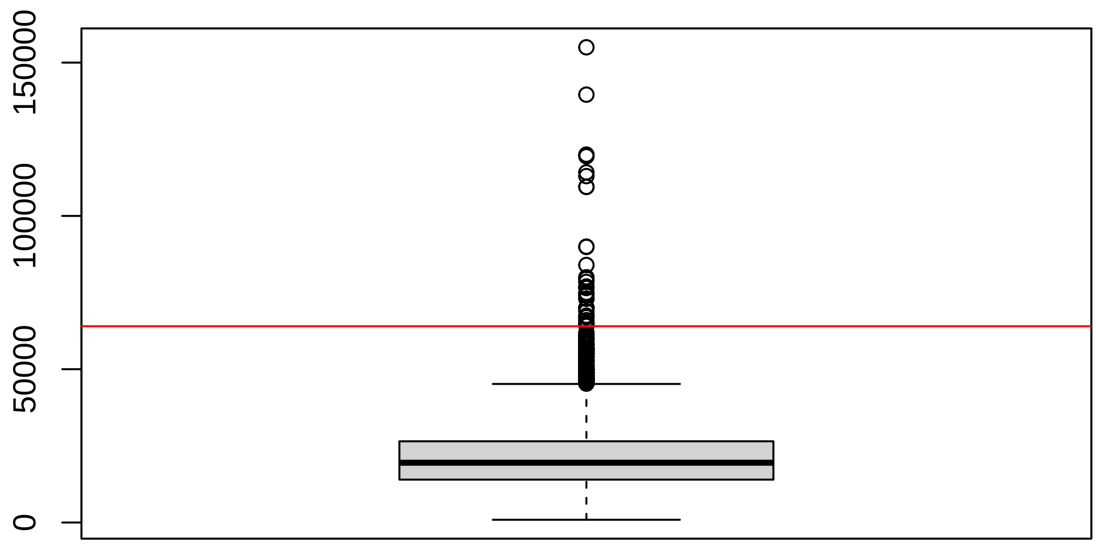
No missing values, no errors identified, and all values fall within a reasonable range, reflecting real car prices in the current market. We'll focus on excluding only the most extreme outliers.

As "price" is out Target Variable, we won't do imputations, so we won't assign NA value to outliers.

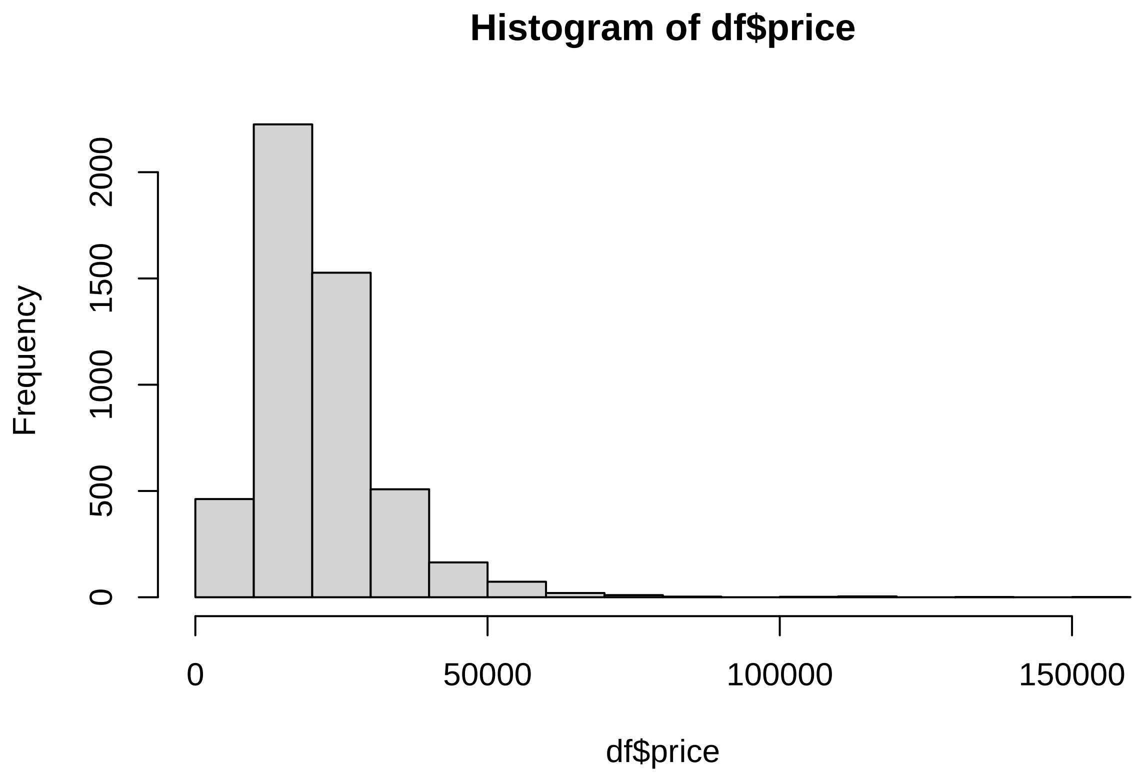
```
summary(df$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     899   13994   19500   21573   26499  154998
```

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




```
sel <- which(df$price <= var_out$souti);
iouts[sel]<-iouts[sel]+1
jouts[3]<-jouts[3]+length(sel)


sel <- which(df$price >= var_out$souts);
iouts[sel]<-iouts[sel]+1
jouts[3]<-jouts[3]+length(sel)

hist(df$price)  #Distribution of "price"
```



Mileage (5):

No missing values or errors are present, given the logical and positive range of all mileage values. Our focus will be on the exclusion of extreme outliers.

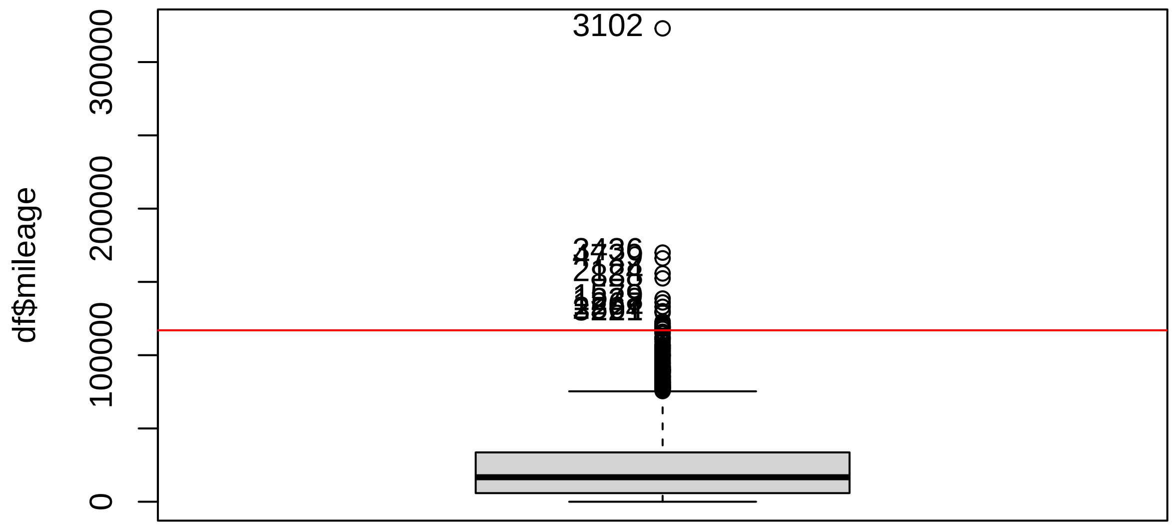
```
summary(df$mileage)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1    5866   16698   23309   33646  323000
```

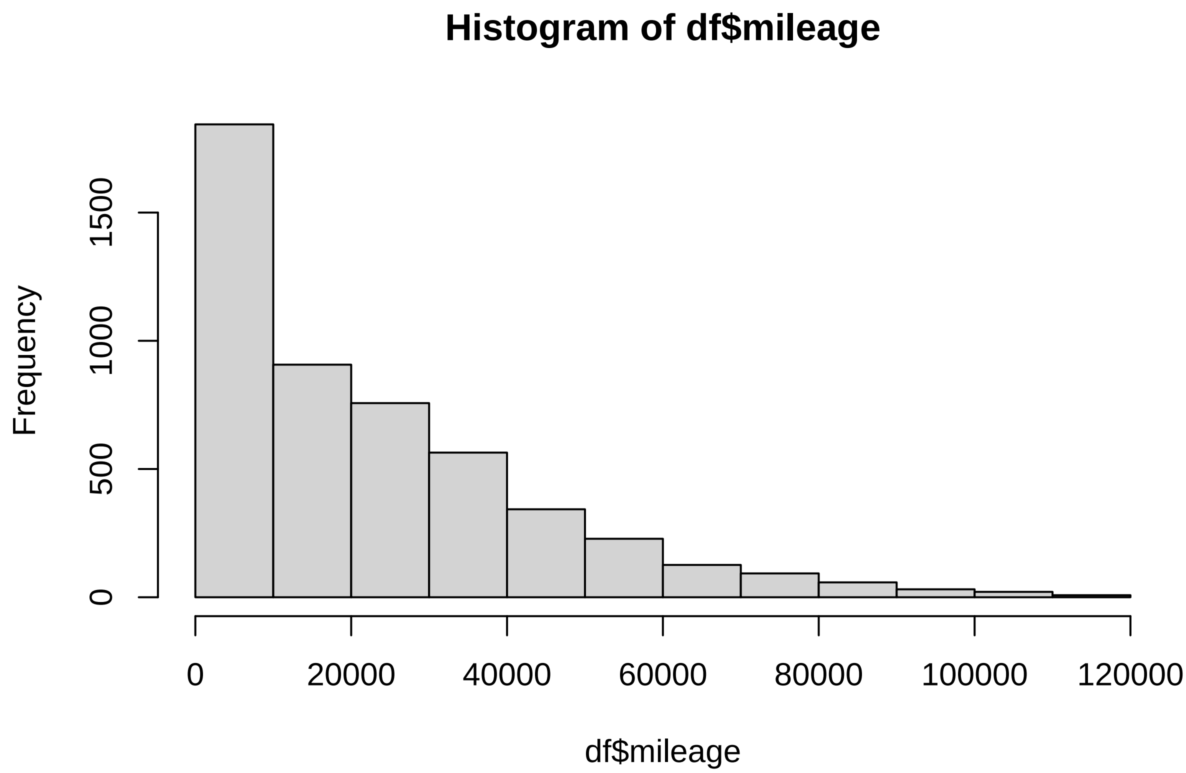
```
# Outlier detection
Boxplot(df$mileage)
```

```
##  [1] 3102 3436 4729 2124  888 1579 1228 1267 2564 3221
```

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




```
sel <- which(df$mileage >= var_out$souts);
iouts[sel]<-iouts[sel]+1
jouts[5]<-jouts[5]+length(sel)
df[sel, "mileage"] <- NA

sel <- which(df$mileage <= var_out$souti);
iouts[sel]<-iouts[sel]+1
jouts[5]<-jouts[5]+length(sel)
df[sel, "mileage"] <- NA

hist(df$mileage)  #Distribution of "mileage"
```



Tax (7):

The summary reveals that there are instances of zero tax values. This is a possibility in specific cases within the UK, considering the dataset's origin.

The tax values are within expected ranges, so our primary concern is identifying extreme outliers.

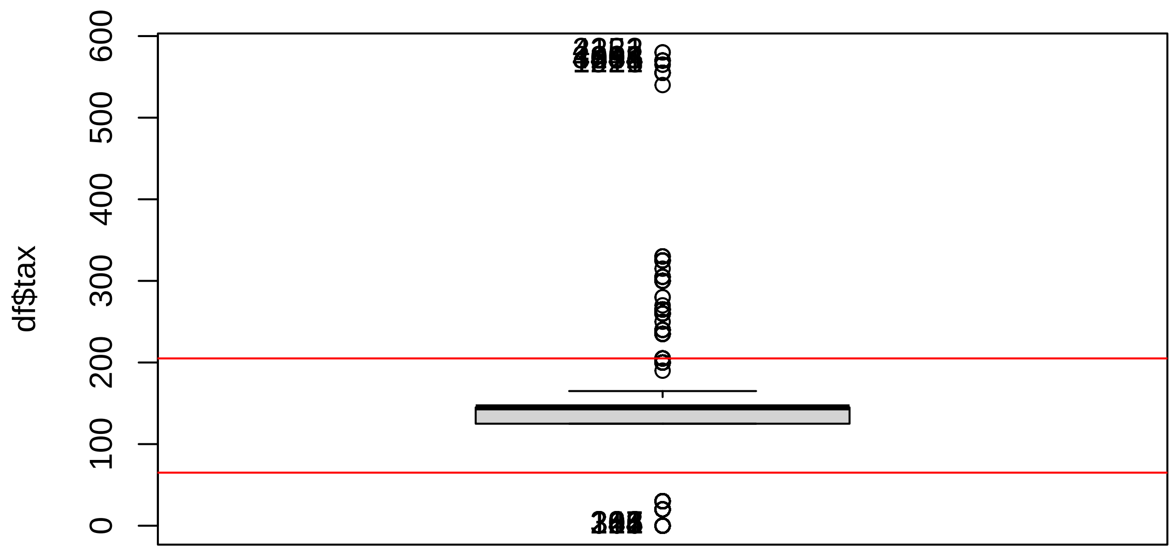
```
summary(df$tax)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   125.0   145.0   125.5   145.0   580.0
```

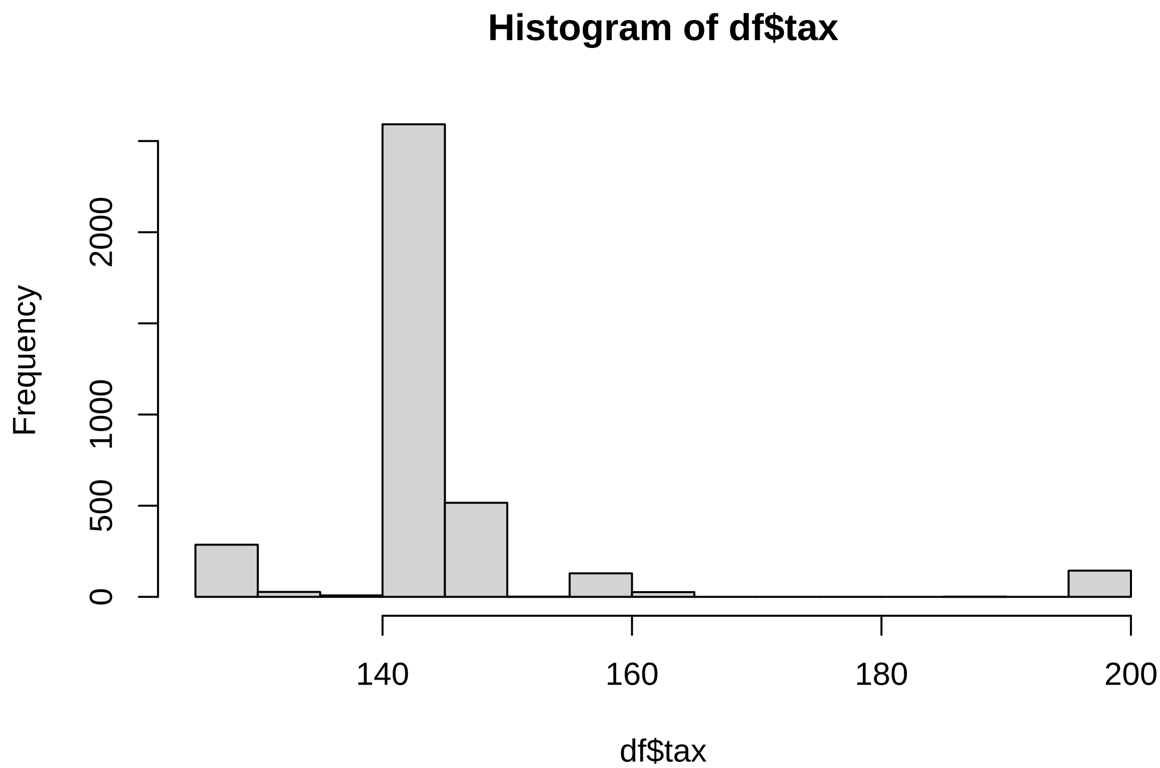
```
# Outlier detection
Boxplot(df$tax)
```

```
##  [1]  101  112  165  206  244  268  316  317  321  381 2131 4252  248  361 3095
## [16] 4434 4604 4682 1221 1916
```

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




```
sel <- which(df$tax >= var_out$souts);
iouts[sel]<-iouts[sel]+1
jouts[7]<- jouts[7] +length(sel)
df[sel, "tax"] <- NA


sel <- which(df$tax <= var_out$souti);
iouts[sel]<-iouts[sel]+1
jouts[7]<- jouts[7] +length(sel)
df[sel, "tax"] <- NA

hist(df$tax)  #Distribution of "tax"
```



MPG (8):

As we can observe from the summary, there are no missing values in this variable. However, it's worth noting that some values are significantly higher than what would be considered normal for miles per gallon (mpg), even though they fall within the possible range. To identify and address these extreme outliers, we will proceed with outlier detection.

Note: We will assume that electric cars, which have an MPG value, are represented as MPGe (Miles Per Gallon Equivalent), in order to prevent any data loss.

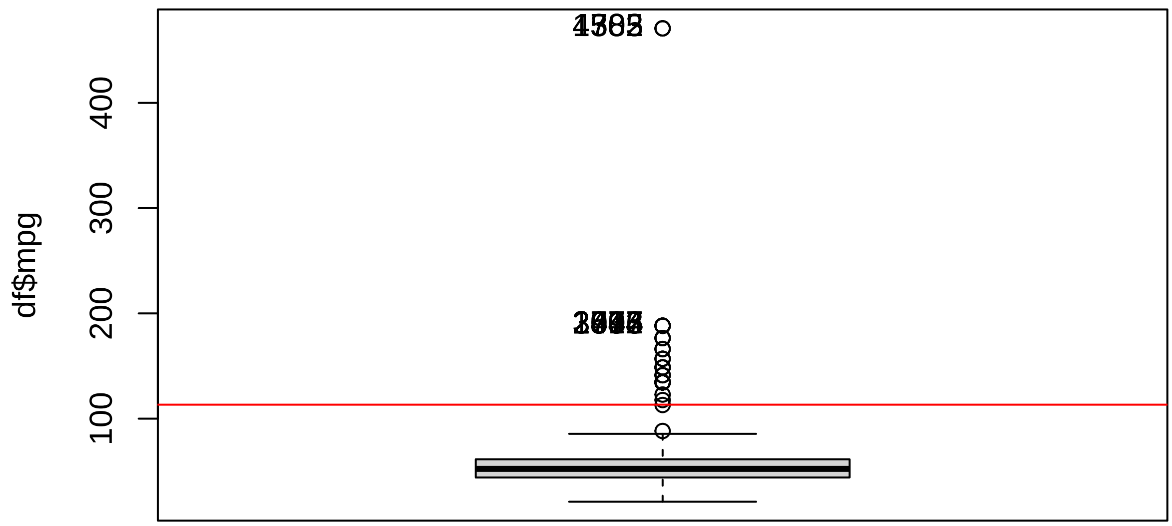
```
summary(df$mpg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   21.10   44.10   52.30   53.67   61.40  470.80
```

```
# Outlier detection
Boxplot(df$mpg)
```

```
##  [1]  383 1785 4502  515 1636 2073 2472 2747 3604 3994
```

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



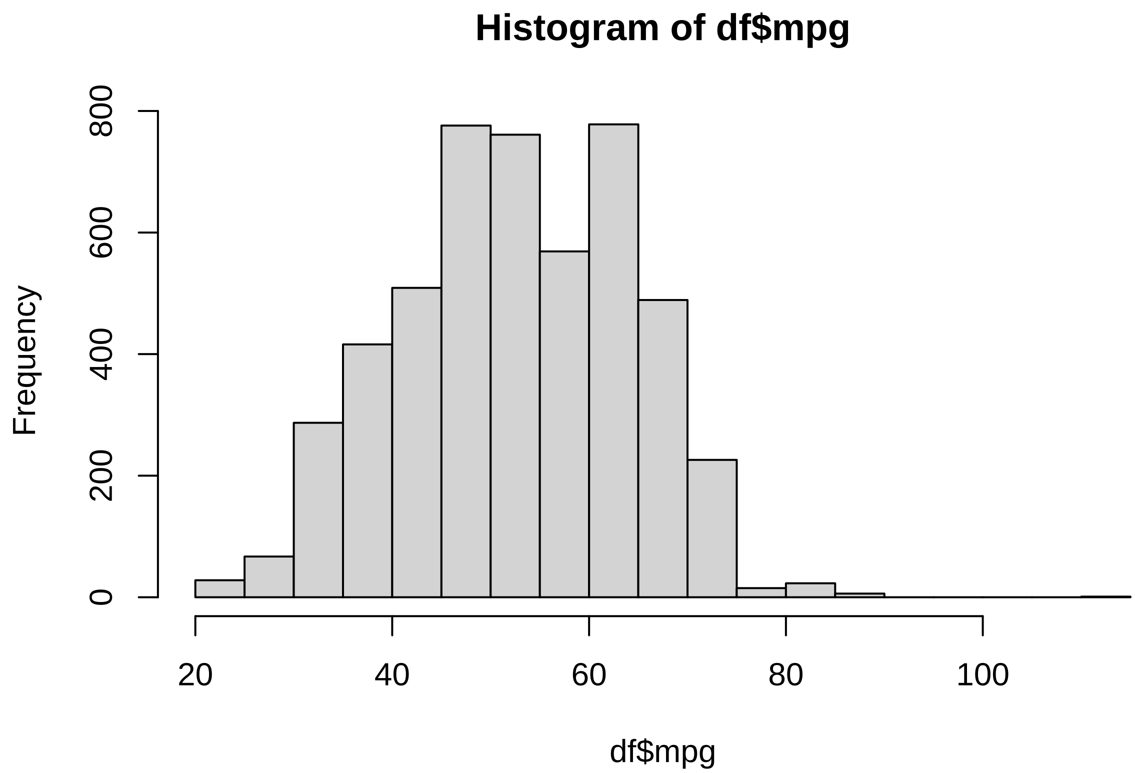

```
var_out$souts
```

```
## 3rd Qu.
##   113.3
```

```
var_out$souti
```

```
## 1st Qu.
##    -7.8
```

```
sel <- which(df$mpg >= var_out$souts);

iouts[sel]<-iouts[sel]+1
jouts[8]<- jouts[8] +length(sel)
df[sel, "mpg"] <- NA


sel <- which(df$mpg <= var_out$souti);

iouts[sel]<-iouts[sel]+1
jouts[8]<- jouts[8] +length(sel)
df[sel, "mpg"] <- NA

hist(df$mpg)  #Distribution of "mpg"
```



Engine size (9):

Through summary, we can see that we have no missing values here. However, we spotted some errors. When a car's engine size is listed as 0, it usually means the car is electric. However, some cars, like the Mercedes C class, might also show 0 as the engine size, but they are not electric; this could be a data issue. It is also an error to find Hybrid, Petrol and Diesel with an engine size 0.

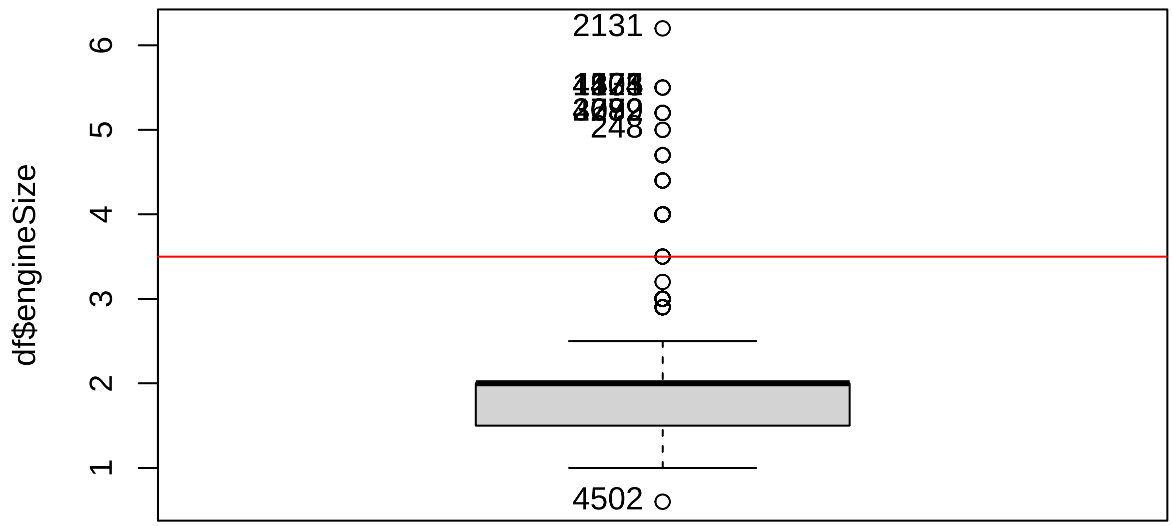
```
summary(df$engineSize)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.500   2.000   1.927   2.000   6.200
```

```
sel <- which(df$engineSize == 0 & (df$model == "Mercedes- C Class" | df$fuelType != "Electric"))

ierrs[sel]<-ierrs[sel]+1
jerrs[9]<-length(sel)
df[sel,"engineSize"]<-NA

# Outlier detection
Boxplot(df$engineSize)
```

```
##  [1] 4502 2131 1173 1221 4434 4505  799 2272 3032 4682  248
```

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




```
sel <- which(df$engineSize >= var_out$souts);
iouts[sel]<-iouts[sel]+1
jouts[9]<- jouts[9] +length(sel)
df[sel, "engineSize"] <- NA


sel <- which(df$engineSize <= var_out$souti);
iouts[sel]<-iouts[sel]+1
jouts[9]<- jouts[9] +length(sel)
df[sel, "engineSize"] <- NA


hist(df$engineSize)  #Distribution of "engineSize"
```

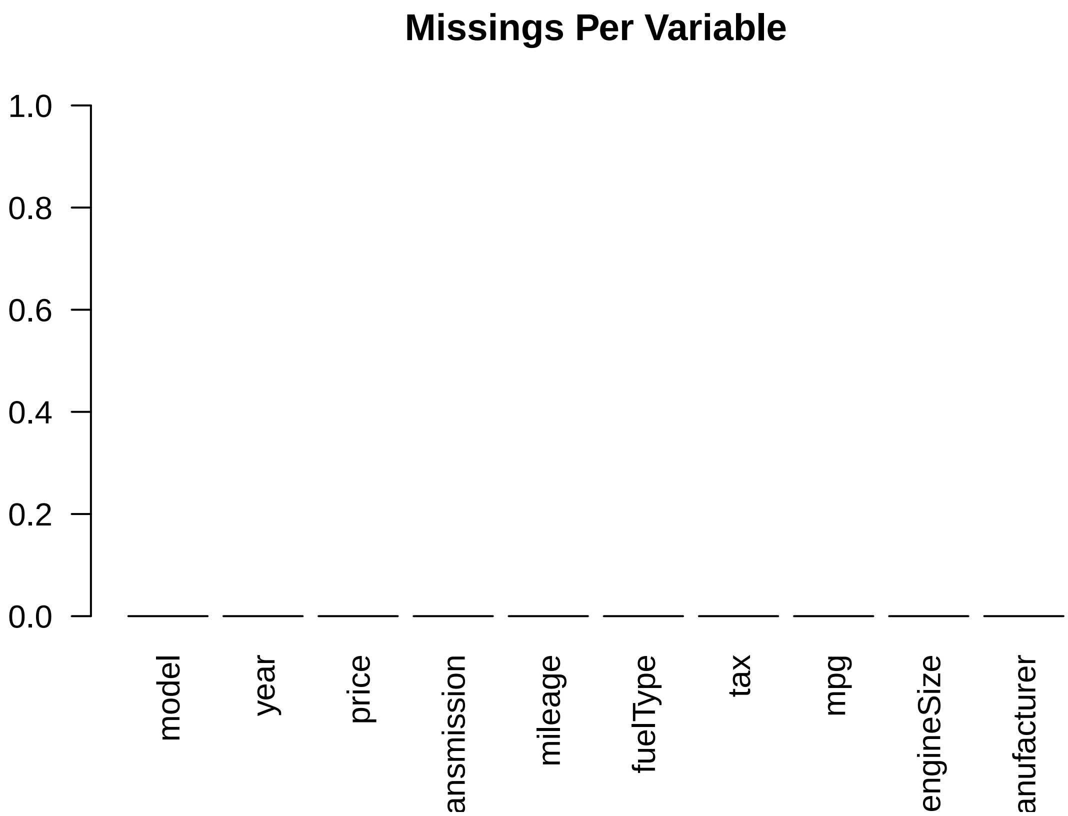
3. Data Quality

3.1 Per Variable

3.1.1 Missings

As we can we see, initially we have no missing values to begin it.

```
labels <- colnames(df[1:10])
# Barplot
barplot(mis1$mis_col$mis_x, names.arg = labels, main = "Missings Per Variable", col = "grey", ylim = c(0
```



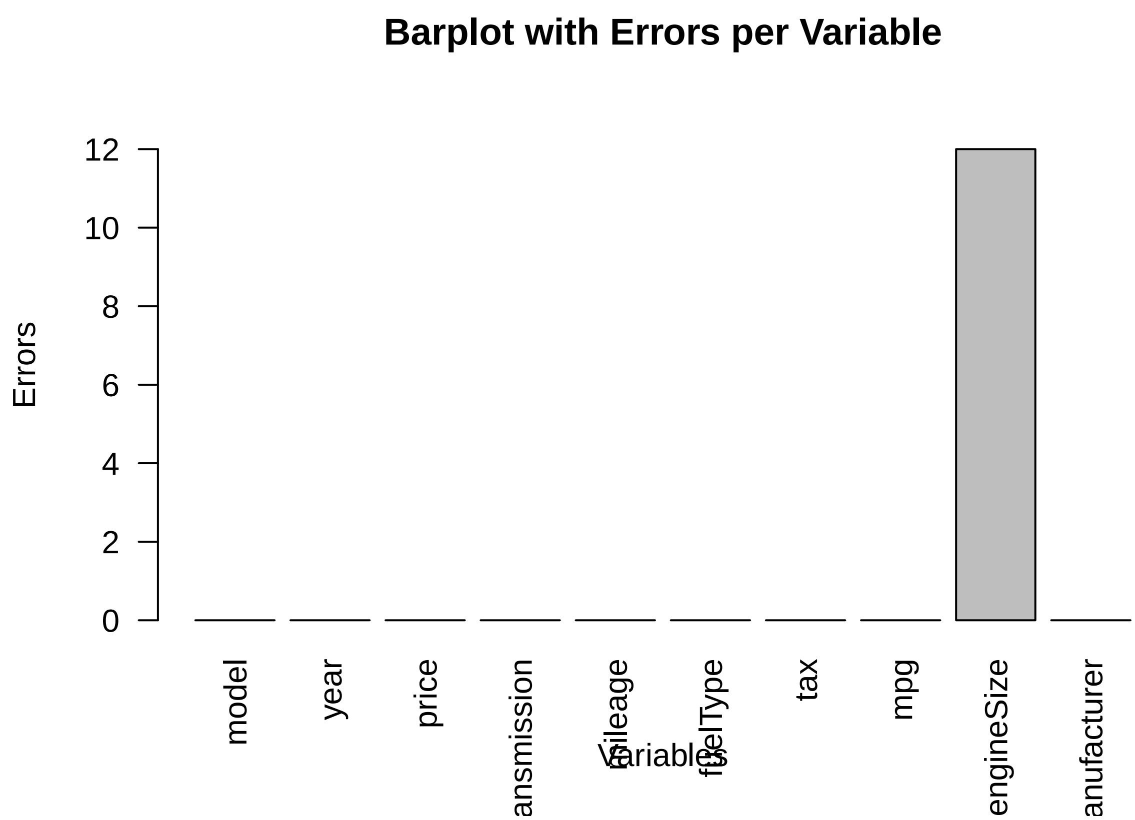
3.1.2 Errors

Only 12 errors in engineSize:

```r
jerrs
```

```
##  [1]  0  0  0  0  0  0  0  0 12  0
```

```r
# Barplot
barplot(jerrs[1:10], names.arg = labels,
        main = "Barplot with Errors per Variable",
        xlab = "Variables", ylab = "Errors",
        col = "grey",
        ylim = c(0, max(jerrs) + 1),
        las = 2)
```

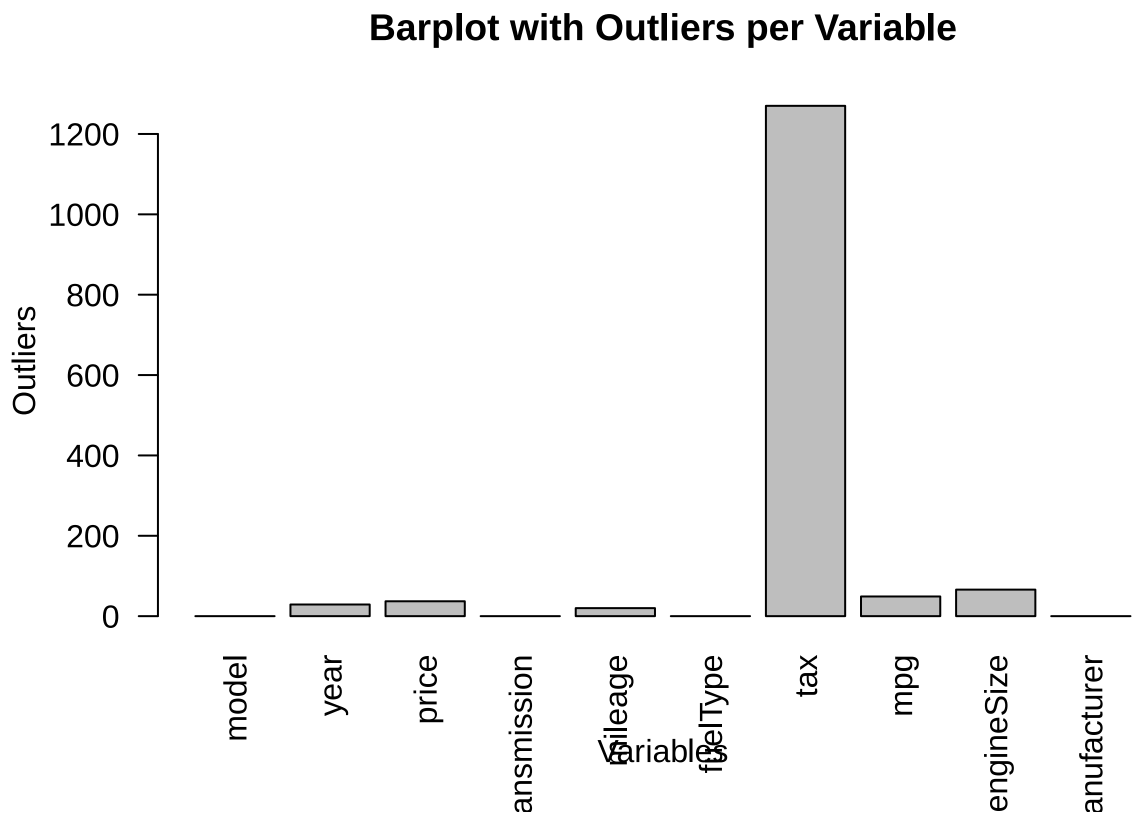


3.1.3 Outliers

```
jouts
```

```
##  [1]    0   29   37    0   20    0 1270   49   66    0
```

```
# Barplot
barplot(jouts[1:10], names.arg = labels,
        main = "Barplot with Outliers per Variable",
        xlab = "Variables", ylab = "Outliers",
        col = "grey",
        ylim = c(0, max(jouts) + 1),
        las = 2)
```



Summary and ranking variables based on number of missings, errors and outliers:

```
# Dataframe with the counts
counts_df <- data.frame(
  Variable = labels,
  Errors = jerrs[1:10],
  Missings = jmis[1:10],
  Outliers = jouts[1:10]
)

# Sort variables based on counts
sorted_errors <- counts_df[order(-counts_df$Errors), c('Variable', 'Errors')]
sorted_missings <- counts_df[order(-counts_df$Missings), c('Variable', 'Missings')]
sorted_outliers <- counts_df[order(-counts_df$Outliers), c('Variable', 'Outliers')]

# Variables and their respective counts for each category
cat("Variables Sorted by Errors:")
```

```
## Variables Sorted by Errors:
```

```
print(sorted_errors)
```

```
##        Variable Errors
## 9    engineSize     12
## 1         model      0
## 2          year      0
## 3         price      0
## 4  transmission      0
```

```
## 5       mileage      0
## 6      fuelType      0
## 7           tax      0
## 8           mpg      0
## 10 manufacturer      0
```

```r
cat("Variables Sorted by Missing Values:")
```

```
## Variables Sorted by Missing Values:
```

```r
print(sorted_missings)
```

```
##        Variable Missings
## 1         model        0
## 2          year        0
## 3         price        0
## 4  transmission        0
## 5       mileage        0
## 6      fuelType        0
## 7           tax        0
## 8           mpg        0
## 9    engineSize        0
## 10 manufacturer        0
```

```r
cat("Variables Sorted by Outliers:")
```

```
## Variables Sorted by Outliers:
```

```r
print(sorted_outliers)
```

```
##        Variable Outliers
## 7           tax     1270
## 9    engineSize       66
## 8           mpg       49
## 3         price       37
## 2          year       29
## 5       mileage       20
## 1         model        0
## 4  transmission        0
## 6      fuelType        0
## 10 manufacturer        0
```

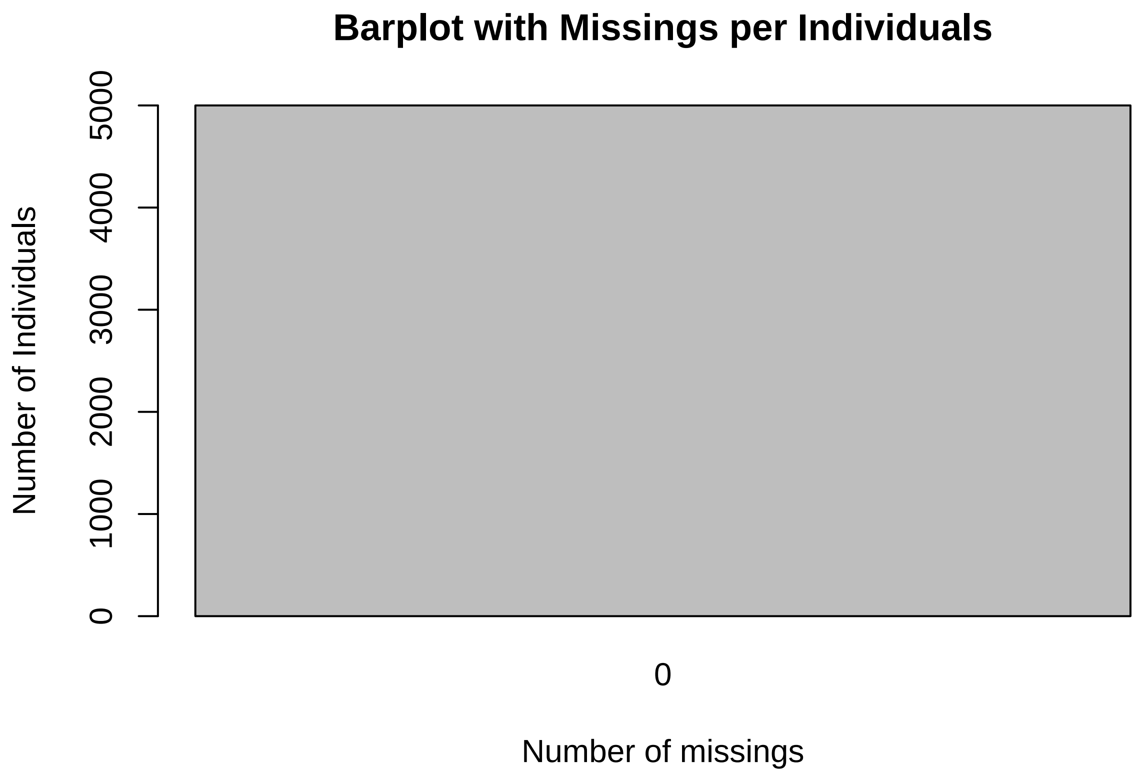
3.2 Per Individuals

3.2.1 Missings

```r
table(imis)
```

```
## imis
##    0
## 5000
```

```r
barplot(table(imis), main = "Barplot with Missings per Individuals",
        xlab = "Number of missings", ylab = "Number of Individuals",
        col = "grey",
        ylim = c(0,5000))
```

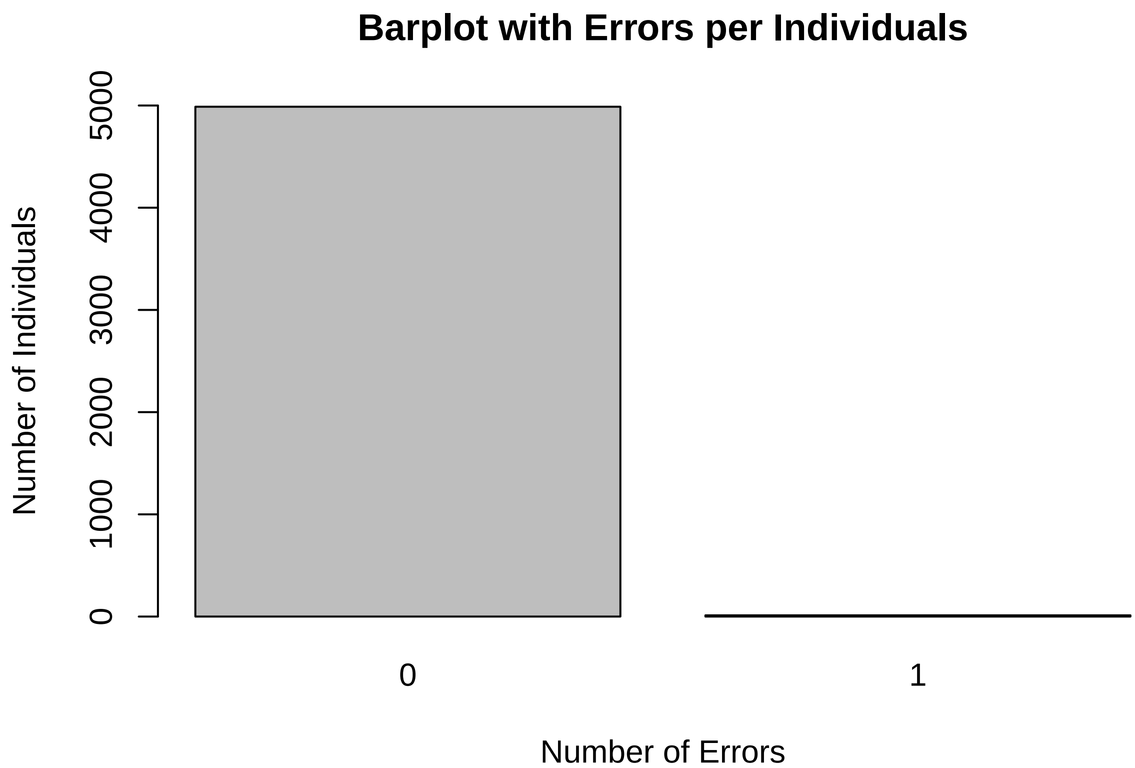


3.2.2 Errors

```
table(ierrs)
```

```
## ierrs
##    0    1
## 4988   12
```

```
barplot(table(ierrs), main = "Barplot with Errors per Individuals",
        xlab = "Number of Errors", ylab = "Number of Individuals",
        col = "grey",
        ylim = c(0,5000))
```

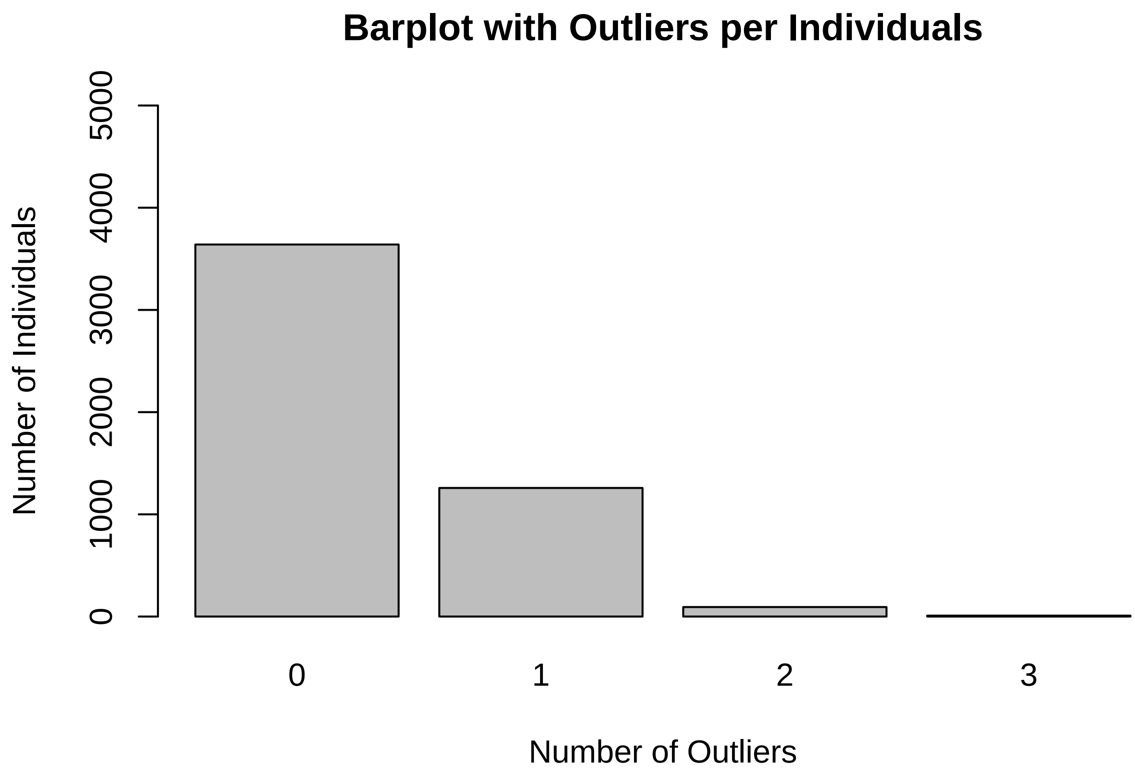


3.2.3 Outliers

```
table(iouts)
```

```
## iouts
##    0    1    2    3
## 3640 1258   93    9
```

```
barplot(table(iouts), main = "Barplot with Outliers per Individuals",
        xlab = "Number of Outliers", ylab = "Number of Individuals",
        col = "grey",
        ylim = c(0,5000))
```



Summary and totals of missings, errors and outliers:

```
# TOTAL OF INDIVIDUAL MISSINGS, ERRORS, OUTLIERS:
total_missings <- sum(imis); total_errors <- sum(ierrs); total_outliers <- sum(iouts);
total_missings; total_errors; total_outliers;
```

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

```
## [1] 12
```

```
## [1] 1471
```

3.3 Multivariant Outliers Detection

We are applying the Mahalanobis method to identify multivariate outliers

```
library(mvoutlier)
```

```
## Loading required package: sgeostat
```

```
#Subset of dataframe with numerical values, without rows that have NAs.

df_temp <- na.omit(df)
numerical_df <- df_temp[, sapply(df_temp, is.numeric)]

# Compute Mahalanobis Distance

mahalanobis_dist <- mahalanobis(numerical_df, colMeans(numerical_df), cov(numerical_df))
```

```r
# Identifying outliers using a threshold
outliers <- numerical_df[mahalanobis_dist > qchisq(0.95, df = 6), ]


# Print the outliers
print("Number of Multivariant Outliers:")
```

```
## [1] "Number of Multivariant Outliers:"
```

```r
length(outliers)
```

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

```r
print("Some Multivariant Outliers:")
```

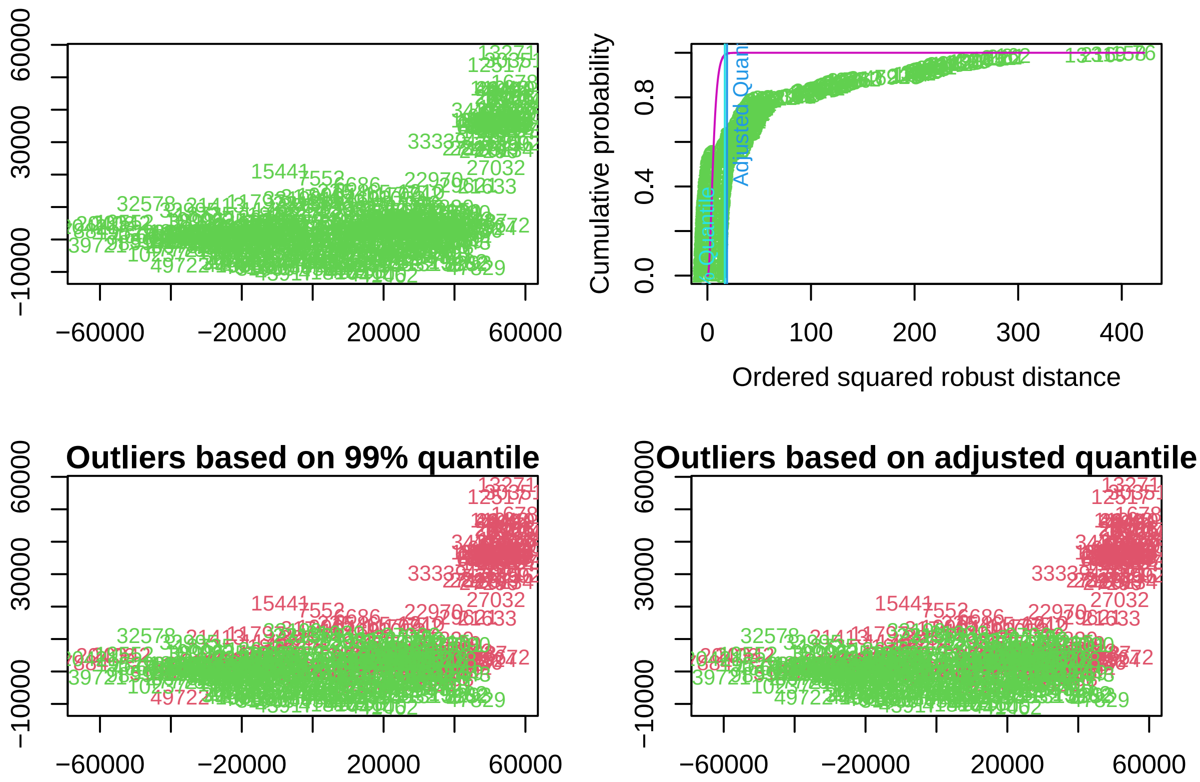
```
## [1] "Some Multivariant Outliers:"
```

```r
head(outliers)
```

```
##       year price mileage tax  mpg engineSize
## 12837 2017 19761   39681 200 39.8        3.0
## 1478  2017 15498   62140 145 64.2        2.0
## 44423 2012  6899   41515 145 47.9        1.4
## 9225  2016 17000   77700 125 58.9        2.0
## 3952  2015 24500   56000 200 47.9        3.0
## 10632 2012 10490   24693 165 51.4        2.0
```

```r
library(mvoutlier)
vout<-aq.plot(outliers, delta=qchisq(0.99, df= 6 ),alpha=0.01)
```

```
## Projection to the first and second robust principal components.
## Proportion of total variation (explained variance): 0.9989416
```



3.4 Correlation between variables

We observe a strong correlation between 'year' and 'mileage,' which is intuitively sensible since both increase as years pass and the vehicle is driven. Additionally, the 'price' variable shows noteworthy correlations with 'year' and 'engine size'.

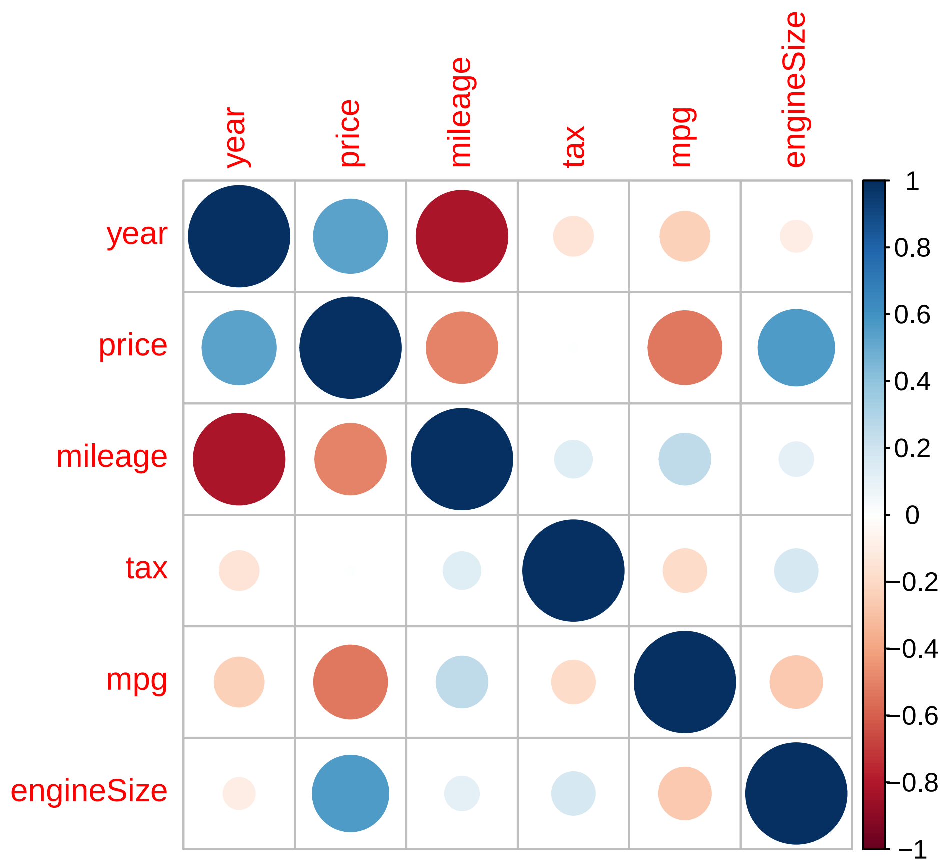
```
# dataset with numerical variables and individuals without NA values.

df_temp <- na.omit(df)
numerical_df <- df_temp[, sapply(df_temp, is.numeric)]
numerical_df <- numerical_df[1:6]
head(numerical_df)
```

```
##       year price mileage tax  mpg engineSize
## 12837 2017 19761   39681 200 39.8        3.0
## 29357 2018 44738   21276 150 36.7        3.0
## 47901 2019 19000   13191 145 44.1        2.0
## 37819 2019 17990    1201 145 57.7        1.6
## 25588 2016 25412   24346 160 51.4        3.0
## 22743 2019 16930    5317 145 49.6        1.6
```

```
# Coorelation matrix
correlation_matrix <- cor(numerical_df)

# Print the correlation matrix
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```r
corrplot(correlation_matrix)
```



Ranking the variables according to correlation:

```r
# Rank of variables by correlation with 'price'
correlations_with_price <- correlation_matrix['price', ]
sorted_correlations <- sort(correlations_with_price, decreasing = TRUE)
print("Variables Ranked by Correlation with Price:")
```

```
## [1] "Variables Ranked by Correlation with Price:"
```

```r
print(sorted_correlations)
```

```
##       price   engineSize         year          tax      mileage          mpg
##  1.000000000  0.565422861  0.531338779  0.007682155 -0.492879437 -0.526867644
```

```r
# Rank of variables by correlation with 'engineSize'
correlations_with_engineSize <- correlation_matrix['engineSize', ]
sorted_correlations <- sort(correlations_with_engineSize, decreasing = TRUE)
print("Variables Ranked by Correlation with engineSize:")
```

```
## [1] "Variables Ranked by Correlation with engineSize:"
```

```r
print(sorted_correlations)
```

```
## engineSize      price        tax    mileage       year        mpg
##  1.0000000  0.5654229  0.1793756  0.1120305 -0.0955428 -0.2640511
```

```r
# Rank of variables by correlation with 'tax'
correlations_with_tax <- correlation_matrix['tax', ]
sorted_correlations <- sort(correlations_with_tax, decreasing = TRUE)
print("Variables Ranked by Correlation with tax:")
```

```
## [1] "Variables Ranked by Correlation with tax:"
```

```r
print(sorted_correlations)
```

```
##          tax   engineSize      mileage        price         year          mpg
##  1.000000000  0.179375569  0.133762385  0.007682155 -0.149407334 -0.180481805
```

```r
# Rank of variables by correlation with 'mileage'
correlations_with_mileage <- correlation_matrix['mileage', ]
sorted_correlations <- sort(correlations_with_mileage, decreasing = TRUE)
print("Variables Ranked by Correlation with mileage:")
```

```
## [1] "Variables Ranked by Correlation with mileage:"
```

```r
print(sorted_correlations)
```

```
##    mileage        mpg        tax engineSize      price       year
##  1.0000000  0.2556323  0.1337624  0.1120305 -0.4928794 -0.8159894
```

```r
# Rank of variables by correlation with 'mpg'
correlations_with_mpg <- correlation_matrix['mpg', ]
sorted_correlations <- sort(correlations_with_mpg, decreasing = TRUE)
print("Variables Ranked by Correlation with MPG:")
```

```
## [1] "Variables Ranked by Correlation with MPG:"
```

```r
print(sorted_correlations)
```

```
##        mpg    mileage        tax       year engineSize      price
##  1.0000000  0.2556323 -0.1804818 -0.2374643 -0.2640511 -0.5268676
```

```r
# Rank of variables by correlation with 'Year'
correlations_with_year <- correlation_matrix['year', ]
sorted_correlations <- sort(correlations_with_year, decreasing = TRUE)
print("Variables Ranked by Correlation with Year:")
```

```
## [1] "Variables Ranked by Correlation with Year:"
```

```r
print(sorted_correlations)
```

```
##       year      price engineSize        tax        mpg    mileage
##  1.0000000  0.5313388 -0.0955428 -0.1494073 -0.2374643 -0.8159894
```

4. Imputation

We will refrain from applying imputation to any missing values in the "price" variable. This variable represents the target variable in our study, and altering or filling in missing values in this variable could introduce bias into our data, potentially skewing the results.

Note: in this case of ours we have no missings at all.

4. 1 Imputation with Numerical Variables

As we can see, missing values are substituted with new values:

```r
quantitative_vars<-names(df)[c(2,3,5,7:9)]

summary(df[,quantitative_vars])
```

```
##       year          price           mileage             tax
##  Min.   :2008   Min.   :   899   Min.   :     1   Min.   :125.0
##  1st Qu.:2016   1st Qu.: 13994   1st Qu.:  5836   1st Qu.:145.0
##  Median :2017   Median : 19500   Median : 16513   Median :145.0
##  Mean   :2017   Mean   : 21573   Mean   : 22834   Mean   :146.9
##  3rd Qu.:2019   3rd Qu.: 26499   3rd Qu.: 33396   3rd Qu.:145.0
##  Max.   :2020   Max.   :154998   Max.   :116000   Max.   :200.0
##  NA's   :29                      NA's   :20       NA's   :1270
##       mpg          engineSize
##  Min.   : 21.10   Min.   :0.6
##  1st Qu.: 44.10   1st Qu.:1.5
##  Median : 52.30   Median :2.0
##  Mean   : 52.51   Mean   :1.9
##  3rd Qu.: 61.40   3rd Qu.:2.0
##  Max.   :113.00   Max.   :3.2
##  NA's   :49       NA's   :78
```

```r
res.input<-imputePCA(df[,quantitative_vars],ncp=5)

summary(res.input$completeObs)
```

```
##       year          price           mileage             tax
##  Min.   :2008   Min.   :   899   Min.   :     1   Min.   :125.0
##  1st Qu.:2016   1st Qu.: 13994   1st Qu.:  5866   1st Qu.:145.0
##  Median :2017   Median : 19500   Median : 16698   Median :145.0
##  Mean   :2017   Mean   : 21573   Mean   : 22977   Mean   :146.9
##  3rd Qu.:2019   3rd Qu.: 26499   3rd Qu.: 33646   3rd Qu.:147.2
##  Max.   :2020   Max.   :154998   Max.   :116000   Max.   :200.0
##       mpg          engineSize
##  Min.   : 21.10   Min.   :0.600
##  1st Qu.: 44.10   1st Qu.:1.500
##  Median : 52.30   Median :2.000
##  Mean   : 52.51   Mean   :1.923
##  3rd Qu.: 60.20   3rd Qu.:2.000
##  Max.   :113.00   Max.   :8.051
```

```r
df[,"year"] <- res.input$completeObs[,"year"]

df[,"price"] <- res.input$completeObs[,"price"]

df[,"mileage"] <- res.input$completeObs[,"mileage"]

df[,"tax"] <- res.input$completeObs[,"tax"]

df[,"mpg"] <- res.input$completeObs[,"mpg"]

df[,"engineSize"] <- res.input$completeObs[,"engineSize"]
```

4.2 Imputation to factors (Categorical Variables)

```r
categorical_vars<-names(df)[c(1,4,6,10)]
summary(df[,categorical_vars])
```

```
##                  model                   transmission      fuelType
##  VW- Golf          : 510   f.Trans-Manual   :1798   Diesel  :2825
##  Mercedes- C Class: 387   f.Trans-SemiAuto :1870   Electric:  15
##  VW- Polo          : 328   f.Trans-Automatic:1332   Hybrid  :  64
##  Mercedes- A Class: 253                            Petrol  :2096
##  BMW- 3 Series     : 237
##  BMW- 1 Series     : 219
##  (Other)           :3066
##    manufacturer
##  Audi    :1057
##  BMW     :1057
##  Mercedes:1337
##  VW      :1549
##
##
##
```

```r
#nb <- estim_ncpMCA(df[, categorical_vars],ncp.max=25) #it stabilizes at ncp = 7

res.input<-imputeMCA(df[,categorical_vars],ncp=7)
summary(res.input$completeObs)
```

```
##                  model                   transmission      fuelType
##  VW- Golf          : 510   f.Trans-Manual   :1798   Diesel  :2825
##  Mercedes- C Class: 387   f.Trans-SemiAuto :1870   Electric:  15
##  VW- Polo          : 328   f.Trans-Automatic:1332   Hybrid  :  64
##  Mercedes- A Class: 253                            Petrol  :2096
##  BMW- 3 Series     : 237
##  BMW- 1 Series     : 219
##  (Other)           :3066
##    manufacturer
##  Audi    :1057
##  BMW     :1057
##  Mercedes:1337
##  VW      :1549
##
##
##
```

```r
df[,"model"] <- res.input$completeObs[,"model"]
df[,"transmission"] <- res.input$completeObs[,"transmission"]
df[,"fuelType"] <- res.input$completeObs[,"fuelType"]
df[,"manufacturer"] <- res.input$completeObs[,"manufacturer"]
```

5. Discretization

Discretization can be important for profiling as it enhances data interpretability, reduces noise, and making the profiling process more effective and more understandable.

```r
# f.Year :
table(df$year, useNA="always")
```

```
##
##             2008             2009             2010 2010.34738403483
##                8               10               14                1
## 2010.66773380449 2010.78983258164 2010.97377044245             2011
##                1                1                1               20
##  2011.1217842681 2011.22075865324 2011.29089233062 2011.35562454868
##                1                1                1                1
## 2011.91223446802             2012 2012.54949928848 2012.73973497351
##                1               38                1                1
## 2012.79396662919 2012.82148845673 2012.85527406511 2012.96681826786
##                1                1                1                1
##             2013 2013.13269141168 2013.22232815254 2013.32869020097
##              127                1                1                1
## 2013.49526927193 2013.54228455266 2013.61323177794 2013.82692418114
##                1                1                1                1
## 2013.90270400965 2013.93149957525             2014 2014.03047094366
##                1                1              177                1
##             2015 2015.18898465221             2016 2016.27386794821
##              440                1              841                1
## 2016.33089399059 2016.40530896584             2017             2018
##                1                1              881              479
##             2019             2020             <NA>
##             1607              329                0
```

```r
quantile(df$year,seq(0,1,0.25))
```

```
##   0%  25%  50%  75% 100%
## 2008 2016 2017 2019 2020
```

```r
min(df$year)
```

```
## [1] 2008
```

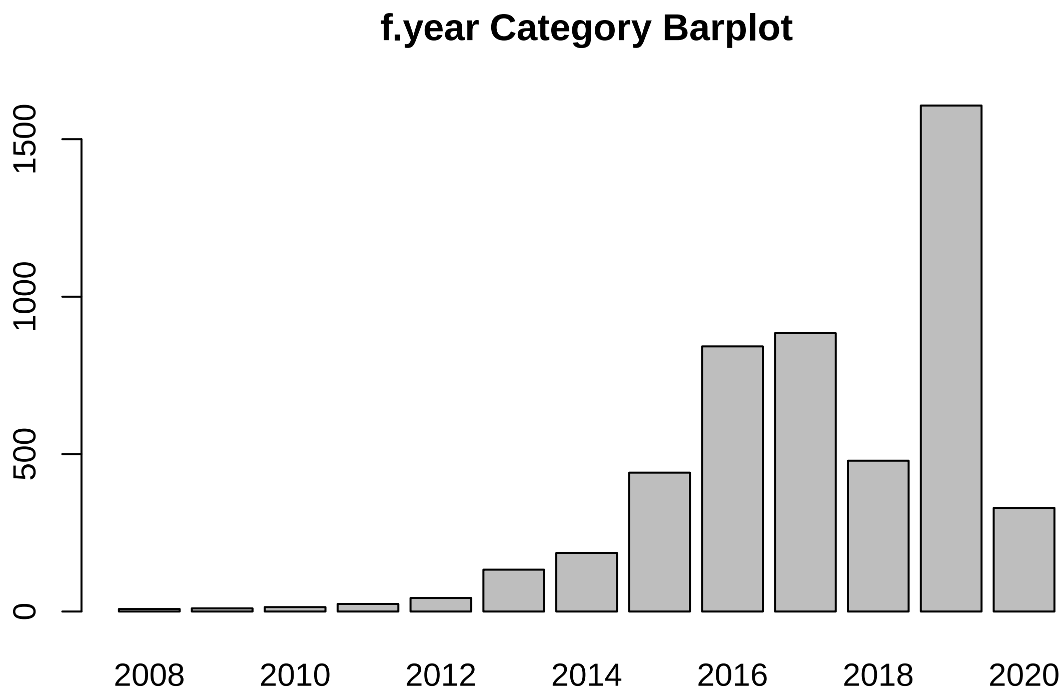
```r
year_labels <- as.character(seq(2008, 2020))
year_breaks <- seq(2007, 2020)
df$f.year <- cut(df$year, breaks = year_breaks, labels = year_labels, include.lowest = TRUE)

summary(df$f.year)
```

```
## 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020
##    8   10   14   24   43  133  186  441  842  884  479 1607  329
```

```r
table(df$f.year, useNA="always")
```

```
##
## 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 <NA>
##    8   10   14   24   43  133  186  441  842  884  479 1607  329    0
```

```r
barplot(summary(df$f.year),main="f.year Category Barplot",col = "Grey")
```




```
# f.Price:
summary(df$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     899   13994   19500   21573   26499  154998
```

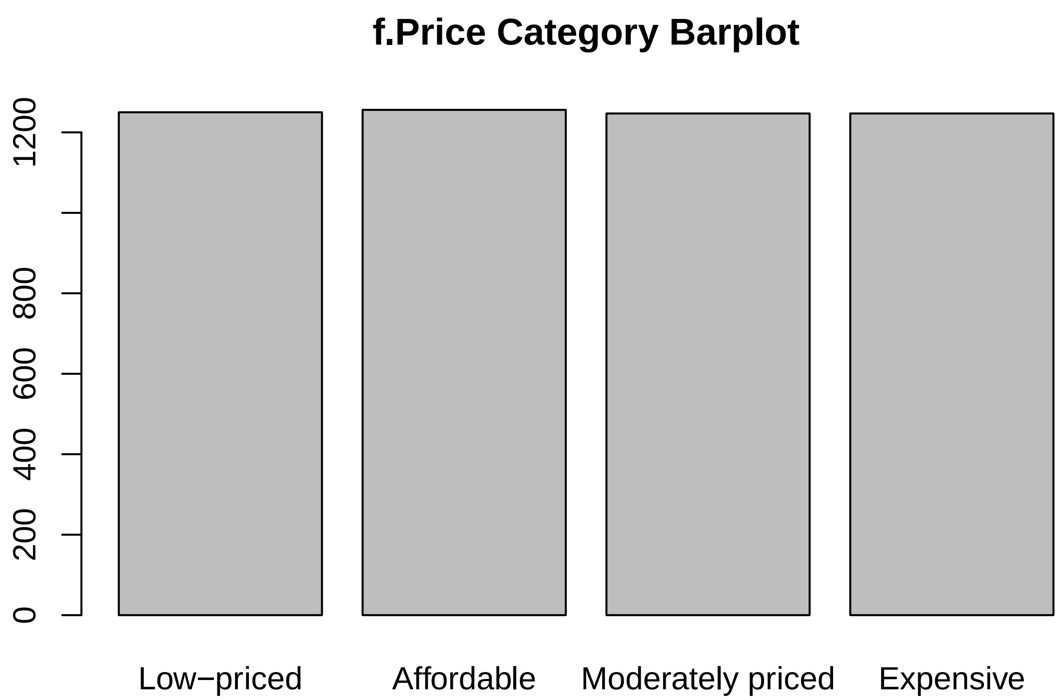
```
quantile(df$price,seq(0,1,0.25),na.rm=TRUE)
```

```
##       0%      25%      50%      75%     100%
##    899.0  13994.5  19500.0  26499.0 154998.0
```

```
df$f.price <- cut(df$price, breaks = c(min(df$price), 13994.5   , 19500    , 26499.0 , max(df$price)), la
table(df$f.price)
```

```
##
##        Low-priced        Affordable Moderately priced         Expensive
##              1250              1256              1247              1247
```

```
barplot(summary(df$f.price),main="f.Price Category Barplot",col = "Grey")
```




```
# f.Mileage: Usage.
summary(df$mileage)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1    5866   16698   22977   33646  116000
```

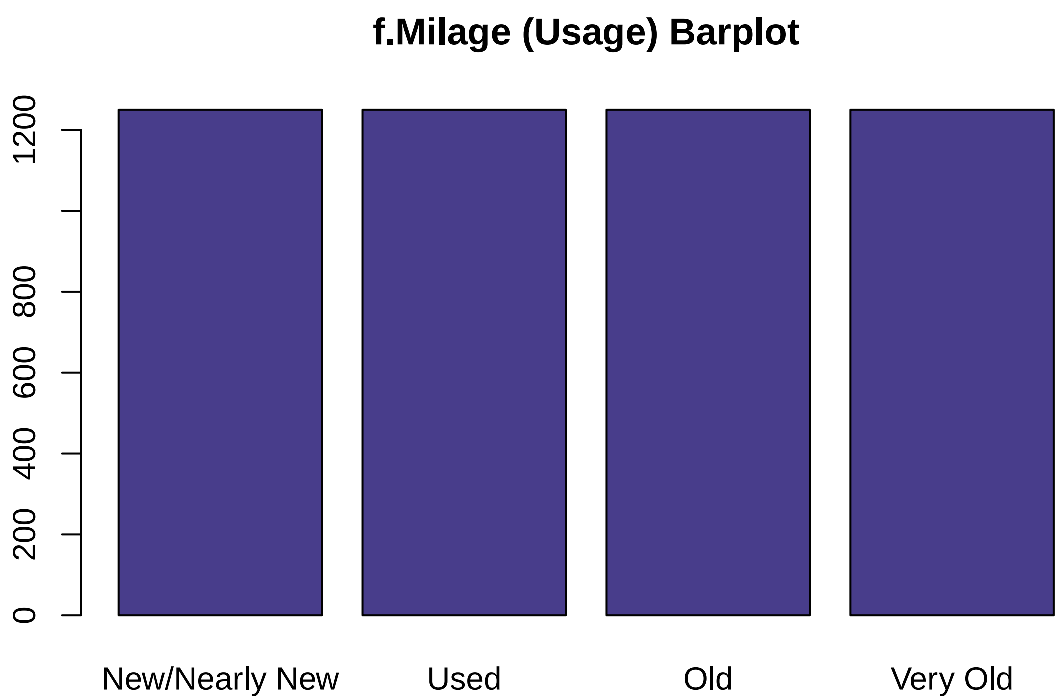
```
quantile(df$mileage,seq(0,1,0.25),na.rm=TRUE)
```

```
##       0%      25%      50%      75%     100%
##      1.0   5866.5  16697.5  33645.5 116000.0
```

```
mileage_labels <- c("New/Nearly New", "Used", "Old", "Very Old")
mileage_intervals <- c(min(df$mileage), 5866.5  , 16697.5, 33645.5, max(df$mileage))
df$f.miles <- cut(df$mileage, breaks = mileage_intervals, labels = mileage_labels, include.lowest = TRUE
table(df$f.miles)
```

```
##
## New/Nearly New           Used            Old       Very Old
##           1250           1250           1250           1250
```

```
barplot(summary(df$f.miles),main="f.Milage (Usage) Barplot",col = "DarkSlateBlue")
```




```
table(df$f.miles,useNA="always")
```

```
##
## New/Nearly New           Used            Old       Very Old           <NA>
##           1250           1250           1250           1250              0
```

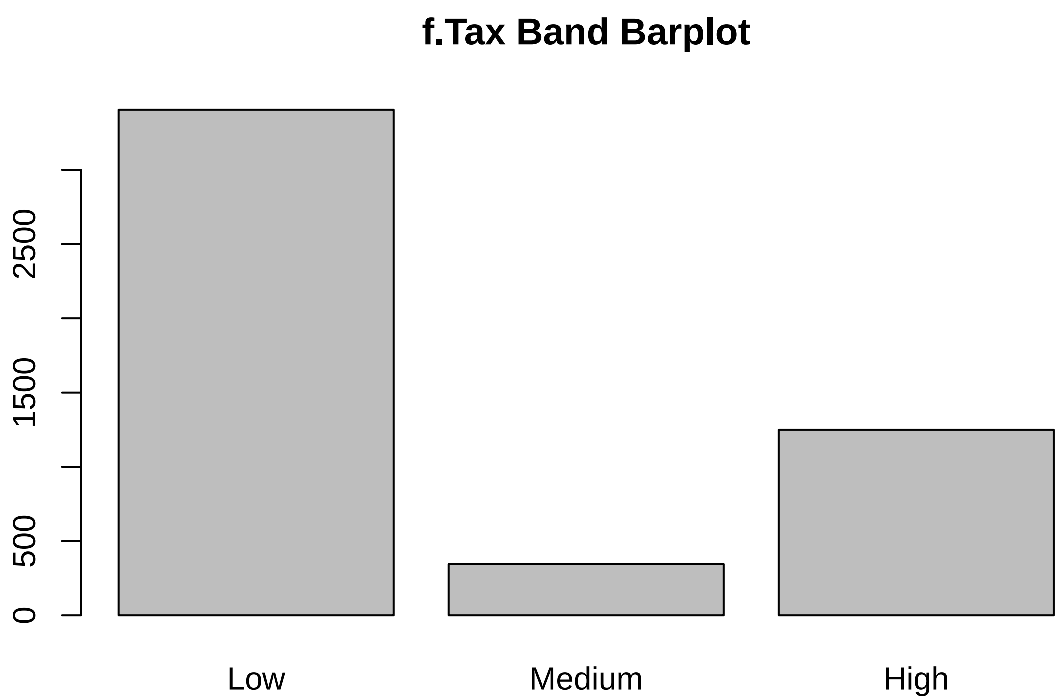
```
# f.Tax:
summary(df$tax)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   125.0   145.0   145.0   146.9   147.2   200.0
```

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

```
##     0%    25%    50%    75%   100%
## 125.00 145.00 145.00 147.19 200.00
```

```
tax_labels <- c("Low", "Medium", "High")
tax_intervals <- c(min(df$tax), 145, 147.19  , max(df$tax))
df$f.tax <- cut(df$tax, breaks = tax_intervals, labels = tax_labels, include.lowest = TRUE)
barplot(summary(df$f.tax),main="f.Tax Band Barplot",col = "Grey")
```




```
# MPG Category: Consumption Category
summary(df$mpg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   21.10   44.10   52.30   52.51   60.20  113.00
```

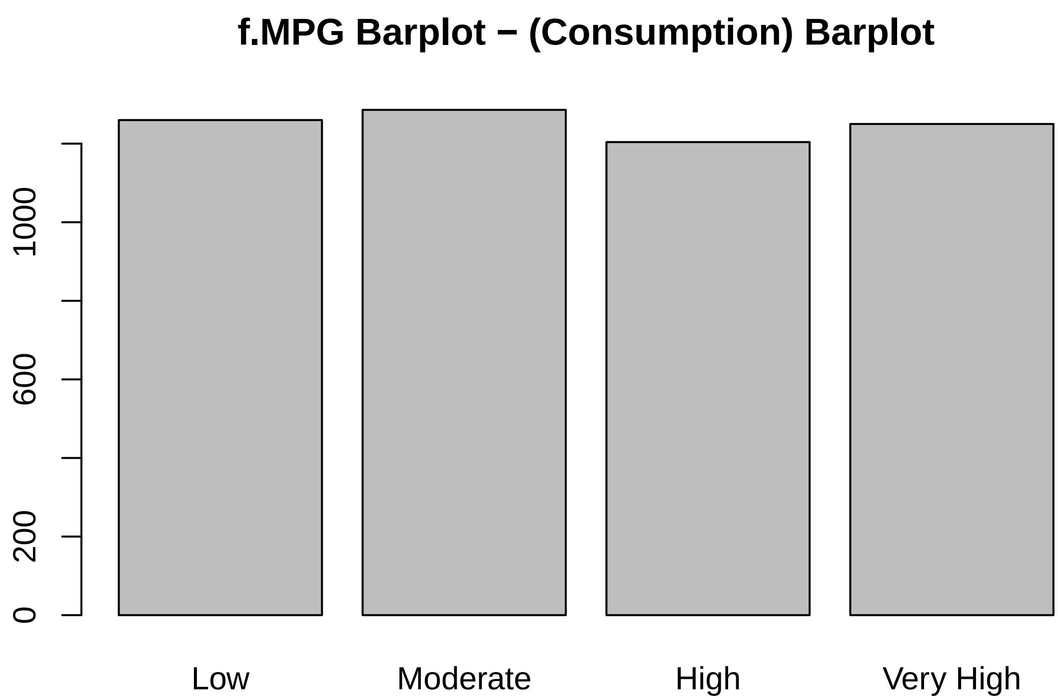
```
quantile(df$mpg,seq(0,1,0.25),na.rm=TRUE)
```

```
##        0%       25%       50%       75%      100%
##  21.10000  44.10000  52.30000  60.19753 113.00000
```

```
mpg_labels <- c("Low", "Moderate", "High", "Very High")
mpg_intervals <- c(min(df$mpg), 44.10, 52.30, 60.20, max(df$mpg))
df$f.mpg <- cut(df$mpg, breaks = mpg_intervals, labels = mpg_labels, include.lowest = TRUE)

table(df$f.mpg)
```

```
##
##       Low  Moderate      High Very High
##      1260      1286      1204      1250
```

```
barplot(summary(df$f.mpg),main="f.MPG Barplot - (Consumption) Barplot",col = "Grey")
```



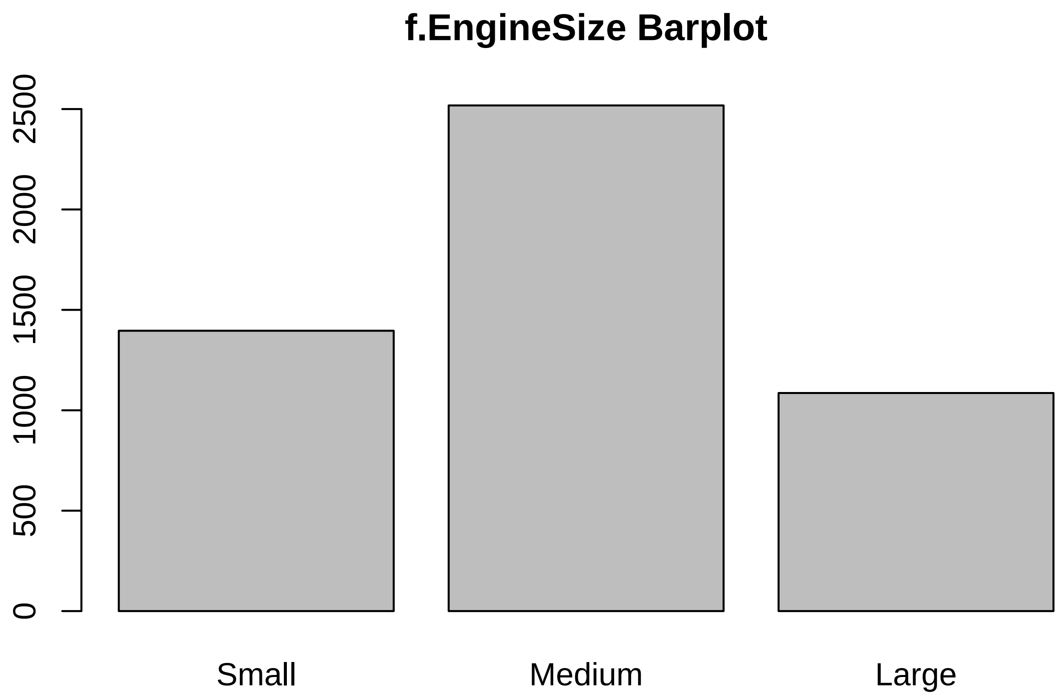

```r
# Engine Size Category: Small, Medium, Large
summary(df$engineSize)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.600   1.500   2.000   1.923   2.000   8.051
```

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

```
##       0%      25%      50%      75%     100%
## 0.600000 1.500000 2.000000 2.000000 8.050534
```

```r
engineSize_labels <- c("Small", "Medium", "Large")
engineSize_intervals <- c(min(df$engineSize), 1.5, 2.0, max(df$engineSize))
df$f.engineSize <- cut(df$engineSize, breaks = engineSize_intervals, labels = engineSize_labels, includ
barplot(summary(df$f.engineSize),main="f.EngineSize Barplot",col = "Grey")
```



6. Profiling

```
library(FactoMineR)
summary(df$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     899   13994   19500   21573   26499  154998
```

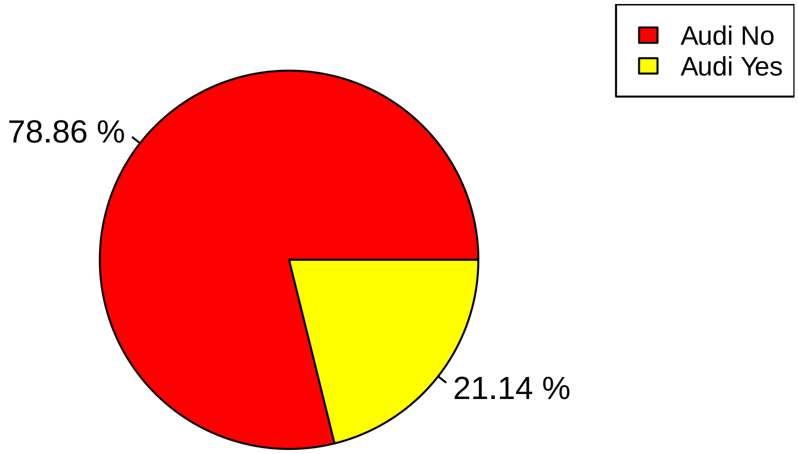
```
# Binary Target: Audi?
df$Audi<-ifelse(df$manufacturer == "Audi",1,0)
df$Audi<-factor(df$Audi,labels=paste("Audi",c("No","Yes")))
summary(df$Audi)
```

```
##  Audi No Audi Yes
##     3943     1057
```

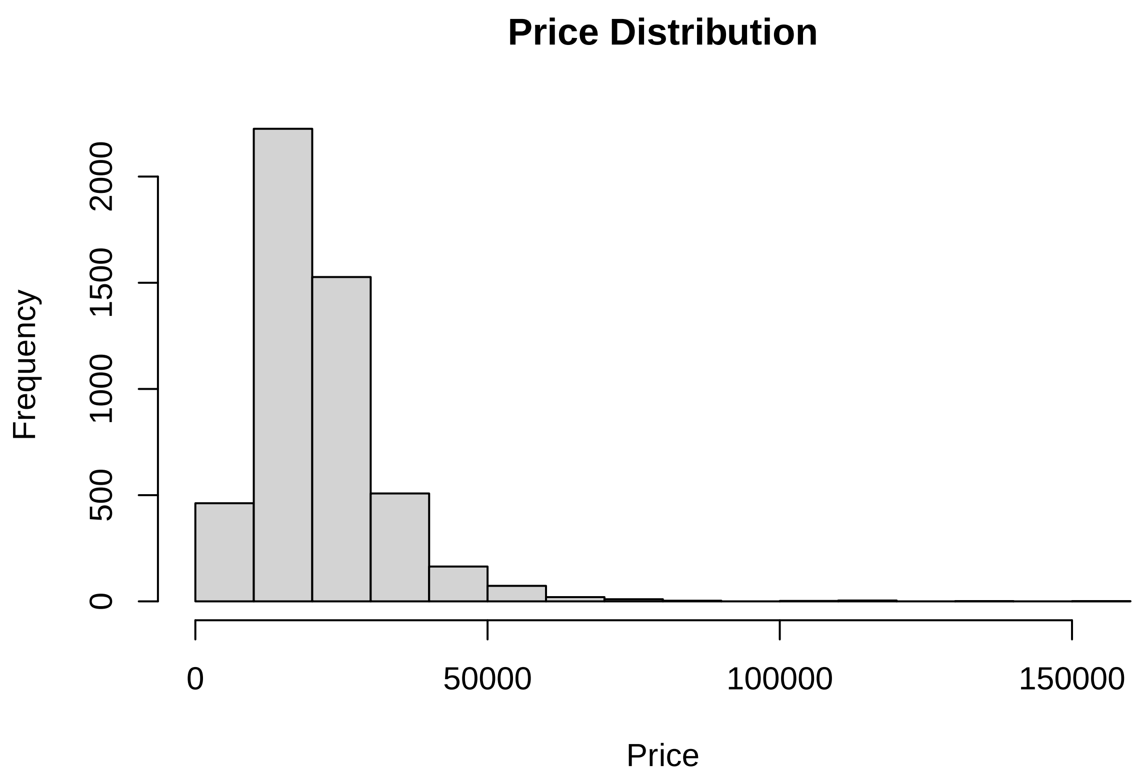
```
# Pie
piepercent<-round(100*(table(df$Audi)/nrow(df)),dig=2); piepercent
```

```
##
##  Audi No Audi Yes
##    78.86    21.14
```

```
pie(table(df$Audi),col=heat.colors(2),labels=paste(piepercent,"%"))
legend("topright", levels(df$Audi), cex = 0.8, fill = heat.colors(2))
```




```
# Histogram for Price
hist(df$price, main = "Price Distribution", xlab = "Price")
```



With Numeric Target "Price":

Clearly, each quantitative variable is correlated to "price," either positively or negatively.

In simple terms, when the year and engine specifications go up, the price tends to rise. On the other hand, an increase in mileage and mpg typically leads to a decrease in price. This straightforward relationship helps us understand how these factors impact pricing.

```
res.condes<- condes(df, 3)

res.condes$quanti
```

```
##            correlation p.value
## engineSize   0.6417973       0
## year         0.5625867       0
## mileage     -0.5160380       0
## mpg         -0.5809686       0
```

In this context, it's evident that the price significantly influences the choice of car category. As the price increases, certain car models become increasingly likely choices compared to others. The same thing happens with the type of transmission.

```
res.condes$quali
```

```
##                      R2      p.value
## model       0.520829965 0.000000e+00
## f.year      0.340415103 0.000000e+00
## f.price     0.697084268 0.000000e+00
## f.miles     0.296621674 0.000000e+00
```

```
## f.mpg        0.306345595  0.000000e+00
## transmission 0.220025494 2.306211e-270
## f.engineSize 0.179779108 9.040068e-216
## manufacturer 0.080505068  1.417320e-90
## f.tax        0.059143057  7.058305e-67
## fuelType     0.007073811  9.692687e-08
## Audi         0.003975412  8.131402e-06
```

There is a lot of information to deduce from this output:

- The price is much likely higher if it's from 2020 year, and if the MPG is categorized as Low, and the engineSize is Large, if the car is New/Likely New (based on mileage discretization),
- The most expensive cars are: BMW- 8 Series, Audi- R8, VW- California, Audi- Q8, BMW- X6…
- Usually cars that are classed as hybrid tend to be more expensive.
- We can also check the cheapest car models that usually are manual transmission and categorized as affordable.

```
df_cat <- as.data.frame(res.condes$category)
df_cat[order(df_cat$Estimate, decreasing = TRUE),]
```

```
##                                     Estimate       p.value
## model=Mercedes- G Class        124434.538822  3.354380e-32
## model=Audi- R8                  72157.788822  3.749753e-47
## model=BMW- M5                   40365.872156  4.734815e-14
## model=BMW- X7                   39468.138822  1.024919e-21
## model=Audi- RS4                 36436.538822  6.410382e-05
## model=Audi- Q8                  31282.538822  1.930085e-15
## model=BMW- 8 Series             30420.205489  1.813806e-09
## model=VW- California            27427.538822  5.807168e-06
## model=BMW- X6                   23078.253108  7.146147e-14
## model=Audi- Q7                  18860.026002  1.979555e-54
## model=Mercedes- GLS Class       17731.605489  5.676987e-20
## f.year=2020                     17666.424269  3.823079e-70
## model=Audi- RS5                 17331.538822  2.061904e-02
## model=Audi- RS6                 17282.538822  3.723359e-06
## model=BMW- M2                   15184.038822  2.633929e-03
## f.price=Expensive               14700.291377  0.000000e+00
## f.year=2019                     13933.236973 8.808905e-206
## model=BMW- 7 Series             10740.205489  1.670965e-09
## f.mpg=Low                       10368.681899  0.000000e+00
## model=BMW- M4                   10144.824537  9.300966e-15
## model=VW- Caravelle              9841.824537  5.119488e-10
## model=BMW- X5                    9517.372156  1.754686e-26
## model=Audi- A8                   8823.872156  5.335379e-08
## model=Mercedes- S Class          8702.712735  6.479387e-14
## f.miles=New/Nearly New           8657.176600 2.879661e-235
## model=Mercedes- GLE Class        7433.557690  2.216762e-26
## f.engineSize=Large               7420.722195 5.050978e-168
## fuelType=Hybrid                  5472.170676  2.083920e-05
## f.tax=Low                        4544.683844  1.435068e-48
## model=Mercedes- SL CLASS         4526.238822  6.060965e-11
## transmission=f.Trans-SemiAuto    4511.282762 4.575840e-119
## f.year=2017                      4453.911485  2.058227e-14
## model=BMW- X4                    3766.129732  1.297516e-07
```

```
## model=VW- Touareg               3558.729299  6.077266e-13
## manufacturer=Mercedes           3034.163366  6.291241e-37
## transmission=f.Trans-Automatic  2735.801771  2.837568e-28
## f.miles=Used                    2579.037400  1.456895e-20
## model=Mercedes- GLC Class       1765.434475  6.106381e-25
## f.year=2016                     1446.487565  9.289086e-58
## manufacturer=Audi               1065.774262  8.131402e-06
## f.price=Moderately priced       1044.051601  1.459049e-04
## f.tax=High                       928.035058  3.502435e-13
## Audi=Audi Yes                    878.066828  8.131402e-06
## manufacturer=BMW                 869.837649  1.284345e-04
## model=BMW- Z4                    636.253108  2.500517e-02
## model=Audi- Q5                   554.635910  5.731692e-18
## f.year=2015                       -9.832546  8.406766e-44
## fuelType=Diesel                  -45.421061  5.239279e-04
## Audi=Audi No                    -878.066828  8.131402e-06
## f.engineSize=Medium            -1090.556974  1.569369e-02
## fuelType=Petrol                -1352.180708  1.550615e-05
## model=BMW- X2                  -1426.169511  1.084891e-03
## f.year=2014                    -1931.137534  1.200559e-28
## model=BMW- X3                  -2101.746892  2.005844e-05
## model=Mercedes- X-CLASS        -2975.261178  4.019808e-02
## f.miles=Old                    -3529.769000  2.359242e-37
## model=Mercedes- V Class        -3718.911178  3.777450e-02
## f.year=2013                    -4246.068612  7.918918e-32
## model=Mercedes- CLS Class      -4327.127844  4.402867e-02
## f.mpg=High                     -4409.728455  1.645703e-56
## f.price=Affordable             -4718.097655  3.525089e-66
## manufacturer=VW                -4969.775277  8.411264e-87
## f.tax=Medium                   -5472.718902  5.116203e-47
## f.mpg=Very High                -5590.278196  2.796377e-95
## model=Mercedes- E Class        -5887.769749  2.367795e-04
## f.engineSize=Small             -6330.165221 3.123262e-110
## model=Mercedes- C Class        -6660.649808  2.677191e-05
## f.year=2010                    -6672.685153  5.584784e-06
## f.year=2012                    -6676.552263  1.223403e-15
## f.year=2011                    -6749.679201  2.097540e-09
## transmission=f.Trans-Manual    -7247.084533 9.069704e-267
## f.miles=Very Old               -7706.445000 1.411152e-182
## f.year=2009                    -8461.570868  1.459745e-05
## f.year=2008                   -10655.220868  9.740801e-06
## model=BMW- 3 Series           -10798.469616  1.216820e-02
## f.price=Low-priced            -11026.245323  0.000000e+00
## model=BMW- 2 Series           -11574.095960  1.371921e-02
## model=BMW- X1                 -11769.015024  4.744161e-02
## model=Audi- A3                -13437.482011  3.164646e-08
## model=VW- Golf                -13696.929805  3.852614e-23
## model=BMW- 1 Series           -14278.082182  1.767152e-12
## model=VW- Passat              -14705.281178  3.751870e-07
## model=Audi- A1                -15944.337090  3.484825e-13
## model=VW- Golf SV             -15963.889749  4.860002e-03
## model=VW- Scirocco            -17626.275992  7.544347e-05
## model=VW- Polo                -19101.698982  5.192534e-64
## model=Mercedes- SLK           -19683.818320  4.244729e-04
```

```
## model=VW- CC                        -20556.016733  2.256001e-03
## model=VW- Beetle                    -22533.127844  3.587711e-05
## model=VW- Up                        -22555.966672  6.750578e-31
```

Profiling binary factor "Audi?" it with all other variables:

```r
res.catdes <- catdes(df,17,proba = 0.05)
```

We observe a relatively weak correlation between Y.bin-'Audi' and the other quantitative variables. However, the presence of very low p-values suggests that there is a connection. It's important to note that while this connection exists, the limited sample size may prevent us from establishing it.

```r
res.catdes$quanti.var
```

```
##               Eta2      P-value
## mpg   0.0092478291 9.489946e-12
## price 0.0039754125 8.131402e-06
## tax   0.0019457989 1.809295e-03
## year  0.0007909104 4.675624e-02
```

Again, we can deduce plenty of information:

- A robust link emerges between this binary variable and the categories. Notably, Audi cars are distinctly associated with the 'Medium Size' engines, 'Low' mpg ratings, and the 'Expensive' category. Furthermore, they tend to favor manual transmission and 'Petrol' as their preferred fuel type.

```r
res.catdes$category
```

```
## $`Audi No`
##                                  Cla/Mod    Mod/Cla Global       p.value
## manufacturer=VW                100.00000 39.2848085  30.98 2.226616e-197
## manufacturer=Mercedes          100.00000 33.9081917  26.74 6.647141e-165
## manufacturer=BMW               100.00000 26.8069997  21.14 2.760168e-125
## model=VW- Golf                 100.00000 12.9343140  10.20  1.316978e-56
## model=Mercedes- C Class        100.00000  9.8148618   7.74  1.720220e-42
## model=VW- Polo                 100.00000  8.3185392   6.56  7.158404e-36
## model=Mercedes- A Class        100.00000  6.4164342   5.06  1.357959e-27
## model=BMW- 3 Series            100.00000  6.0106518   4.74  7.579104e-26
## model=BMW- 1 Series            100.00000  5.5541466   4.38  6.866524e-24
## model=VW- Tiguan               100.00000  4.8947502   3.86  4.456163e-21
## model=Mercedes- E Class        100.00000  4.4382450   3.50  3.852925e-19
## f.engineSize=Large              87.93738 24.2201370  21.72  3.986209e-18
## model=Mercedes- GLC Class      100.00000  2.9165610   2.30  9.620669e-13
## model=BMW- 2 Series            100.00000  2.9165610   2.30  9.620669e-13
## model=BMW- 5 Series            100.00000  2.7643926   2.18  4.149981e-12
## model=VW- Passat               100.00000  2.5361400   2.00  3.703873e-11
## model=VW- Up                   100.00000  2.3078874   1.82  3.290728e-10
## model=BMW- 4 Series            100.00000  2.1557190   1.70  1.408067e-09
## f.mpg=High                      84.88372 25.9193507  24.08  1.618647e-09
## model=Mercedes- GLA Class      100.00000  2.0542734   1.62  3.707033e-09
## model=VW- T-Roc                100.00000  1.7499366   1.38  6.728526e-08
## model=BMW- X1                  100.00000  1.6484910   1.30  1.765149e-07
## model=Mercedes- GLE Class      100.00000  1.3441542   1.06  3.170051e-06
## model=Mercedes- CL Class       100.00000  1.2934314   1.02  5.126008e-06
## model=Mercedes- B Class        100.00000  1.2934314   1.02  5.126008e-06
## model=BMW- X3                  100.00000  1.2427086   0.98  8.286995e-06
## model=VW- Touareg              100.00000  1.0651788   0.84  4.444368e-05
## model=BMW- X5                  100.00000  1.0651788   0.84  4.444368e-05
```

```
## f.price=Low-priced            82.72000 26.2236875  25.00  8.995141e-05
## f.mpg=Very High               82.56000 26.1729647  25.00  1.762084e-04
## model=Mercedes- SL CLASS     100.00000  0.7608420   0.60  7.861929e-04
## fuelType=Hybrid               93.75000  1.5216840   1.28  1.232824e-03
## model=VW- T-Cross            100.00000  0.7101192   0.56  1.268094e-03
## fuelType=Diesel               80.49558 57.6718235  56.50  1.283879e-03
## model=VW- Scirocco           100.00000  0.6847578   0.54  1.610376e-03
## model=VW- Touran             100.00000  0.6086736   0.48  3.296956e-03
## model=VW- Sharan             100.00000  0.6086736   0.48  3.296956e-03
## model=Mercedes- CLS Class    100.00000  0.6086736   0.48  3.296956e-03
## model=BMW- X2                100.00000  0.6086736   0.48  3.296956e-03
## model=Mercedes- S Class      100.00000  0.5833122   0.46  4.185957e-03
## transmission=f.Trans-SemiAuto  80.96257 38.3971595  37.40  4.714115e-03
## model=VW- Arteon             100.00000  0.5579508   0.44  5.314382e-03
## model=BMW- X4                100.00000  0.5579508   0.44  5.314382e-03
## model=VW- Golf SV            100.00000  0.5325894   0.42  6.746637e-03
## model=BMW- M4                100.00000  0.5325894   0.42  6.746637e-03
## model=Mercedes- V Class      100.00000  0.5072280   0.40  8.564428e-03
## f.miles=Used                  81.44000 25.8179051  25.00  9.294643e-03
## model=BMW- 6 Series          100.00000  0.4311438   0.34  1.751421e-02
## fuelType=Electric            100.00000  0.3804210   0.30  2.820989e-02
## model=Mercedes- X-CLASS      100.00000  0.3804210   0.30  2.820989e-02
## model=Mercedes- GLS Class    100.00000  0.3804210   0.30  2.820989e-02
## model=Mercedes- GL Class     100.00000  0.3804210   0.30  2.820989e-02
## model=VW- Caravelle          100.00000  0.3550596   0.28  3.579906e-02
## model=Mercedes- SLK          100.00000  0.3550596   0.28  3.579906e-02
## model=Audi- RS3                0.00000  0.0000000   0.04  4.465661e-02
## model=Audi- SQ5                0.00000  0.0000000   0.06  9.426316e-03
## model=Audi- S3                 0.00000  0.0000000   0.06  9.426316e-03
## model=Audi- RS6                0.00000  0.0000000   0.08  1.988260e-03
## model=Audi- R8                 0.00000  0.0000000   0.08  1.988260e-03
## model=Audi- Q8                 0.00000  0.0000000   0.10  4.190629e-04
## f.price=Expensive             75.06014 23.7382704  24.94  1.811609e-04
## fuelType=Petrol               76.04962 40.4260715  41.92  3.815371e-05
## transmission=f.Trans-Manual   75.63960 34.4915039  35.96  3.342572e-05
## model=Audi- A7                 0.00000  0.0000000   0.22  3.616280e-08
## model=Audi- A8                 0.00000  0.0000000   0.24  7.581939e-09
## f.engineSize=Medium           74.90071 47.8316003  50.36  4.491623e-12
## f.mpg=Low                     70.87302 22.6477302  25.20  4.959163e-15
## model=Audi- TT                 0.00000  0.0000000   0.68  7.403402e-24
## model=Audi- Q7                 0.00000  0.0000000   0.78  2.725205e-27
## model=Audi- A5                 0.00000  0.0000000   1.30  2.758792e-45
## model=Audi- Q2                 0.00000  0.0000000   1.46  7.189375e-51
## model=Audi- A6                 0.00000  0.0000000   1.62  1.778571e-56
## model=Audi- Q5                 0.00000  0.0000000   2.06  5.170703e-72
## model=Audi- A1                 0.00000  0.0000000   2.74  2.249093e-96
## model=Audi- A4                 0.00000  0.0000000   2.84 5.402444e-100
## model=Audi- Q3                 0.00000  0.0000000   2.86 1.017546e-100
## model=Audi- A3                 0.00000  0.0000000   3.84 9.964262e-137
## manufacturer=Audi              0.00000  0.0000000  21.14  0.000000e+00
##                                  v.test
## manufacturer=VW               29.972707
## manufacturer=Mercedes         27.367713
## manufacturer=BMW              23.807993
```

```
## model=VW- Golf                  15.854101
## model=Mercedes- C Class         13.661658
## model=VW- Polo                  12.503335
## model=Mercedes- A Class         10.885069
## model=BMW- 3 Series             10.512338
## model=BMW- 1 Series             10.078647
## model=VW- Tiguan                 9.421281
## model=Mercedes- E Class          8.941107
## f.engineSize=Large               8.679183
## model=Mercedes- GLC Class        7.135827
## model=BMW- 2 Series              7.135827
## model=BMW- 5 Series              6.931977
## model=VW- Passat                 6.615464
## model=VW- Up                     6.284423
## model=BMW- 4 Series              6.054555
## f.mpg=High                       6.032078
## model=Mercedes- GLA Class        5.896762
## model=VW- T-Roc                  5.398273
## model=BMW- X1                    5.222509
## model=Mercedes- GLE Class        4.659483
## model=Mercedes- CL Class         4.559563
## model=Mercedes- B Class          4.559563
## model=BMW- X3                    4.457632
## model=VW- Touareg                4.083075
## model=BMW- X5                    4.083075
## f.price=Low-priced               3.916211
## f.mpg=Very High                  3.750889
## model=Mercedes- SL CLASS         3.357611
## fuelType=Hybrid                  3.231175
## model=VW- T-Cross                3.223104
## fuelType=Diesel                  3.219559
## model=VW- Scirocco               3.154021
## model=VW- Touran                 2.938603
## model=VW- Sharan                 2.938603
## model=Mercedes- CLS Class        2.938603
## model=BMW- X2                    2.938603
## model=Mercedes- S Class          2.863797
## transmission=f.Trans-SemiAuto    2.825946
## model=VW- Arteon                 2.787333
## model=BMW- X4                    2.787333
## model=VW- Golf SV                2.709098
## model=BMW- M4                    2.709098
## model=Mercedes- V Class          2.628969
## f.miles=Used                     2.601022
## model=BMW- 6 Series              2.375731
## fuelType=Electric                2.194355
## model=Mercedes- X-CLASS          2.194355
## model=Mercedes- GLS Class        2.194355
## model=Mercedes- GL Class         2.194355
## model=VW- Caravelle              2.099202
## model=Mercedes- SLK              2.099202
## model=Audi- RS3                 -2.007875
## model=Audi- SQ5                 -2.596193
## model=Audi- S3                  -2.596193
```

```
## model=Audi- RS6                -3.091980
## model=Audi- R8                 -3.091980
## model=Audi- Q8                 -3.527778
## f.price=Expensive              -3.743935
## fuelType=Petrol                -4.118385
## transmission=f.Trans-Manual    -4.148776
## model=Audi- A7                 -5.508634
## model=Audi- A8                 -5.777499
## f.engineSize=Medium            -6.920780
## f.mpg=Low                      -7.827943
## model=Audi- TT                -10.071246
## model=Audi- Q7                -10.821420
## model=Audi- A5                -14.122537
## model=Audi- Q2                -15.001394
## model=Audi- A6                -15.835213
## model=Audi- Q5                -17.945857
## model=Audi- A1                -20.831375
## model=Audi- A4                -21.226794
## model=Audi- Q3                -21.305125
## model=Audi- A3                -24.888285
## manufacturer=Audi                    -Inf
##
## $`Audi Yes`
##                                Cla/Mod     Mod/Cla Global       p.value
## manufacturer=Audi            100.00000 100.0000000  21.14  0.000000e+00
## model=Audi- A3               100.00000  18.1646168   3.84 9.964262e-137
## model=Audi- Q3               100.00000  13.5288553   2.86 1.017546e-100
## model=Audi- A4               100.00000  13.4342479   2.84 5.402444e-100
## model=Audi- A1               100.00000  12.9612110   2.74  2.249093e-96
## model=Audi- Q5               100.00000   9.7445601   2.06  5.170703e-72
## model=Audi- A6               100.00000   7.6631977   1.62  1.778571e-56
## model=Audi- Q2               100.00000   6.9063387   1.46  7.189375e-51
## model=Audi- A5               100.00000   6.1494797   1.30  2.758792e-45
## model=Audi- Q7               100.00000   3.6896878   0.78  2.725205e-27
## model=Audi- TT               100.00000   3.2166509   0.68  7.403402e-24
## f.mpg=Low                     29.12698  34.7209082  25.20  4.959163e-15
## f.engineSize=Medium           25.09929  59.7918638  50.36  4.491623e-12
## model=Audi- A8               100.00000   1.1352886   0.24  7.581939e-09
## model=Audi- A7               100.00000   1.0406812   0.22  3.616280e-08
## transmission=f.Trans-Manual   24.36040  41.4380322  35.96  3.342572e-05
## fuelType=Petrol               23.95038  47.4929044  41.92  3.815371e-05
## f.price=Expensive             24.93986  29.4228950  24.94  1.811609e-04
## model=Audi- Q8               100.00000   0.4730369   0.10  4.190629e-04
## model=Audi- RS6              100.00000   0.3784295   0.08  1.988260e-03
## model=Audi- R8               100.00000   0.3784295   0.08  1.988260e-03
## model=Audi- SQ5              100.00000   0.2838221   0.06  9.426316e-03
## model=Audi- S3               100.00000   0.2838221   0.06  9.426316e-03
## model=Audi- RS3              100.00000   0.1892148   0.04  4.465661e-02
## model=VW- Caravelle            0.00000   0.0000000   0.28  3.579906e-02
## model=Mercedes- SLK            0.00000   0.0000000   0.28  3.579906e-02
## fuelType=Electric              0.00000   0.0000000   0.30  2.820989e-02
## model=Mercedes- X-CLASS        0.00000   0.0000000   0.30  2.820989e-02
## model=Mercedes- GLS Class      0.00000   0.0000000   0.30  2.820989e-02
## model=Mercedes- GL Class       0.00000   0.0000000   0.30  2.820989e-02
```

```
## model=BMW- 6 Series             0.00000  0.0000000  0.34  1.751421e-02
## f.miles=Used                   18.56000 21.9489120 25.00  9.294643e-03
## model=Mercedes- V Class         0.00000  0.0000000  0.40  8.564428e-03
## model=VW- Golf SV               0.00000  0.0000000  0.42  6.746637e-03
## model=BMW- M4                   0.00000  0.0000000  0.42  6.746637e-03
## model=VW- Arteon                0.00000  0.0000000  0.44  5.314382e-03
## model=BMW- X4                   0.00000  0.0000000  0.44  5.314382e-03
## transmission=f.Trans-SemiAuto  19.03743 33.6802271 37.40  4.714115e-03
## model=Mercedes- S Class         0.00000  0.0000000  0.46  4.185957e-03
## model=VW- Touran                0.00000  0.0000000  0.48  3.296956e-03
## model=VW- Sharan                0.00000  0.0000000  0.48  3.296956e-03
## model=Mercedes- CLS Class       0.00000  0.0000000  0.48  3.296956e-03
## model=BMW- X2                   0.00000  0.0000000  0.48  3.296956e-03
## model=VW- Scirocco              0.00000  0.0000000  0.54  1.610376e-03
## fuelType=Diesel                19.50442 52.1286660 56.50  1.283879e-03
## model=VW- T-Cross               0.00000  0.0000000  0.56  1.268094e-03
## fuelType=Hybrid                 6.25000  0.3784295  1.28  1.232824e-03
## model=Mercedes- SL CLASS        0.00000  0.0000000  0.60  7.861929e-04
## f.mpg=Very High                17.44000 20.6244087 25.00  1.762084e-04
## f.price=Low-priced             17.28000 20.4351939 25.00  8.995141e-05
## model=VW- Touareg               0.00000  0.0000000  0.84  4.444368e-05
## model=BMW- X5                   0.00000  0.0000000  0.84  4.444368e-05
## model=BMW- X3                   0.00000  0.0000000  0.98  8.286995e-06
## model=Mercedes- CL Class        0.00000  0.0000000  1.02  5.126008e-06
## model=Mercedes- B Class         0.00000  0.0000000  1.02  5.126008e-06
## model=Mercedes- GLE Class       0.00000  0.0000000  1.06  3.170051e-06
## model=BMW- X1                   0.00000  0.0000000  1.30  1.765149e-07
## model=VW- T-Roc                 0.00000  0.0000000  1.38  6.728526e-08
## model=Mercedes- GLA Class       0.00000  0.0000000  1.62  3.707033e-09
## f.mpg=High                     15.11628 17.2185430 24.08  1.618647e-09
## model=BMW- 4 Series             0.00000  0.0000000  1.70  1.408067e-09
## model=VW- Up                    0.00000  0.0000000  1.82  3.290728e-10
## model=VW- Passat                0.00000  0.0000000  2.00  3.703873e-11
## model=BMW- 5 Series             0.00000  0.0000000  2.18  4.149981e-12
## model=Mercedes- GLC Class       0.00000  0.0000000  2.30  9.620669e-13
## model=BMW- 2 Series             0.00000  0.0000000  2.30  9.620669e-13
## f.engineSize=Large             12.06262 12.3935667 21.72  3.986209e-18
## model=Mercedes- E Class         0.00000  0.0000000  3.50  3.852925e-19
## model=VW- Tiguan                0.00000  0.0000000  3.86  4.456163e-21
## model=BMW- 1 Series             0.00000  0.0000000  4.38  6.866524e-24
## model=BMW- 3 Series             0.00000  0.0000000  4.74  7.579104e-26
## model=Mercedes- A Class         0.00000  0.0000000  5.06  1.357959e-27
## model=VW- Polo                  0.00000  0.0000000  6.56  7.158404e-36
## model=Mercedes- C Class         0.00000  0.0000000  7.74  1.720220e-42
## model=VW- Golf                  0.00000  0.0000000 10.20  1.316978e-56
## manufacturer=BMW                0.00000  0.0000000 21.14 2.760168e-125
## manufacturer=Mercedes           0.00000  0.0000000 26.74 6.647141e-165
## manufacturer=VW                 0.00000  0.0000000 30.98 2.226616e-197
##                                   v.test
## manufacturer=Audi                    Inf
## model=Audi- A3                 24.888285
## model=Audi- Q3                 21.305125
## model=Audi- A4                 21.226794
## model=Audi- A1                 20.831375
```

```
## model=Audi- Q5                  17.945857
## model=Audi- A6                  15.835213
## model=Audi- Q2                  15.001394
## model=Audi- A5                  14.122537
## model=Audi- Q7                  10.821420
## model=Audi- TT                  10.071246
## f.mpg=Low                        7.827943
## f.engineSize=Medium              6.920780
## model=Audi- A8                   5.777499
## model=Audi- A7                   5.508634
## transmission=f.Trans-Manual      4.148776
## fuelType=Petrol                  4.118385
## f.price=Expensive                3.743935
## model=Audi- Q8                   3.527778
## model=Audi- RS6                  3.091980
## model=Audi- R8                   3.091980
## model=Audi- SQ5                  2.596193
## model=Audi- S3                   2.596193
## model=Audi- RS3                  2.007875
## model=VW- Caravelle             -2.099202
## model=Mercedes- SLK             -2.099202
## fuelType=Electric               -2.194355
## model=Mercedes- X-CLASS         -2.194355
## model=Mercedes- GLS Class       -2.194355
## model=Mercedes- GL Class        -2.194355
## model=BMW- 6 Series             -2.375731
## f.miles=Used                    -2.601022
## model=Mercedes- V Class         -2.628969
## model=VW- Golf SV               -2.709098
## model=BMW- M4                   -2.709098
## model=VW- Arteon                -2.787333
## model=BMW- X4                   -2.787333
## transmission=f.Trans-SemiAuto  -2.825946
## model=Mercedes- S Class         -2.863797
## model=VW- Touran                -2.938603
## model=VW- Sharan                -2.938603
## model=Mercedes- CLS Class       -2.938603
## model=BMW- X2                   -2.938603
## model=VW- Scirocco              -3.154021
## fuelType=Diesel                 -3.219559
## model=VW- T-Cross               -3.223104
## fuelType=Hybrid                 -3.231175
## model=Mercedes- SL CLASS        -3.357611
## f.mpg=Very High                 -3.750889
## f.price=Low-priced              -3.916211
## model=VW- Touareg               -4.083075
## model=BMW- X5                   -4.083075
## model=BMW- X3                   -4.457632
## model=Mercedes- CL Class        -4.559563
## model=Mercedes- B Class         -4.559563
## model=Mercedes- GLE Class       -4.659483
## model=BMW- X1                   -5.222509
## model=VW- T-Roc                 -5.398273
## model=Mercedes- GLA Class       -5.896762
```

```
## f.mpg=High                       -6.032078
## model=BMW- 4 Series              -6.054555
## model=VW- Up                     -6.284423
## model=VW- Passat                 -6.615464
## model=BMW- 5 Series              -6.931977
## model=Mercedes- GLC Class        -7.135827
## model=BMW- 2 Series              -7.135827
## f.engineSize=Large               -8.679183
## model=Mercedes- E Class          -8.941107
## model=VW- Tiguan                 -9.421281
## model=BMW- 1 Series             -10.078647
## model=BMW- 3 Series             -10.512338
## model=Mercedes- A Class         -10.885069
## model=VW- Polo                  -12.503335
## model=Mercedes- C Class         -13.661658
## model=VW- Golf                  -15.854101
## manufacturer=BMW                -23.807993
## manufacturer=Mercedes           -27.367713
## manufacturer=VW                 -29.972707
```