Deliverable 2

PCA, Clustering and MCA

Pere Arnau Alegre & Andrés Jiménez González

May 1, 2022

Contents

1 Data description

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

1.1 Variables

- Model
  - A string indicating the model of the car.
- Year
  - A discrete numeric variable to indicate the year the car was sold
- Price
  - Continuous variable indicating the price at which the car was sold

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

2 Loading of Required Packages for the deliverable

We load the necessary packages and set the working directory

```r
# setwd('C:/Users/TOREROS-II/Documents/GitHub/adei/adei/deliverable2')
setwd("C:/Users/Arnau/Desktop/adei/deliverable2")
# Load Required Packages
options(contrasts = c("contr.treatment", "contr.treatment"))
requiredPackages <- c("missMDA", "chemometrics", "mvoutlier", "effects",
    "FactoMineR", "car", "factoextra", "RColorBrewer", "dplyr", "ggmap",
    "ggthemes", "knitr", "corrplot")
missingPackages <- requiredPackages[!(requiredPackages %in% installed.packages()[,
    "Package"])]
if (length(missingPackages)) install.packages(missingPackages)
lapply(requiredPackages, require, character.only = TRUE)
```

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

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

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

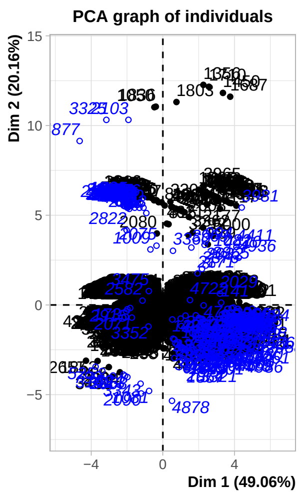
2.1 Some useful functions

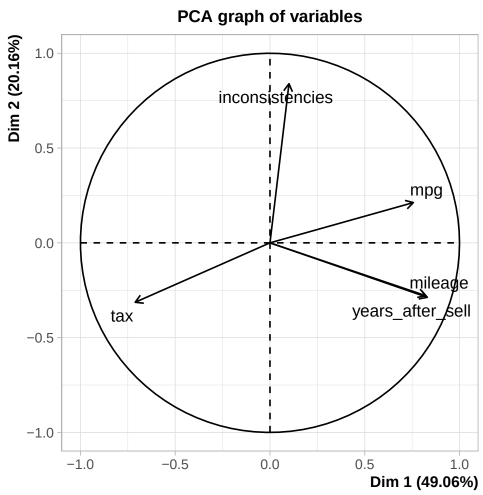
```r
calcQ <- function(x) {
    # Function to calculate the different quartiles
    s.x <- summary(x)
    iqr <- s.x[5] - s.x[2]
    list(souti = s.x[2] - 3 * iqr, mouti = s.x[2] - 1.5 * iqr, min = s.x[1],
        q1 = s.x[2], q2 = s.x[3], q3 = s.x[5], max = s.x[6], mouts = s.x[5] +
            1.5 *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

3 Principal Component Analysis

3.1 Calculate total variance explained by each principal component

```
vars_con <- c("mileage", "tax", "mpg", "years_after_sell", "inconsistencies")
vars_dis <- c("transmission", "fuelType", "engineSize", "manufacturer")
vars_res <- c("price", "Audi")
res.pca <- PCA(df[, vars_con], ind.sup = llmout)
```






```
summary(res.pca)
```

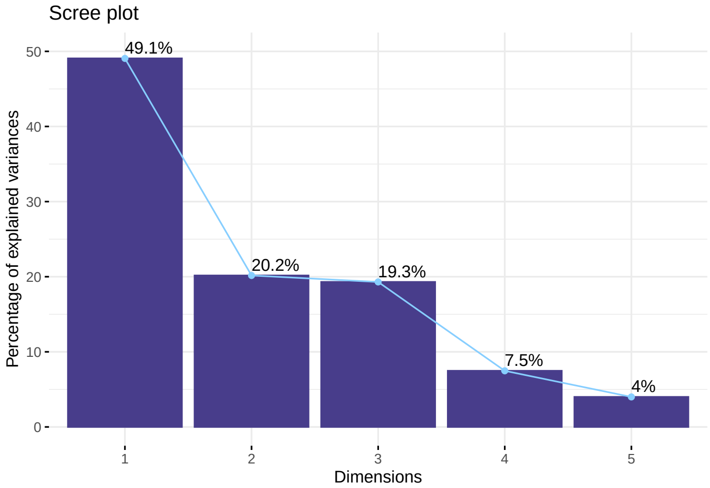
```
##
## Call:
## PCA(X = df[, vars_con], ind.sup = llmout)
##
##
## Eigenvalues
##                        Dim.1   Dim.2   Dim.3   Dim.4   Dim.5
## Variance               2.453   1.008   0.965   0.374   0.200
## % of var.             49.062  20.163  19.303   7.475   3.998
## Cumulative % of var.  49.062  69.225  88.528  96.002 100.000
##
## Individuals (the 10 first)
##                     Dist    Dim.1    ctr   cos2    Dim.2    ctr   cos2
## 1               |  1.026 |  0.556  0.003  0.294 | -0.283  0.002  0.076 |
## 2               |  1.342 |  0.431  0.002  0.103 | -0.811  0.013  0.366 |
## 3               |  3.701 |  2.831  0.068  0.585 | -1.419  0.041  0.147 |
## 4               |  1.831 |  0.996  0.008  0.296 | -0.059  0.000  0.001 |
## 5               |  1.234 | -0.725  0.004  0.345 | -0.476  0.005  0.149 |
## 6               |  0.846 |  0.265  0.001  0.098 | -0.129  0.000  0.023 |
## 7               |  1.781 |  1.187  0.012  0.444 |  0.363  0.003  0.042 |
## 8               |  2.819 |  2.262  0.043  0.644 |  0.654  0.009  0.054 |
## 9               |  0.805 |  0.368  0.001  0.209 | -0.418  0.004  0.269 |
## 10              |  1.659 | -1.612  0.022  0.944 |  0.098  0.000  0.004 |
##                  Dim.3    ctr   cos2
## 1                0.087  0.000  0.007 |
```

```
## 2                  0.954  0.020  0.506 |
## 3                  1.632  0.057  0.194 |
## 4                 -0.366  0.003  0.040 |
## 5                  0.622  0.008  0.254 |
## 6                 -0.130  0.000  0.024 |
## 7                 -0.907  0.018  0.260 |
## 8                 -1.515  0.049  0.289 |
## 9                  0.388  0.003  0.232 |
## 10                -0.170  0.001  0.010 |
##
## Supplementary individuals (the 10 first)
##                      Dist    Dim.1   cos2    Dim.2   cos2    Dim.3   cos2
## 59                |  7.194 | -1.672  0.054 |  5.591  0.604 |  4.133  0.330 |
## 130               |  4.312 |  3.446  0.639 | -0.778  0.033 |  0.681  0.025 |
## 141               |  3.788 |  3.258  0.740 |  0.130  0.001 | -0.754  0.040 |
## 209               |  4.428 |  4.034  0.830 | -0.621  0.020 |  0.314  0.005 |
## 361               |  3.400 |  2.221  0.427 | -0.781  0.053 |  0.673  0.039 |
## 403               |  7.444 | -1.966  0.070 |  5.414  0.529 |  4.493  0.364 |
## 450               |  7.289 | -1.971  0.073 |  5.797  0.632 |  3.863  0.281 |
## 460               |  7.550 | -2.399  0.101 |  5.666  0.563 |  4.154  0.303 |
## 496               |  7.434 | -1.872  0.063 |  5.356  0.519 |  4.577  0.379 |
## 521               |  7.450 | -1.999  0.072 |  5.431  0.531 |  4.468  0.360 |
##
## Variables
##                     Dim.1    ctr   cos2    Dim.2    ctr   cos2    Dim.3    ctr
## mileage           |  0.822 27.554  0.676 | -0.279  7.716  0.078 |  0.384 15.254
## tax               | -0.711 20.618  0.506 | -0.313  9.746  0.098 |  0.473 23.143
## mpg               |  0.756 23.328  0.572 |  0.212  4.475  0.045 | -0.430 19.202
## years_after_sell  |  0.830 28.091  0.689 | -0.287  8.187  0.083 |  0.353 12.890
## inconsistencies   |  0.100  0.410  0.010 |  0.839 69.875  0.704 |  0.534 29.511
##                    cos2
## mileage           0.147 |
## tax               0.223 |
## mpg               0.185 |
## years_after_sell  0.124 |
## inconsistencies   0.285 |
```

According to the Kaiser criteria we should keep 2 dimensions, because it says that we should keep that dimensions with variance > 1.

3.2 Elbow

```r
fviz_screeplot(res.pca, addlabels = TRUE, ylim = c(0, 50), barfill = "darkslateblue",
    barcolor = "darkslateblue", linecolor = "skyblue1")
```



As we have a not-so-ideal scree plot curve, we have to choose between a couple ways of deciding how many dimensions keep: Kaiser rule: pick PCs with eigenvalues of at least 1. Proportion of variance plot: the selected PCs should be able to describe at least 80% of the variance.

If we follow Kaiser rule we should keep only 2 dimensions, but considering that the 3rd dimensions variance is 0.965 (which is very close to 1) and that with only 2 dimensions we don't describe at least 80% of the variance, we have decided to keep 3 dimensions.

3.3 Interpreting the axes

Variables point of view coordinates, quality of representation, contribution of the variables ### Axe 1

```
res.dimdes <- dimdesc(res.pca, axes = 1:3, proba = 0.01)
res.dimdes$Dim.1
```

```
## $quanti
##                    correlation      p.value
## years_after_sell     0.8301146 0.000000e+00
## mileage              0.8221468 0.000000e+00
## mpg                  0.7564711 0.000000e+00
## inconsistencies      0.1002873 2.750029e-12
## tax                 -0.7111829 0.000000e+00
##
## attr(,"class")
## [1] "condes" "list"
```

Dimension 1 has a great direct correlation with variables years_after_sell, mileage and mpg, and a great inverse correlation with variable tax.

3.3.1 Axe 2

```
res.dimdes$Dim.2
```

```
## $quanti
##                    correlation       p.value
## inconsistencies      0.8393053   0.000000e+00
```

```
## mpg                  0.2124111  1.856174e-50
## mileage             -0.2788963  4.118839e-87
## years_after_sell    -0.2872934  1.501178e-92
## tax                 -0.3134575 9.988358e-111
##
## attr(,"class")
## [1] "condes" "list"
```

Dimension 2 has a great direct correlation with variable inconsistencies; and a low inverse correlation with variables tax, years_after_sell and mileage

3.3.2 Axe 3

```r
res.dimdes$Dim.3
```

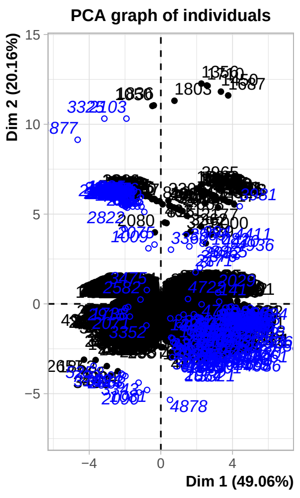
```
## $quanti
##                    correlation       p.value
## inconsistencies      0.5336913  0.000000e+00
## tax                  0.4726139 1.098575e-267
## mileage              0.3836979 2.010273e-169
## years_after_sell     0.3527130 1.140268e-141
## mpg                 -0.4304934 1.884852e-217
##
## attr(,"class")
## [1] "condes" "list"
```

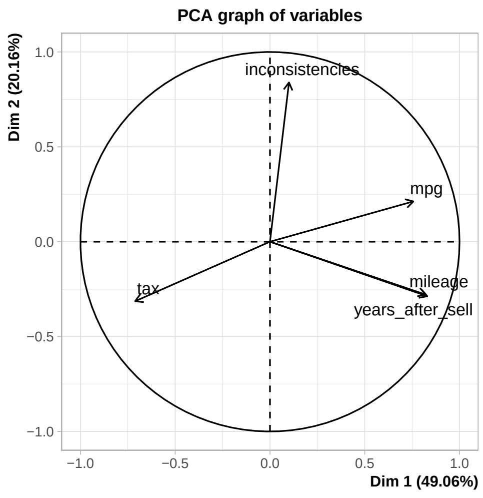
Dimension 3 has a good direct correlation with variable inconsistencies and tax; and a good inverse correlation with variables mpg.

3.4 Individuals point of view

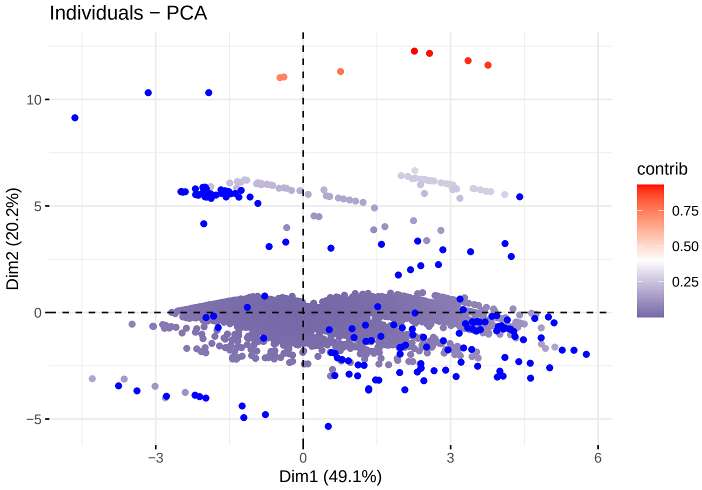
3.4.1 Contribution

```r
res.pca <- PCA(df[, vars_con], ind.sup = llmout)
```





```r
# We perform a PCA using multivariate outliers as supplementary
# observations
fviz_pca_ind(res.pca, col.ind = "contrib", geom = "point") + scale_color_gradient2(low = "da
    mid = "white", high = "red", midpoint = 0.4)
```

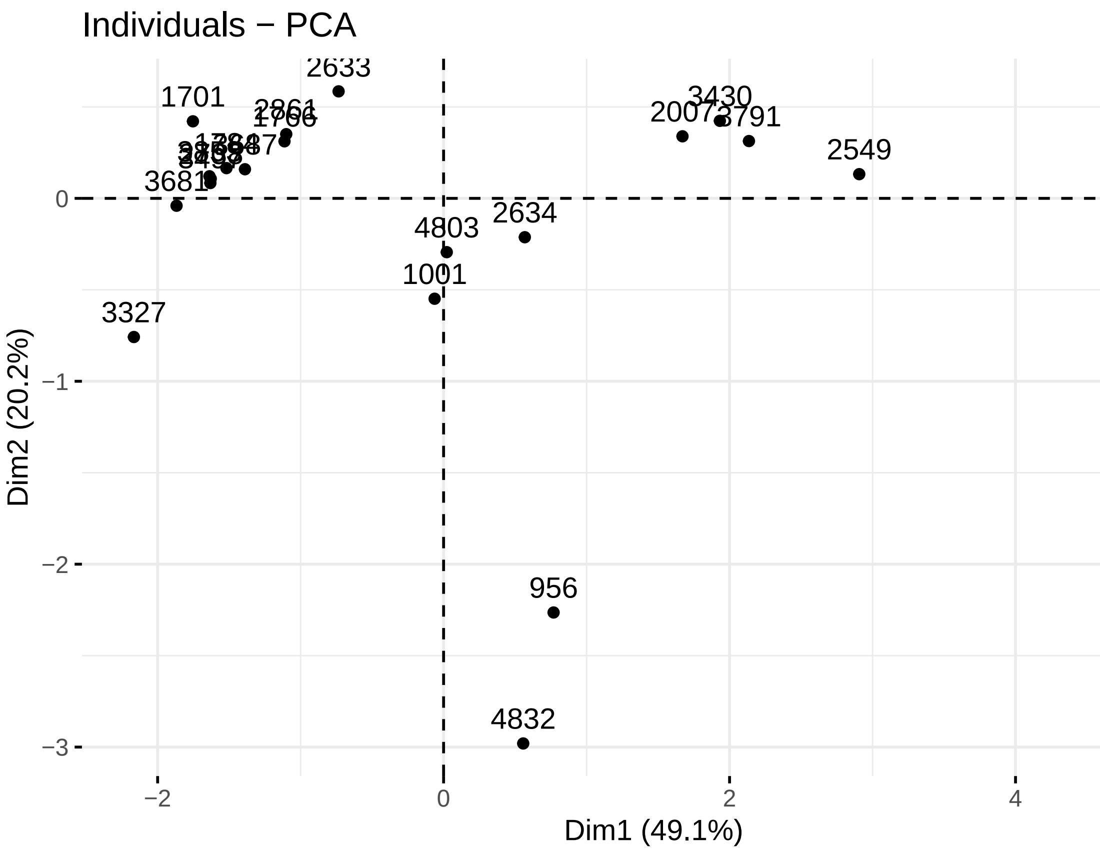


We can see that there are some individuals that are too contributive. So now, let s try to understand them better with extreme individuals.

3.4.2 Extreme individuals

```r
rang <- order(res.pca$ind$coord[, 1])
contrib.extremes <- c(row.names(df)[rang[1:10]], row.names(df)[rang[(length(rang) -
    10):length(rang)]])
```

```r
fviz_pca_ind(res.pca, select.ind = list(names = contrib.extremes))
```



3.4.2.1 In dimension 1:

We can now have a look at them:

```
df[955, 1:19]  #most contributive observation in positive 1st dimension
```

```
##         model year price   transmission mileage      fuelType tax  mpg
## 955 Audi- A3 2016  7500 f.Trans-Manual  116310 f.Fuel-Diesel   0 83.1
##     engineSize manufacturer     f.price Audi years_after_sell         f.tax
## 955        1.6         Audi super cheap  Yes                6 f.tax-[0,125]
##                       f.mileage             f.mpg      f.year inconsistencies
## 955 f.mil-(3.45e+04,1.19e+05] f.mpg-(61.4,88.3] [2008,2016]               0
##         mout
## 955 MvOut.Yes
```

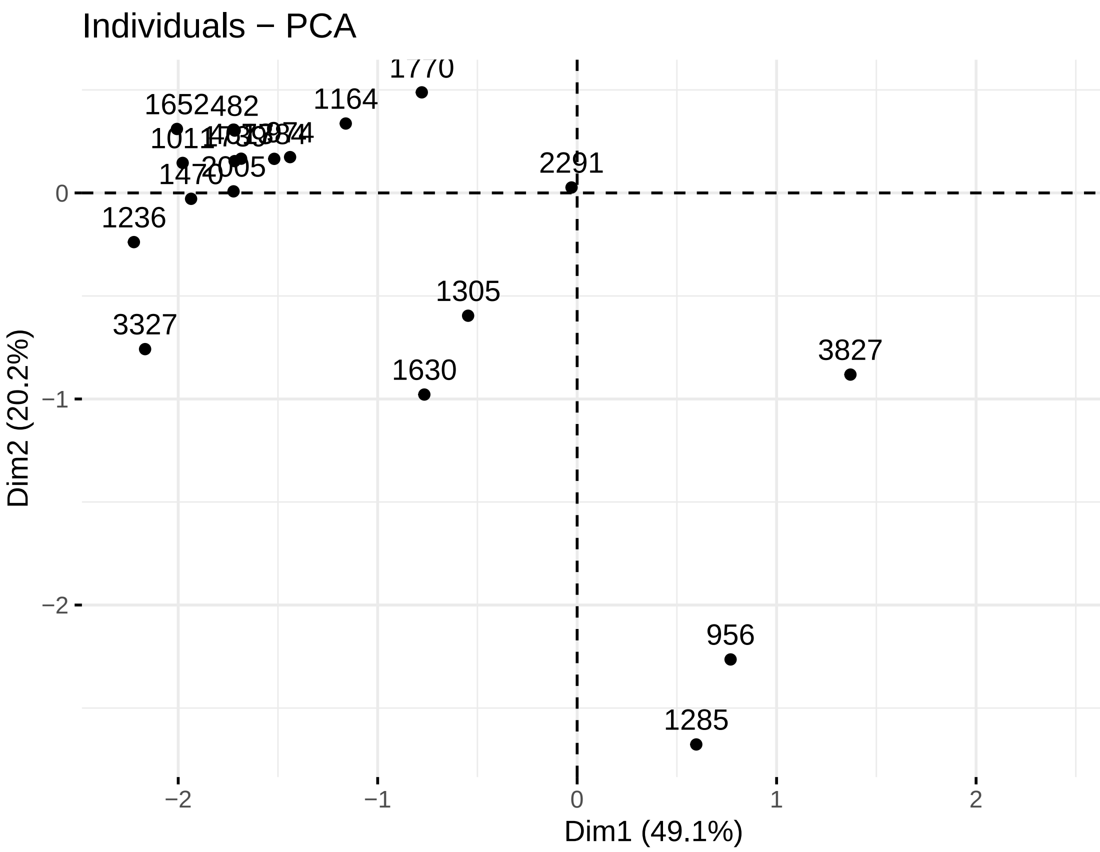
```
df[3327, 1:19]  #most contributive observation in negative 1st dimension
```

```
##                    model year price      transmission mileage      fuelType tax
## 3327 Mercedes- X-CLASS 2018 28990 f.Trans-Automatic   10000 f.Fuel-Diesel 240
##       mpg engineSize manufacturer             f.price Audi years_after_sell
## 3327 35.8        2.3     Mercedes extremely expensive   No                4
##                  f.tax                 f.mileage            f.mpg      f.year
## 3327 f.tax-(155,580] f.mil-(5.99e+03,1.71e+04] f.mpg-[20,44.8] (2017,2019]
##      inconsistencies     mout
## 3327               0 MvOut.No
```

The 1st observation, 955, has a very high mileage and mpg, is super cheap, has low taxes and was sold long ago. The 2nd observation, 3327, is the opposite of the 1st. It has very low mileage, but it is extremly expensive and has very high taxes.

```r
rang <- order(res.pca$ind$coord[, 2])
contrib.extremes <- c(row.names(df)[rang[1:10]], row.names(df)[rang[(length(rang) -
    10):length(rang)]])
```

```r
fviz_pca_ind(res.pca, select.ind = list(names = contrib.extremes))
```



3.4.2.2 In dimension 2:

We can now have a look at them:

```r
df[3327, 1:19]  #most contributive observation in positive 2nd dimension
```

```
##                    model year price      transmission mileage      fuelType tax
## 3327 Mercedes- X-CLASS 2018 28990 f.Trans-Automatic   10000 f.Fuel-Diesel 240
##       mpg engineSize manufacturer             f.price Audi years_after_sell
## 3327 35.8        2.3     Mercedes extremely expensive   No                4
##                f.tax                     f.mileage           f.mpg      f.year
## 3327 f.tax-(155,580] f.mil-(5.99e+03,1.71e+04] f.mpg-[20,44.8] (2017,2019]
##      inconsistencies     mout
## 3327               0 MvOut.No
```

```r
df[1396, 1:19]  #most contributive observation in negative 2nd dimension
```

```
##                model year price      transmission mileage      fuelType tax  mpg
## 1396 BMW- 5 Series 2013 10037 f.Trans-Automatic   77262 f.Fuel-Diesel 125 60.1
##      engineSize manufacturer     f.price Audi years_after_sell         f.tax
```

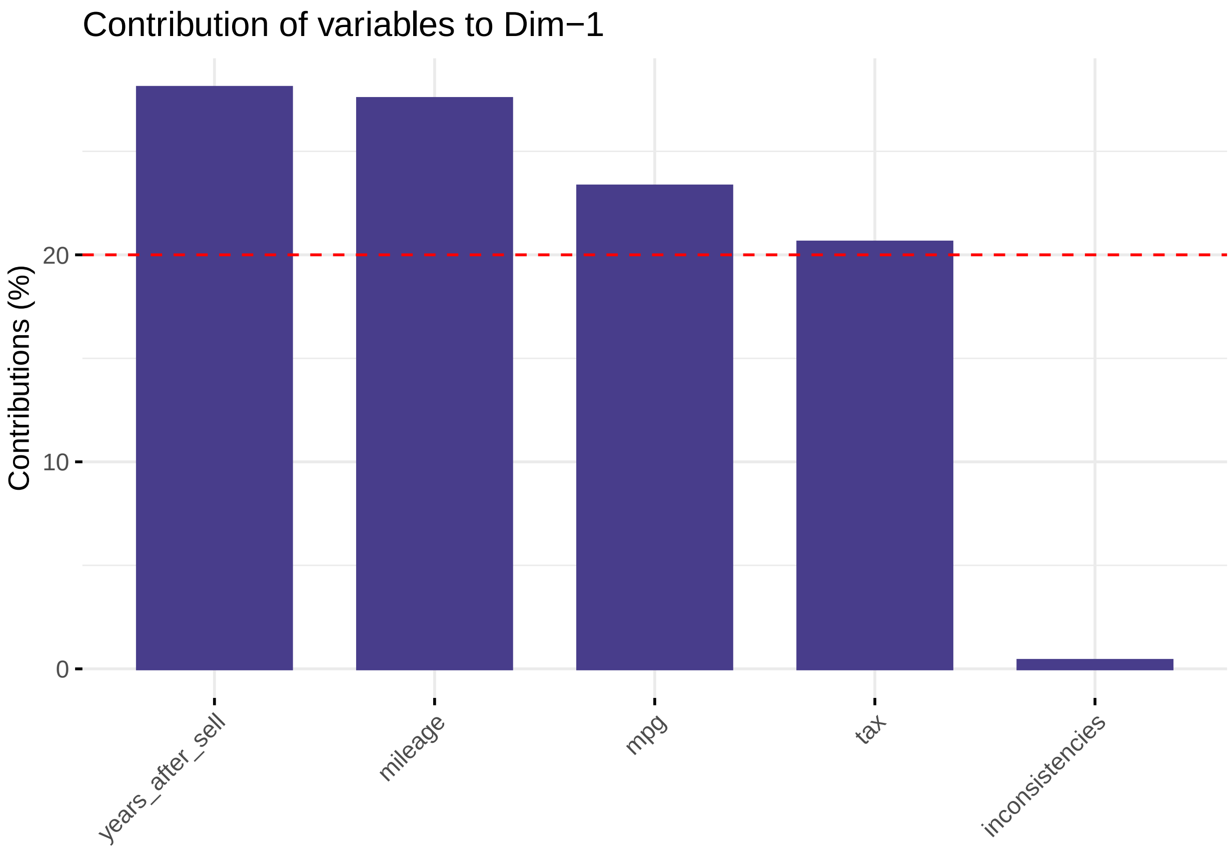
```
## 1396          2          BMW super cheap    No                 9 f.tax-[0,125]
##                        f.mileage             f.mpg      f.year inconsistencies
## 1396 f.mil-(3.45e+04,1.19e+05] f.mpg-(53.3,61.4] [2008,2016]                 0
##           mout
## 1396 MvOut.No
```

3.5 Interpreting the axes: Variables point of view coordinates, quality of representation, contribution of the variables

```
res.des <- dimdesc(res.pca)
```

3.5.1 First dimension

```
fviz_contrib(  # contributions of variables to PC1
  res.pca,
  fill = "darkslateblue",
  color = "darkslateblue",
  choice = "var",
  axes = 1,
  top = 5)
```




```
res.des$Dim.1
```

```
## $quanti
```

```
##                      correlation      p.value
## years_after_sell   0.8301146 0.000000e+00
## mileage            0.8221468 0.000000e+00
## mpg                0.7564711 0.000000e+00
## inconsistencies    0.1002873 2.750029e-12
## tax               -0.7111829 0.000000e+00
##
## attr(,"class")
## [1] "condes" "list"
```

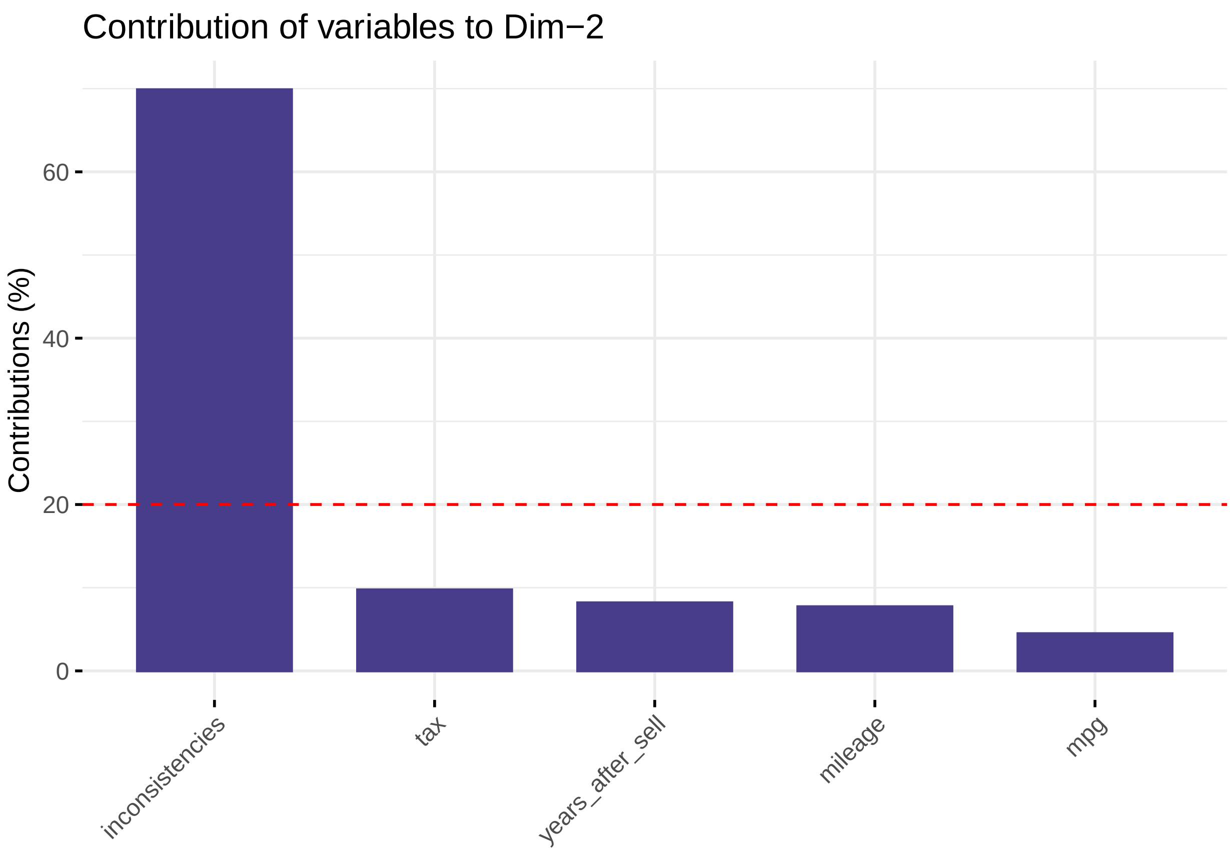
In the first dimension we see that for the quantitative variables the most positively related, from more to less, are:

- Years_after_sell (0.83)
- mileage (0.82)
- mpg (0.75)

Finally we have that variable tax is negatively correlated with a -0.71 cor value.

3.5.2 Second dimension

```
fviz_contrib(  # contributions of variables to PC1
  res.pca,
  fill = "darkslateblue",
  color = "darkslateblue",
  choice = "var",
  axes = 2,
  top = 5)
```



```r
res.des$Dim.2
```

```
## $quanti
##                     correlation       p.value
## inconsistencies     0.8393053  0.000000e+00
## mpg                 0.2124111  1.856174e-50
## mileage            -0.2788963  4.118839e-87
## years_after_sell   -0.2872934  1.501178e-92
## tax                -0.3134575 9.988358e-111
##
## attr(,"class")
## [1] "condes" "list"
```

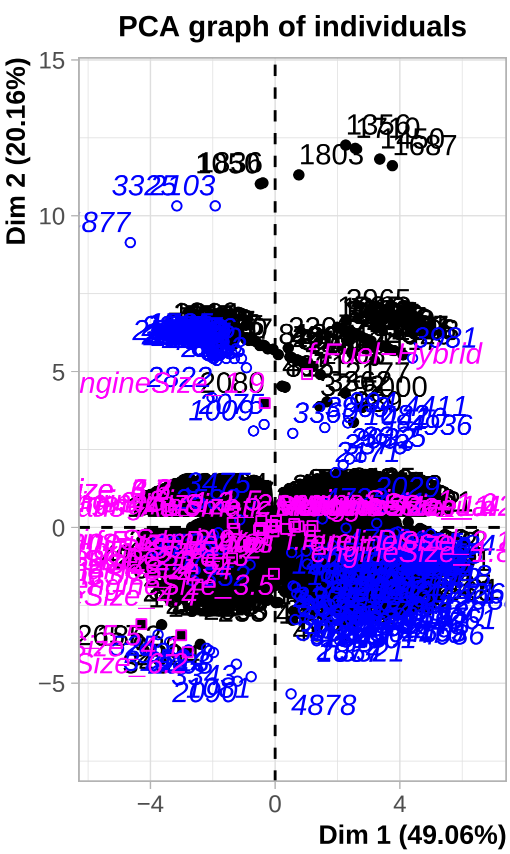
For the second dimension we see that the most positively correlated variable is "inconsistencies" with a value of 0.84, and negatively correlated is tax with a -0.31 value.

We can conclude, then, that the first dimension is the one with the biggest correlations and that explains most of the data.

3.6 PCA taking into account also supplementary variables

Our supplementary variables are Price and Audi.

```r
res.pca <- PCA(df[, c(vars_res, vars_con, vars_dis)], ind.sup = llmout,
    quanti.sup = 1, quali.sup = c(2, 8:11))
```



The variable price is strongly negatively related with the first dimension axis. This means that it has a great correlation with variable tax and inverse correlation with variables mpg, mileage and years_after_sell.

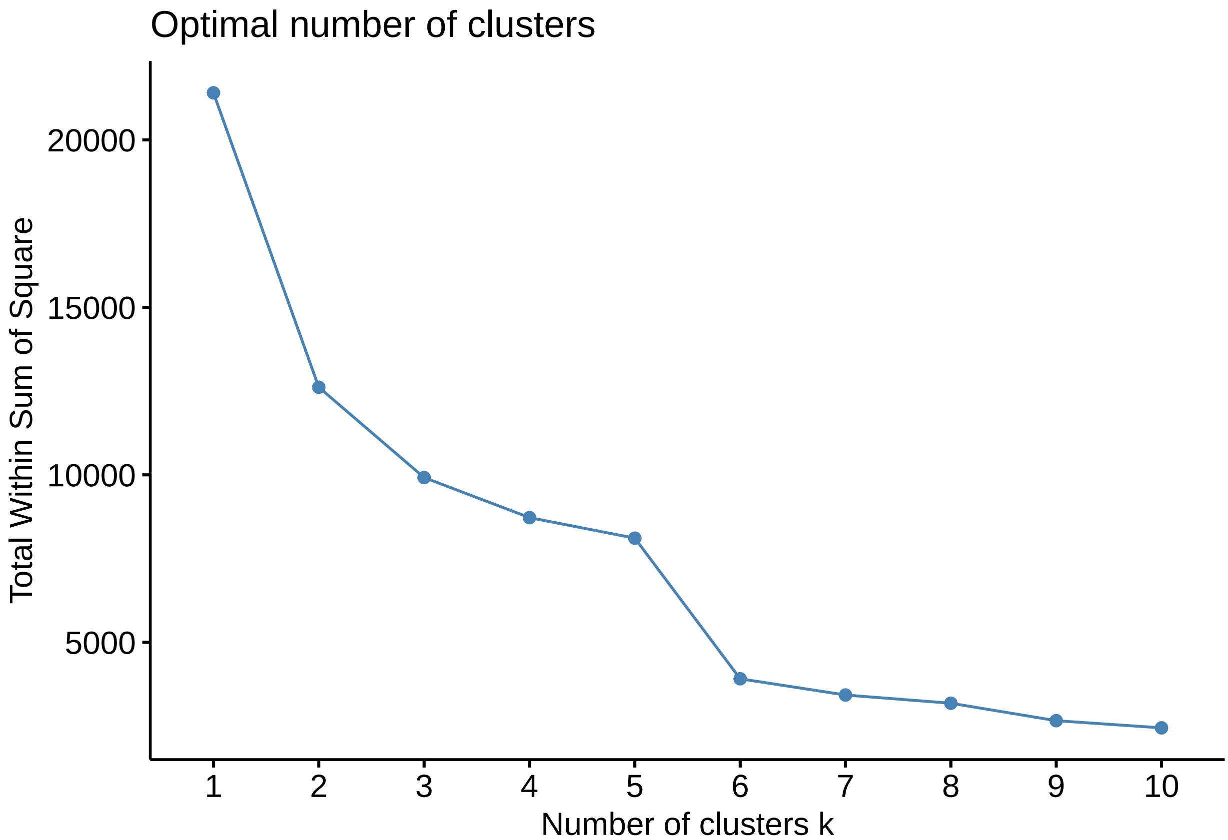
The variable Audi has very similar centroids.

4 Kmeans Classification

We determined that we had to keep 3 dimensions of the PCA to maintain an inertia above 80%. Therefore we will have to use 3 dimensions for the computation of the Kmeans.

4.1 Optimal number of clusters

```
fviz_nbclust(res.pca$ind$coord[, 1:3], kmeans, method = "wss")
```



Using the elbow method, we determined that the optimal number of clusters to compute for kmeans was 5. With 5 clusters we retain sufficient inertia and also we keep the within distance small and the between distance big.

4.2 Kmeans computation and visualization

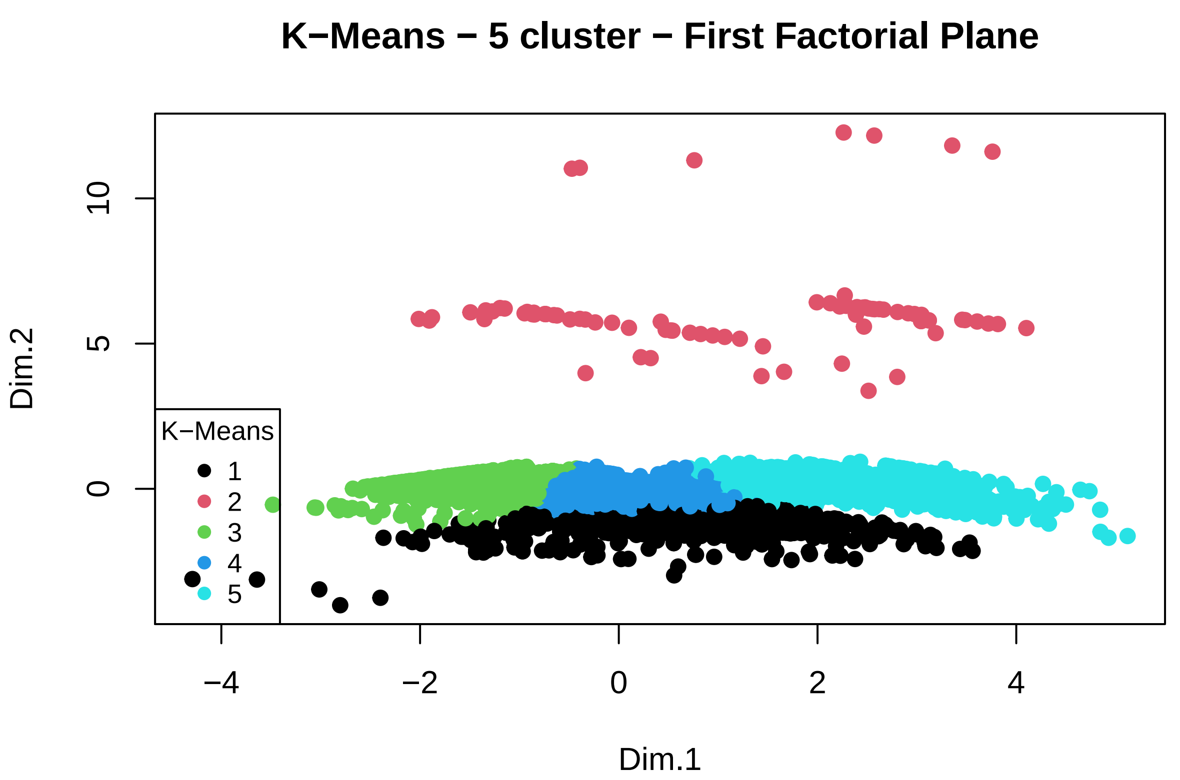
```
set.seed(1)
# when setting a seed before the generation of kmeans we force kmeans
# to generate the same result (a cluster with an inertia > 0.75)
res.km <- kmeans(res.pca$ind$coord[, 1:3], 5)

res.km$betweenss/res.km$totss  #calculate total retained inertia
```

```
## [1] 0.7888098
```

```
table(res.km$cluster)
```

```
##
##    1    2    3    4    5
##  547   82 1979 1159 1069
```

```
ff <- factor(res.km$cluster)
plot(res.pca$ind$coord[, 1:3], col = ff, pch = 19, main = "K-Means - 5 cluster - First Facto
legend("bottomleft", title = "K-Means", legend = levels(ff), col = 1:5,
    pch = 19, cex = 0.8)
```



4.2.1 Gain in inertia (in %)

After executing kmeans we get a retained inertia of the 78.88098%

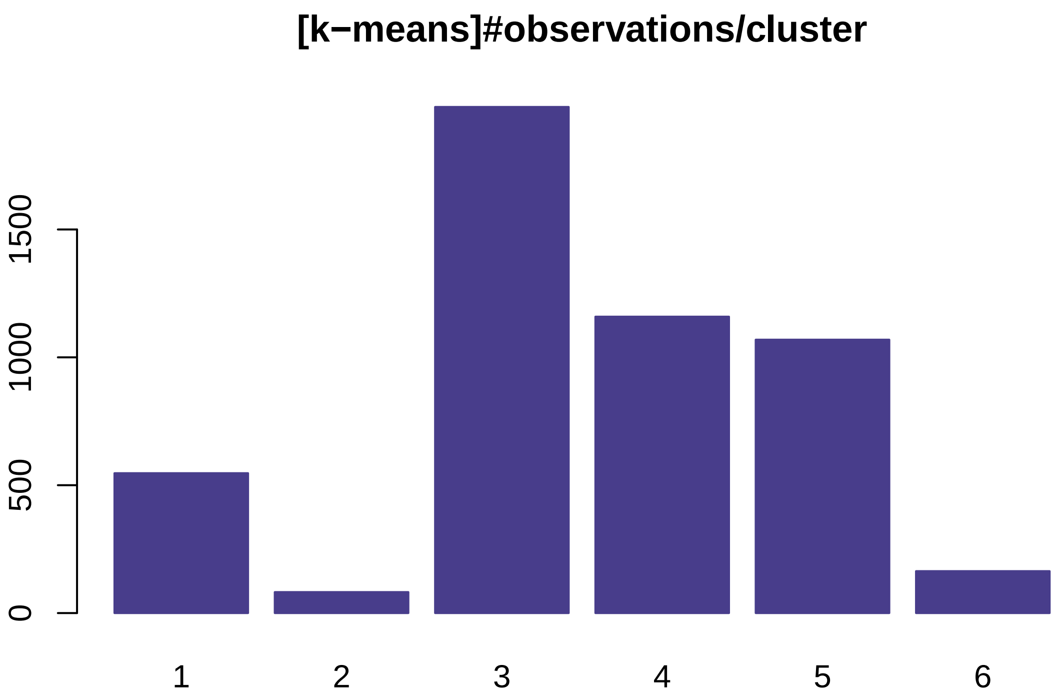
```
100 * (res.km$betweenss/res.km$totss)
```

```
## [1] 78.88098
```

4.3 Profiling of clusters

```
df$kmeans_clust <- 0
k = 5
df[-llmout, "kmeans_clust"] <- res.km$cluster
df[llmout, "kmeans_clust"] <- k + 1
```

```
df$kmeans_clust <- factor(df$kmeans_clust)
# observations that are multivariant outliers will be put in cluster
# 6
barplot(table(df$kmeans_clust), col = "darkslateblue", border = "darkslateblue",
    main = "[k-means]#observations/cluster")
```




```
res.cat <- catdes(df[c(2:12, 18:20)], num.var = 14, proba = 0.01)
```

We proceed to explain the data obtained.

4.3.1 Description of clusters by categorical variables

```
res.cat$test.chi2
```

```
##                    p.value  df
## fuelType      0.000000e+00  10
## engineSize    0.000000e+00 130
## f.price       0.000000e+00  20
## mout          0.000000e+00   5
## transmission 2.809481e-122  10
## manufacturer  9.400534e-29  15
```

We start with the description of the categorical variables that characterize the clusters, so in this output we do not have dimensions because it is the total association. We can see the intensity of the variables, in our case the variables that affect more to the clustering are **fuelType**, **engineSyze** and **f.price**, because are the one with the smallest p.value. We excluded from this test the factor variables that resulted from the grouping of the quantiles of the numerical variables because we will analyze these numerical variables later on. This way we reduce redundant information.

Next, we want to see for each cluster which are the categories that characterize them.

```
res.cat$category
```

```
## $`1`
##                                  Cla/Mod    Mod/Cla Global      p.value
## engineSize=3                   28.721541  29.9817185  11.42 8.315277e-37
## f.price=cheap                  17.764471  32.5411335  20.04 2.258241e-13
## transmission=f.Trans-Automatic 16.287879  39.3053016  26.40 2.661544e-12
## mout=MvOut.No                  11.311001 100.0000000  96.72 3.999105e-09
## manufacturer=BMW               15.867159  31.4442413  21.68 1.654497e-08
## f.price=super cheap            14.400000  26.3254113  20.00 1.423580e-04
## fuelType=f.Fuel-Diesel         12.307692  64.3510055  57.20 3.133362e-04
## fuelType=f.Fuel-Hybrid          1.515152   0.1828154   1.32 4.631522e-03
## fuelType=f.Fuel-Petrol          9.353905  35.4661792  41.48 2.351332e-03
## engineSize=1.6                  5.753425   3.8391225   7.30 4.179805e-04
## engineSize=1.3                  0.000000   0.0000000   1.48 1.767654e-04
## transmission=f.Trans-Manual     8.385985  26.6910420  34.82 1.706403e-05
## f.price=very expensive          6.772908  12.4314442  20.08 7.164641e-07
## mout=MvOut.Yes                  0.000000   0.0000000   3.28 3.999105e-09
## manufacturer=VW                 7.104914  19.5612431  30.12 3.616747e-09
## f.price=extremely expensive     4.221106   7.6782450  19.90 9.404352e-17
## engineSize=1                    0.000000   0.0000000   7.48 2.484678e-20
## engineSize=1.5                  0.000000   0.0000000  10.40 1.904449e-28
##                                   v.test
## engineSize=3                   12.673290
## f.price=cheap                   7.332544
## transmission=f.Trans-Automatic  6.994527
## mout=MvOut.No                   5.884230
## manufacturer=BMW                5.644721
## f.price=super cheap             3.804035
## fuelType=f.Fuel-Diesel          3.604017
## fuelType=f.Fuel-Hybrid         -2.831604
## fuelType=f.Fuel-Petrol         -3.041845
## engineSize=1.6                 -3.528463
## engineSize=1.3                 -3.750098
## transmission=f.Trans-Manual    -4.300205
## f.price=very expensive         -4.956847
## mout=MvOut.Yes                 -5.884230
## manufacturer=VW                -5.900831
## f.price=extremely expensive    -8.312074
## engineSize=1                   -9.239148
## engineSize=1.5                -11.062623
##
## $`2`
##                                   Cla/Mod   Mod/Cla Global      p.value
## fuelType=f.Fuel-Hybrid         71.2121212 57.317073   1.32 1.222741e-75
## transmission=f.Trans-Automatic  3.5606061 57.317073  26.40 3.077885e-09
## engineSize=2                    2.8680688 73.170732  41.84 8.483396e-09
## manufacturer=BMW                3.1365314 41.463415  21.68 5.151954e-05
```

```
## transmission=f.Trans-SemiAuto   1.0314595 24.390244  38.78 5.926477e-03
## engineSize=1                    0.0000000  0.000000   7.48 1.613366e-03
## transmission=f.Trans-Manual     0.8615738 18.292683  34.82 9.882790e-04
## engineSize=3                    0.1751313  1.219512  11.42 5.623995e-04
## fuelType=f.Fuel-Diesel          0.8741259 30.487805  57.20 1.001736e-06
## fuelType=f.Fuel-Petrol          0.4821601 12.195122  41.48 6.529925e-09
##                                   v.test
## fuelType=f.Fuel-Hybrid         18.403886
## transmission=f.Trans-Automatic  5.927389
## engineSize=2                    5.758561
## manufacturer=BMW                4.048624
## transmission=f.Trans-SemiAuto  -2.751822
## engineSize=1                   -3.153479
## transmission=f.Trans-Manual    -3.293842
## engineSize=3                   -3.449128
## fuelType=f.Fuel-Diesel         -4.891297
## fuelType=f.Fuel-Petrol         -5.802588
##
## $`3`
##                                 Cla/Mod      Mod/Cla Global       p.value
## f.price=extremely expensive   82.613065  41.53612936  19.90 1.879344e-214
## f.price=very expensive        67.928287  34.46184942  20.08  2.603456e-92
## mout=MvOut.No                 40.922250 100.00000000  96.72  2.139684e-37
## transmission=f.Trans-SemiAuto 50.644662  49.62102072  38.78  5.903404e-37
## fuelType=f.Fuel-Petrol        49.421408  51.79383527  41.48  5.590803e-33
## engineSize=1.5                61.538462  16.16978272  10.40  1.410001e-26
## engineSize=1.3                95.945946   3.58767054   1.48  1.914833e-25
## engineSize=2                  43.929254  46.43759474  41.84  1.005270e-07
## engineSize=2.9                91.666667   0.55583628   0.24  2.958053e-04
## engineSize=3                  45.884413  13.23900960  11.42  1.153016e-03
## engineSize=2.3                87.500000   0.35371400   0.16  8.512792e-03
## engineSize=1.8                 4.545455   0.05053057   0.44  2.458097e-04
## fuelType=f.Fuel-Hybrid        13.636364   0.45477514   1.32  4.171380e-06
## f.price=expensive             30.730731  15.51288530  19.98  9.985429e-11
## fuelType=f.Fuel-Diesel        33.041958  47.75138959  57.20  9.593013e-28
## engineSize=1.2                 0.000000   0.00000000   2.56  3.295616e-29
## mout=MvOut.Yes                 0.000000   0.00000000   3.28  2.139684e-37
## engineSize=1.4                 8.454810   1.46538656   6.86  3.799289e-41
## transmission=f.Trans-Manual   26.708788  23.49671551  34.82  2.853257e-43
## engineSize=2.1                 6.075949   1.21273370   7.90  1.085203e-56
## f.price=cheap                 11.576846   5.86154624  20.04 3.801024e-104
## f.price=super cheap            5.200000   2.62758969  20.00 8.159918e-168
##                                  v.test
## f.price=extremely expensive    31.255504
## f.price=very expensive         20.378338
## mout=MvOut.No                  12.779307
## transmission=f.Trans-SemiAuto  12.700127
## fuelType=f.Fuel-Petrol         11.962421
## engineSize=1.5                 10.669753
## engineSize=1.3                 10.424590
## engineSize=2                    5.325769
## engineSize=2.9                  3.618946
## engineSize=3                    3.250255
```

```
## engineSize=2.3                  2.631024
## engineSize=1.8                 -3.666587
## fuelType=f.Fuel-Hybrid         -4.602655
## f.price=expensive              -6.467172
## fuelType=f.Fuel-Diesel        -10.916688
## engineSize=1.2                -11.218841
## mout=MvOut.Yes                -12.779307
## engineSize=1.4                -13.434441
## transmission=f.Trans-Manual   -13.791855
## engineSize=2.1                -15.866258
## f.price=cheap                 -21.671625
## f.price=super cheap           -27.611220
##
## $`4`
##                                 Cla/Mod     Mod/Cla Global      p.value
## f.price=expensive             44.544545  38.3951682  19.98 1.461310e-64
## mout=MvOut.No                 23.966088 100.0000000  96.72 7.218176e-20
## engineSize=2.1                39.240506  13.3735979   7.90 9.214134e-14
## engineSize=1.4                38.483965  11.3891286   6.86 4.467651e-11
## f.price=cheap                 29.640719  25.6255393  20.04 1.119939e-07
## manufacturer=Mercedes         28.160484  32.0966350  26.42 8.211276e-07
## fuelType=f.Fuel-Diesel        25.384615  62.6402071  57.20 1.812381e-05
## transmission=f.Trans-SemiAuto 25.270758  42.2778257  38.78 5.463228e-03
## transmission=f.Trans-Automatic 20.606061  23.4685073  26.40 9.299511e-03
## engineSize=4                   7.317073   0.2588438   0.82 9.215515e-03
## manufacturer=VW               20.385126  26.4883520  30.12 1.957671e-03
## fuelType=f.Fuel-Petrol        20.877531  37.3597929  41.48 1.122762e-03
## engineSize=1.6                16.438356   5.1768766   7.30 1.083668e-03
## engineSize=1.3                 4.054054   0.2588438   1.48 6.927215e-06
## fuelType=f.Fuel-Hybrid         0.000000   0.0000000   1.32 2.423895e-08
## mout=MvOut.Yes                 0.000000   0.0000000   3.28 7.218176e-20
## f.price=super cheap           12.600000  10.8714409  20.00 1.108339e-20
## f.price=extremely expensive    6.331658   5.4357204  19.90 8.112354e-55
##                                   v.test
## f.price=expensive              16.966181
## mout=MvOut.No                   9.124329
## engineSize=2.1                  7.451705
## engineSize=1.4                  6.587676
## f.price=cheap                   5.306103
## manufacturer=Mercedes           4.930278
## fuelType=f.Fuel-Diesel          4.286833
## transmission=f.Trans-SemiAuto   2.778371
## transmission=f.Trans-Automatic -2.600843
## engineSize=4                   -2.603954
## manufacturer=VW                -3.096580
## fuelType=f.Fuel-Petrol         -3.257808
## engineSize=1.6                 -3.267853
## engineSize=1.3                 -4.495913
## fuelType=f.Fuel-Hybrid         -5.578652
## mout=MvOut.Yes                 -9.124329
## f.price=super cheap            -9.325143
## f.price=extremely expensive   -15.593082
##
```

```
## $`5`
##                                 Cla/Mod      Mod/Cla Global       p.value
## f.price=super cheap          57.2000000  53.50795136  20.00 1.773025e-180
## transmission=f.Trans-Manual  37.9666858  61.83348924  34.82  3.806234e-93
## f.price=cheap                37.2255489  34.89242283  20.04  8.281735e-39
## engineSize=1.2               60.1562500   7.20299345   2.56  6.522275e-22
## engineSize=2.1               40.7594937  15.06080449   7.90  8.545077e-20
## mout=MvOut.No                22.1050455 100.00000000  96.72  3.496737e-18
## engineSize=1.4               39.9416910  12.81571562   6.86  5.939195e-16
## fuelType=f.Fuel-Diesel       25.1748252  67.35266604  57.20  2.177881e-14
## manufacturer=VW              27.4900398  38.72778297  30.12  1.018050e-11
## engineSize=1                 35.5614973  12.44153414   7.48  5.203427e-11
## engineSize=1.6               35.0684932  11.97380730   7.30  3.774904e-10
## fuelType=f.Fuel-Hybrid        9.0909091   0.56127222   1.32  8.957589e-03
## engineSize=1.5               16.9230769   8.23199252  10.40  7.573920e-03
## engineSize=4                  0.0000000   0.00000000   0.82  4.981569e-05
## engineSize=1.3                0.0000000   0.00000000   1.48  1.602975e-08
## manufacturer=BMW             14.2988930  14.49953227  21.68  3.071294e-11
## fuelType=f.Fuel-Petrol       16.5380906  32.08606174  41.48  1.235027e-12
## engineSize=2                 16.2045889  31.71188026  41.84  1.973592e-14
## transmission=f.Trans-Automatic 14.0909091  17.39943873  26.40  7.854427e-15
## mout=MvOut.Yes                0.0000000   0.00000000   3.28  3.496737e-18
## f.price=expensive            11.2112112  10.47708138  19.98  1.998234e-20
## transmission=f.Trans-SemiAuto 11.4492006  20.76707203  38.78  4.515149e-45
## engineSize=3                  0.1751313   0.09354537  11.42  2.517563e-62
## f.price=very expensive        1.1952191   1.12254443  20.08  1.218997e-97
## f.price=extremely expensive   0.0000000   0.00000000  19.90 1.769945e-118
##                                  v.test
## f.price=super cheap           28.646486
## transmission=f.Trans-Manual   20.472251
## f.price=cheap                 13.029807
## engineSize=1.2                 9.620955
## engineSize=2.1                 9.106031
## mout=MvOut.No                  8.694073
## engineSize=1.4                 8.090558
## fuelType=f.Fuel-Diesel         7.639665
## manufacturer=VW                6.803927
## engineSize=1                   6.564996
## engineSize=1.6                 6.263062
## fuelType=f.Fuel-Hybrid        -2.613669
## engineSize=1.5                -2.670496
## engineSize=4                  -4.056490
## engineSize=1.3                -5.650161
## manufacturer=BMW              -6.643111
## fuelType=f.Fuel-Petrol        -7.101396
## engineSize=2                  -7.652337
## transmission=f.Trans-Automatic -7.769906
## mout=MvOut.Yes                -8.694073
## f.price=expensive             -9.262434
## transmission=f.Trans-SemiAuto -14.087783
## engineSize=3                 -16.661005
## f.price=very expensive       -20.970530
## f.price=extremely expensive  -23.141513
```

```
##
## $`6`
##                                  Cla/Mod    Mod/Cla Global       p.value
## mout=MvOut.Yes                100.0000000 100.000000   3.28 1.147016e-312
## f.price=super cheap             8.7000000  53.048780  20.00  1.309814e-21
## engineSize=4                   43.9024390  10.975610   0.82  8.284124e-17
## engineSize=4.4                 75.0000000   3.658537   0.16  3.045585e-08
## engineSize=5.2                100.0000000   3.048780   0.10  3.576936e-08
## transmission=f.Trans-Automatic   5.1515152  41.463415  26.40  2.151086e-05
## manufacturer=BMW                5.3505535  35.365854  21.68  4.477860e-05
## engineSize=5.5                 75.0000000   1.829268   0.08  1.364281e-04
## engineSize=3                    5.9544658  20.731707  11.42  4.790041e-04
## f.price=extremely expensive     5.1256281  31.097561  19.90  5.445480e-04
## engineSize=4.2                100.0000000   1.219512   0.04  1.069494e-03
## engineSize=3.2                100.0000000   1.219512   0.04  1.069494e-03
## f.price=cheap                   1.8962076  11.585366  20.04  3.874116e-03
## manufacturer=VW                 2.0584329  18.902439  30.12  9.804885e-04
## engineSize=1                    0.5347594   1.219512   7.48  2.862684e-04
## transmission=f.Trans-SemiAuto   2.0113461  23.780488  38.78  3.792769e-05
## engineSize=1.5                  0.0000000   0.000000  10.40  1.097915e-08
## f.price=expensive               0.4004004   2.439024  19.98  1.072549e-11
## f.price=very expensive          0.2988048   1.829268  20.08  8.089801e-13
## mout=MvOut.No                   0.0000000   0.000000  96.72 1.147016e-312
##                                   v.test
## mout=MvOut.Yes                 37.799751
## f.price=super cheap             9.548979
## engineSize=4                    8.327108
## engineSize=4.4                  5.538796
## engineSize=5.2                  5.510560
## transmission=f.Trans-Automatic   4.248602
## manufacturer=BMW                4.081331
## engineSize=5.5                  3.814556
## engineSize=3                    3.492231
## f.price=extremely expensive     3.457828
## engineSize=4.2                  3.271577
## engineSize=3.2                  3.271577
## f.price=cheap                  -2.888234
## manufacturer=VW                -3.296066
## engineSize=1                   -3.627419
## transmission=f.Trans-SemiAuto  -4.119754
## engineSize=1.5                 -5.714865
## f.price=expensive              -6.796415
## f.price=very expensive         -7.159625
## mout=MvOut.No                 -37.799751
```

- Cluster 1
  - The first thing we can notice is that cluster 1 contains 28.7% of all cars with engineSize=3 in the sample. On average 11.42% of the cars have an engine of that size, but in cluster 1 those cars are overrepresented (29.98%). In addition, cheap cars are overrepresented (20% of global mean vs 32.5% mean in cluster 1). Furthermore, automatic cars are overrepresented (26.4% of global mean vs 39.3% mean in cluster 1). Finally, the cluster has very few expensive and extremly expensive cars.

- Cluster 2
  - The first thing we can notice is that cluster 2 contains 71.21% of hybrid fuel cars in the sample. On average 1.32% of the cars use hybrid fuel, but in cluster 2 hybrid cars are overrepresented (57.32%). Diesel cars represent 57.2% of the sample, but 0.87% of them are included in Cluster 2. The same can be seen for cars that run on petrol. Cars with an engineSize of 1 represent 7.48% of the sample, but 0% of them are included in Cluster 1. Finally, on average 26.4% of the cars use an automatic transmission, but in cluster 2 automatics cars are overrepresented (57.32%) and manual and semi-automatic cars are underrepresented.
- Cluster 3
  - The first thing we can notice is that cluster 3 contains 82.6% of all extremly expensive cars in the sample. On average 19.9% of the cars are extremly expensive, but in cluster 3 those cars are overrepresented (41.54%). In addition, very expensive cars are overrepresented (20% of global mean vs 34.5% mean in cluster 3). Furthermore, semi-automatic cars are overrepresented (38.78% of global mean vs 49.62% mean in cluster 3). Finally, manual, cheap and super cheap cars are underrepresented in the class.
- Cluster 4
  - In cluster 4 we can see that there is more expensive cars than in the gobal (38.3951682% in cluster 4 vs. 19.98% in global). It happens the same with cheap cars (25.6255393% in cluster 4 vs. 20.04% in global). But for the super cheap and extremely expensive cars is the opposite, they are underrepresented. For the extremely expensive cars the global percentage is 19.90%, while the percentage in cluster 4 is 5.4357204%, and for super cheap cars the global percentage is 20.00% and the cluster 4 percentage is 10.8714409%. So we can say that cluster 4 has the cars with an average price similar to the mean.
- Cluster 5
  - Cluster 5 is defined by the cheapest cars. We can confirm this it contains a great percentage of super cheap and cheap cars and a very little percentage of extremely expensive and very expensive cars. The percentage of super cheap cars in cluster 5 is 53.51% while in the global is 20.00%, and the percentage of cheap cars in cluster 5 is 34.89% while in the global is 20.04%. We can observe the opposite behaviour with the expensive cars: the percentage of extremely expensive cars in cluster 5 is 1.12% while in the global is 20.08%, and the percentage of cheap cars in cluster 5 is 0% while in the global is 19.90%. We can also highlight that it has a lot of cars with manual transmission (61.83% in cluster 4 vs. 34.82% in global).
- Cluster 6
- This cluster groups all multivariant outliers in the sample. What we can observe from this cluser is that contains extreme observations like super cheap and extremely expensive cars and cars with a very high engine size. Also, cars with automatic transmission and that are BMW are overrepresented while semi-auto, expensive and very expensive cars are underrepresented.

We now proceed to see the quantitative variables that characterizes the clusters. We can see in the output from res.cat$quanti.var all the numeric variables that characterize the clusters. From a more detailed look, variables **year**, **mileage**, **tax** and **inconsistencies** maintain a strong relation with the cluster number because of their high eta2 value.

```
res.cat$quanti.var
```

```
##                          Eta2 P-value
## year                0.6772676       0
## price               0.3073707       0
```

```
## mileage         0.5889493       0
## tax             0.6814825       0
## mpg             0.4694748       0
## inconsistencies 0.6933317       0
```

```
res.cat$quanti
```

```
## $`1`
##                     v.test Mean in category Overall mean sd in category
## mileage          27.492521      47044.17002  23289.51910   16667.473669
## tax              22.592350        182.95247    122.91100      63.198662
## inconsistencies  -4.054791          0.00000      0.02960       0.000000
## price            -9.383459      17188.67642  21418.53580    6608.614458
## mpg             -10.286867         48.18702     53.00322       7.502146
## year            -29.963359       2014.80622   2017.21640       1.373195
##                   Overall sd       p.value
## mileage         2.141130e+04 2.157078e-166
## tax             6.585652e+01 5.153182e-113
## inconsistencies 1.808973e-01  5.017917e-05
## price           1.117048e+04  6.384322e-21
## mpg             1.160193e+01  8.076171e-25
## year            1.993281e+00 2.947434e-197
##
## $`2`
##                    v.test Mean in category Overall mean sd in category
## inconsistencies 53.283128         1.085366       0.0296   2.794253e-01
## mileage          2.948203     30203.785906   23289.5191   2.117783e+04
## tax             -4.917460        87.439024     122.9110   7.656594e+01
##                   Overall sd      p.value
## inconsistencies 1.808973e-01 0.000000e+00
## mileage         2.141130e+04 3.196271e-03
## tax             6.585652e+01 8.767435e-07
##
## $`3`
##                     v.test Mean in category Overall mean sd in category
## year             51.872974        2019.0232  2017.21640      0.5655294
## price            33.828421       28021.8883  21418.53580   9493.8441666
## tax              21.003053         147.0819    122.91100     12.5263288
## inconsistencies  -9.363726           0.0000      0.02960      0.0000000
## mpg             -36.397305          45.6240     53.00322      8.9126588
## mileage         -46.282125        5972.7383  23289.51910   4836.6612785
##                   Overall sd       p.value
## year            1.993281e+00  0.000000e+00
## price           1.117048e+04 7.537260e-251
## tax             6.585652e+01  6.150355e-98
## inconsistencies 1.808973e-01  7.697348e-21
## mpg             1.160193e+01 4.696054e-290
## mileage         2.141130e+04  0.000000e+00
##
## $`4`
##                    v.test Mean in category Overall mean sd in category
## mpg             14.933185         57.46411     53.00322       7.649678
```

```
## tax              10.861718        141.32873    122.91100      21.249854
## inconsistencies -6.355075           0.00000      0.02960       0.000000
## price           -8.122821      19082.29508  21418.53580    5350.982501
##                   Overall sd      p.value
## mpg             1.160193e+01 2.004571e-50
## tax             6.585652e+01 1.754185e-27
## inconsistencies 1.808973e-01 2.083243e-10
## price           1.117048e+04 4.554722e-16
##
## $`5`
##                     v.test Mean in category Overall mean sd in category
## mpg              38.998449         65.27474     53.00322       6.247068
## mileage          25.478738      38085.43966  23289.51910   17359.708872
## inconsistencies  -6.033072          0.00000      0.02960       0.000000
## price           -28.419255      12808.48644  21418.53580    3657.472232
## year            -29.611931       2015.61553   2017.21640       1.120847
## tax             -56.152035         22.61459    122.91100      17.143188
##                   Overall sd       p.value
## mpg             1.160193e+01  0.000000e+00
## mileage         2.141130e+04 3.392108e-143
## inconsistencies 1.808973e-01  1.608721e-09
## price           1.117048e+04 1.169358e-177
## year            1.993281e+00 1.049165e-192
## tax             6.585652e+01  0.000000e+00
##
## $`6`
##                     v.test Mean in category Overall mean sd in category
## inconsistencies  23.763343     3.597561e-01      0.02960   5.166401e-01
## mileage          21.715996     5.900055e+04  23289.51910   4.074805e+04
## tax               9.767916     1.723171e+02    122.91100   1.373457e+02
## price             9.074843     2.920411e+04  21418.53580   3.231820e+04
## mpg              -9.021451     4.496451e+01     53.00322   1.745252e+01
## year            -16.588688     2.014677e+03   2017.21640   3.680761e+00
##                   Overall sd       p.value
## inconsistencies 1.808973e-01 7.998070e-125
## mileage         2.141130e+04 1.448700e-104
## tax             6.585652e+01  1.545994e-22
## price           1.117048e+04  1.138415e-19
## mpg             1.160193e+01  1.856145e-19
## year            1.993281e+00  8.413887e-62
```

We want to know now which variables are associated with the quantitative variables.

- Cluster 1
  - We can see that for cluster 1 we have cars that have a lot of mileage (average mean of class of 47044 vs 23289 overall mean) and cars with high taxes (60 units over the overall mean). The cluster contains cheaper cars than the global mean (4230 units cheaper on average). It also contains cars with lower mpg and cars that were sold on before the average selling year of the dataset. Finally the cluster does not cocntain any obervation with an inconsistency.
- Cluster 2

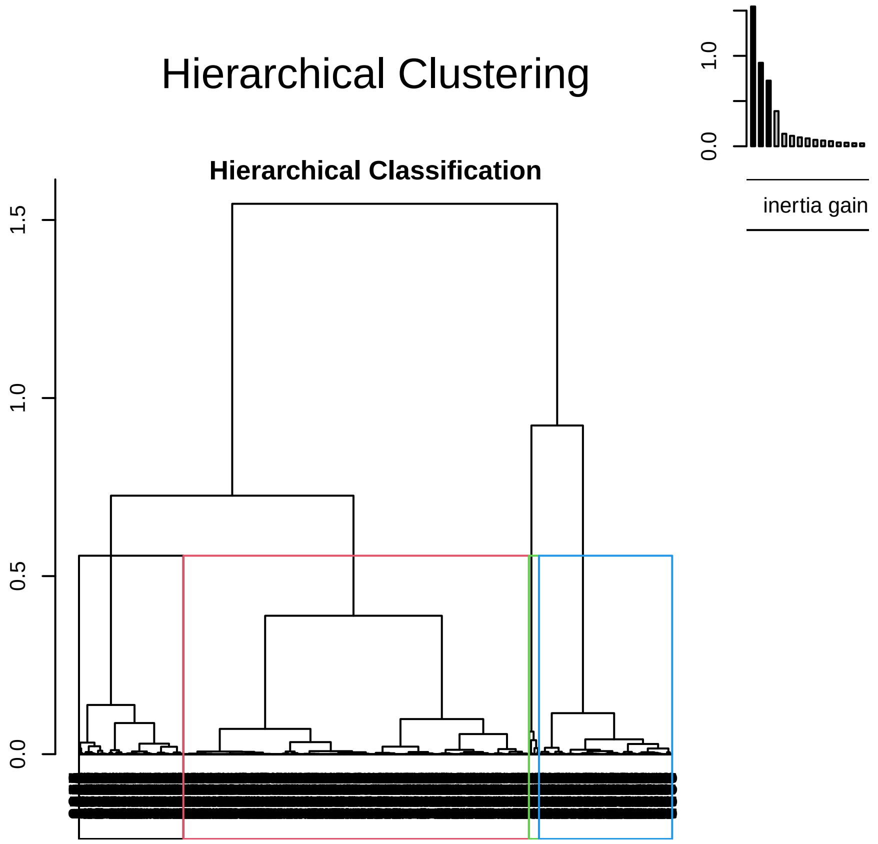
- For cluster 2, we have cars with a higher mileage than the global average (30203 mean in cluster 2 vs 23289 global mean). We also have cars that have lower taxes. This cluster groups almost all observations with inconsistencies. Some other observations with inconsistencies are in cluster 6.

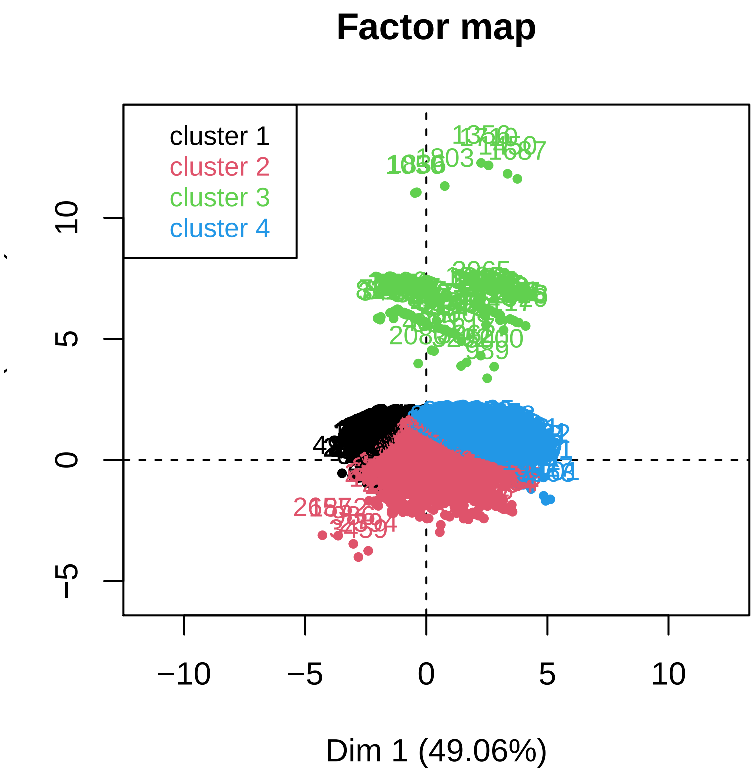
- Cluster 3
  - This cluster gropus cars that were sold in 2019 on average. The cars also have a higher average price (6603 units over the global mean). In addition, these cars have lower taxes. Finally, we do not have any observation with an inconsistency.
- Cluster 4
  - In cluster 4 we can't see a great difference between global mean and the mean of the cluster but we can highlight that cars in cluster 4 have more mpg and taxes than in the global mean. The global mean of tax is 122.91 and its mean in cluster 5 is 141.33, and the global mean of mpg is 53 and its mean in cluster 5 is 57.46.
- Cluster 5
  - In cluster 5 we can highlight that the price of the cars is very cheap, the mean of the cluster is 12808.48 while the global mean is almost double: 21418.53. It is also very remarkable that the taxes are very low, the mean of the cluster is 22.61 while the global mean is almost double: 122.91. This variables have lower mean that the global but we also have variables that highlight for having a higher mean, like mpg and mileage. The global mean of mpg is 53 and its mean in cluster 5 is 65.27, and the global mean of mileage is 23289.52 and its mean in cluster 5 is 38085.44.
- Cluster 6
  - In cluster 6 we can see that the inconsistencies are higher that in the overall mean, as well as the mileage, tax and price: Mean in category Overall mean sd in category inconsistencies 0.3597561 0.02960 mileage 59000.55 23289.51910 tax 172.3171 122.91100 price 29204.11 21418.53580

The variable mean mpg in cluster 6 (44.96) is lower than the overall mean (53), and the cars in cluster 6 are older because the mean of the variable year is 2014 and in the overall mean is 2017.

4.4 HCPC computation and visualization

```
# dis<-dist(res.pca$ind$coord[,1:2])
res.hcpc <- HCPC(res.pca, nb.clust = -1, proba = 0.01)
```

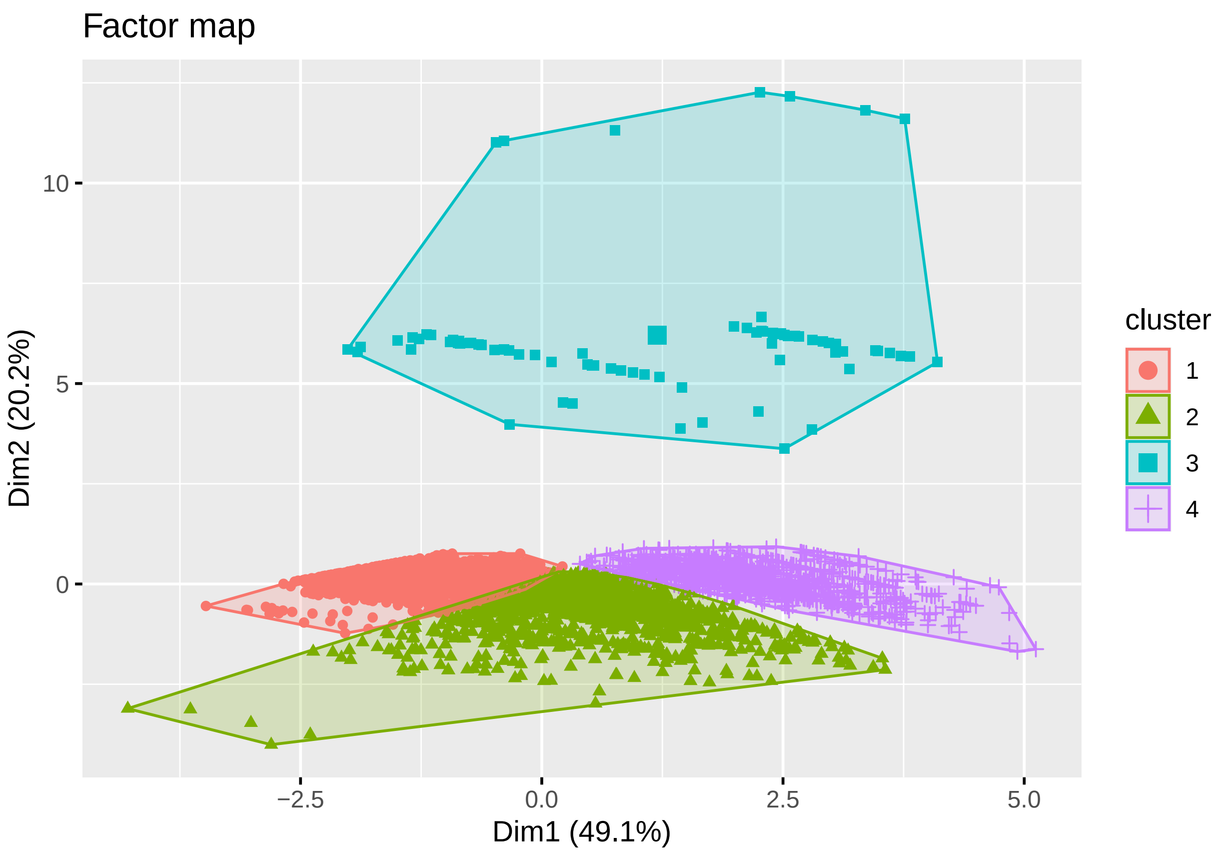





```
100 * (res.hcpc$call$t$within[1] - res.hcpc$call$t$within[1:10])/(res.hcpc$call$t$within[1])
```

```
## [1]  0.00000 30.90886 49.36759 63.88594 71.66041 74.42473 76.72917 78.70069
## [9] 80.44950 81.86984
```

```
# Individuals facor map
fviz_cluster(res.hcpc, geom = "point", main = "Factor map")
```



4.4.1 Gain in inertia (in %)

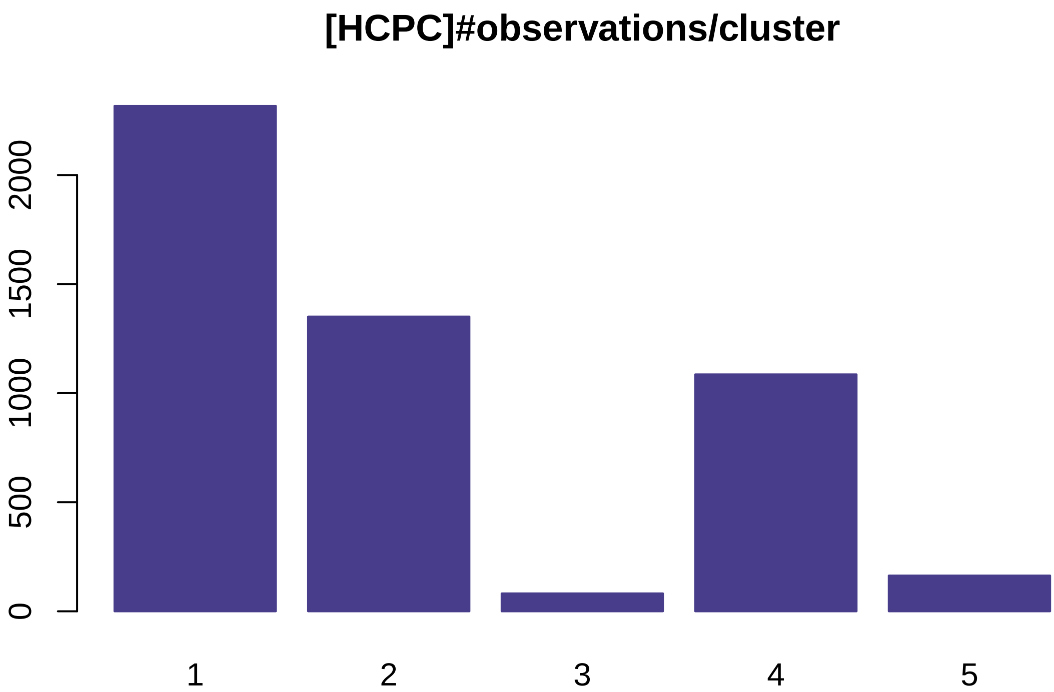
After executing HCPC we get a retained inertia of the 63.88594%

```
100 * (res.hcpc$call$t$within[1] - res.hcpc$call$t$within[4])/(res.hcpc$call$t$within[1])
```

```
## [1] 63.88594
```

4.5 Profiling of clusters

```
df$HCPC_clust <- 0
k = 4
df[-llmout, "HCPC_clust"] <- res.hcpc$data.clust$clust
df[llmout, "HCPC_clust"] <- k + 1
df$HCPC_clust <- factor(df$HCPC_clust)
# observations that are multivariant outliers will be put in cluster
# 6
barplot(table(df$HCPC_clust), col = "darkslateblue", border = "darkslateblue",
    main = "[HCPC]#observations/cluster")
```



We proceed to explain the data obtained.

4.5.1 Description of clusters by categorical variables

```
res.hcpc$desc.var$test.chi2
```

```
##                    p.value df
## fuelType      0.000000e+00  6
## engineSize   4.283295e-208 63
## transmission 2.011091e-116  6
## manufacturer  3.742777e-25  9
```

We can see the intensity of the variables, in our case the variables that affect more to the clustering are **fuelType**, **engineSize**,**transmission** and **manufacturer** because are those ones with the smallest p.value. We excluded from this test the factor variables that resulted from the grouping of the quantiles of the numerical variables because we will analyze these numerical variables later on. This way we reduce redundant information.

Next, we want to see for each cluster which are the categories that characterize them.

```
res.hcpc$desc.var$category
```

```
## $`1`
##                                  Cla/Mod     Mod/Cla     Global      p.value
## transmission=f.Trans-SemiAuto  60.052632 49.24471299 39.2886683 2.809901e-42
## engineSize=engineSize_1.5      71.538462 16.05524385 10.7526882 8.693997e-31
## fuelType=f.Fuel-Petrol         57.605985 49.84894260 41.4598842 5.790668e-30
```

```
## engineSize=engineSize_1.3     100.000000  3.19378507  1.5301902 1.215151e-24
## engineSize=engineSize_2        51.253071 45.01510574 42.0802316 7.390293e-05
## engineSize=engineSize_2.9     100.000000  0.51791109  0.2481390 1.441509e-04
## engineSize=engineSize_3        53.817505 12.47302546 11.1042184 3.712360e-03
## manufacturer=Mercedes          51.319876 28.52826931 26.6335815 4.284692e-03
## engineSize=engineSize_2.3     100.000000  0.30211480  0.1447477 5.767997e-03
## engineSize=engineSize_1.8       5.263158  0.04315926  0.3928867 7.874475e-05
## fuelType=f.Fuel-Hybrid         14.285714  0.38843332  1.3027295 1.872576e-08
## fuelType=f.Fuel-Diesel         41.654624 49.76262408 57.2373863 6.394255e-24
## engineSize=engineSize_1.2       3.174603  0.17263703  2.6054591 4.096882e-30
## engineSize=engineSize_1.4      16.913947  2.46007769  6.9685691 6.922273e-35
## engineSize=engineSize_2.1      16.666667  2.80535175  8.0645161 3.175807e-41
## transmission=f.Trans-Manual    33.610451 24.42813984 34.8221671 1.348019e-48
##                                   v.test
## transmission=f.Trans-SemiAuto  13.625883
## engineSize=engineSize_1.5      11.535932
## fuelType=f.Fuel-Petrol         11.371614
## engineSize=engineSize_1.3      10.247443
## engineSize=engineSize_2         3.963358
## engineSize=engineSize_2.9       3.800935
## engineSize=engineSize_3         2.901622
## manufacturer=Mercedes           2.856404
## engineSize=engineSize_2.3       2.760687
## engineSize=engineSize_1.8      -3.948189
## fuelType=f.Fuel-Hybrid         -5.623379
## fuelType=f.Fuel-Diesel        -10.085647
## engineSize=engineSize_1.2     -11.401773
## engineSize=engineSize_1.4     -12.321696
## engineSize=engineSize_2.1     -13.447704
## transmission=f.Trans-Manual   -14.649957
##
## $`2`
##                                 Cla/Mod     Mod/Cla     Global      p.value
## engineSize=engineSize_3        45.810056 18.20873427 11.1042184 6.079019e-21
## fuelType=f.Fuel-Diesel         31.864162 65.28497409 57.2373863 1.379844e-12
## engineSize=engineSize_2.1      41.538462 11.99111769  8.0645161 1.712592e-09
## manufacturer=BMW               35.380117 26.86898594 21.2158809 3.983416e-09
## transmission=f.Trans-Automatic 33.067093 30.64396743 25.8891646 3.336743e-06
## engineSize=engineSize_1.8      63.157895  0.88823094  0.3928867 1.681687e-03
## transmission=f.Trans-Manual    25.296912 31.53219837 34.8221671 2.687036e-03
## engineSize=engineSize_1.6      16.666667  4.36713546  7.3200993 2.903645e-07
## fuelType=f.Fuel-Hybrid          1.587302  0.07401925  1.3027295 2.492455e-08
## fuelType=f.Fuel-Petrol         23.341646 34.64100666 41.4598842 1.674917e-09
## engineSize=engineSize_1.3       0.000000  0.00000000  1.5301902 2.375938e-11
## manufacturer=VW                21.084746 23.01998520 30.5004136 8.753091e-13
## engineSize=engineSize_1        10.483871  2.88675056  7.6923077 2.066496e-17
## engineSize=engineSize_1.5      10.769231  4.14507772 10.7526882 1.502714e-23
##                                    v.test
## engineSize=engineSize_3          9.388622
## fuelType=f.Fuel-Diesel           7.086060
## engineSize=engineSize_2.1        6.022957
## manufacturer=BMW                 5.884881
## transmission=f.Trans-Automatic   4.648922
```

```
## engineSize=engineSize_1.8        3.141354
## transmission=f.Trans-Manual     -3.001443
## engineSize=engineSize_1.6       -5.129596
## fuelType=f.Fuel-Hybrid          -5.573797
## fuelType=f.Fuel-Petrol          -6.026555
## engineSize=engineSize_1.3       -6.680829
## manufacturer=VW                 -7.148814
## engineSize=engineSize_1         -8.489993
## engineSize=engineSize_1.5      -10.001391
##
## $`3`
##                                   Cla/Mod   Mod/Cla    Global      p.value
## fuelType=f.Fuel-Hybrid         74.6031746 57.317073  1.302730 1.271646e-76
## transmission=f.Trans-Automatic  3.7539936 57.317073 25.889165 1.501176e-09
## engineSize=engineSize_2         2.9484029 73.170732 42.080232 1.092404e-08
## manufacturer=BMW                3.3138402 41.463415 21.215881 3.155262e-05
## transmission=f.Trans-SemiAuto   1.0526316 24.390244 39.288668 4.449444e-03
## engineSize=engineSize_1         0.0000000  0.000000  7.692308 1.331555e-03
## transmission=f.Trans-Manual     0.8907363 18.292683 34.822167 9.839476e-04
## engineSize=engineSize_3         0.1862197  1.219512 11.104218 7.334224e-04
## fuelType=f.Fuel-Diesel          0.9031792 30.487805 57.237386 9.604332e-07
## fuelType=f.Fuel-Petrol          0.4987531 12.195122 41.459884 6.609818e-09
##                                  v.test
## fuelType=f.Fuel-Hybrid         18.526105
## transmission=f.Trans-Automatic  6.044238
## engineSize=engineSize_2         5.715720
## manufacturer=BMW                4.161961
## transmission=f.Trans-SemiAuto  -2.844405
## engineSize=engineSize_1        -3.209091
## transmission=f.Trans-Manual    -3.295077
## engineSize=engineSize_3        -3.376770
## fuelType=f.Fuel-Diesel         -4.899577
## fuelType=f.Fuel-Petrol         -5.800549
##
## $`4`
##                                   Cla/Mod     Mod/Cla     Global      p.value
## transmission=f.Trans-Manual    40.2019002 62.33885820 34.8221671 1.344284e-99
## engineSize=engineSize_1.4      48.0712166 14.91712707  6.9685691 5.383301e-27
## engineSize=engineSize_1.2      61.9047619  7.18232044  2.6054591 6.696807e-22
## engineSize=engineSize_2.1      41.2820513 14.82504604  8.0645161 3.356361e-18
## manufacturer=VW                29.4237288 39.96316759 30.5004136 3.988695e-14
## engineSize=engineSize_1        35.7526882 12.24677716  7.6923077 1.211898e-09
## fuelType=f.Fuel-Diesel         25.5780347 65.19337017 57.2373863 1.374542e-09
## engineSize=engineSize_1.6      35.8757062 11.69429098  7.3200993 2.390485e-09
## fuelType=f.Fuel-Hybrid          9.5238095  0.55248619  1.3027295 8.361967e-03
## manufacturer=Mercedes          19.5652174 23.20441989 26.6335815 3.412570e-03
## engineSize=engineSize_4         0.0000000  0.00000000  0.4755997 2.837469e-03
## engineSize=engineSize_1.5      16.9230769  8.10313076 10.7526882 1.036074e-03
## fuelType=f.Fuel-Petrol         18.5536160 34.25414365 41.4598842 3.571846e-08
## engineSize=engineSize_1.3       0.0000000  0.00000000  1.5301902 5.691891e-09
## manufacturer=BMW               14.9122807 14.08839779 21.2158809 1.515598e-11
## transmission=f.Trans-Automatic 14.4568690 16.66666667 25.8891646 3.830521e-16
## engineSize=engineSize_2        16.2653563 30.47882136 42.0802316 5.369623e-19
```

```
## transmission=f.Trans-SemiAuto  12.0000000 20.99447514 39.2886683 1.877483e-47
## engineSize=engineSize_3         0.1862197  0.09208103 11.1042184 7.170500e-62
##                                    v.test
## transmission=f.Trans-Manual     21.183900
## engineSize=engineSize_1.4       10.758858
## engineSize=engineSize_1.2        9.618239
## engineSize=engineSize_2.1        8.698724
## manufacturer=VW                  7.561362
## engineSize=engineSize_1          6.078662
## fuelType=f.Fuel-Diesel           6.058433
## engineSize=engineSize_1.6        5.968770
## fuelType=f.Fuel-Hybrid          -2.637094
## manufacturer=Mercedes           -2.927903
## engineSize=engineSize_4         -2.984818
## engineSize=engineSize_1.5       -3.280543
## fuelType=f.Fuel-Petrol          -5.510811
## engineSize=engineSize_1.3       -5.825567
## manufacturer=BMW                -6.746399
## transmission=f.Trans-Automatic  -8.143803
## engineSize=engineSize_2         -8.904357
## transmission=f.Trans-SemiAuto  -14.469899
## engineSize=engineSize_3        -16.598290
```

- Cluster 1
  - The first thing we can notice is that cluster 1 contains the 60.052632% of all the cars with semi-automatic transmission. The 49.84894260% of its cars use petrol while the global percenatge of cars that use petrol is 41.4598842%, that means that is overrepresented.
- Cluster 2
  - In cluster 2 we can see that The 65.28497409% of its cars use diesel while the global percenatge of cars that use petrol is 57.2373863%, that means that is overrepresented. The manufacturer BMW as well as cars with automatic transmission are overrepresented.
- Cluster 3
  - Cluster 3 contains 74.6% of all hybrid cars from the data set. On average 1.3% of the cars use hybrid fuel, but in cluster 3 hybrid cars are overrepresented (57.32%). Automatic cars are also overrepresented in the class with a 57.3% in comparison to a 25.8% global mean. BMW cars are also overrepresented in cluster 3 with a 41.46% in comparison to a 21.2% global mean.
- Cluster 4
  - Cluster 4 contains 40.2% of all manual cars from the data set. On average 34.8% of the cars are manual, but in cluster 4 manual cars are overrepresented (62.3%). Automatic cars are also overrepresented in the class with a 57.3% in comparison to a 25.8% global mean. VW cars are overrepresented where on the other hand, Mercedes and BMW cars are underrepresented.

We now proceed to see the quantitative variables that characterizes the clusters. We can see in the output from **res.hcpc**$desc.var$**quanti.var** all the numeric variables that characterize the clusters. From a more detailed look, variables **years_after_sell**, **mileage**, **tax** and **inconsistencies** maintain a strong relation with the cluster number.

```r
res.hcpc$desc.var$quanti.var
```

```
##                        Eta2 P-value
## price              0.3842227       0
## mileage            0.5492157       0
## tax                0.7965344       0
## mpg                0.3694009       0
## years_after_sell 0.6742560       0
## inconsistencies  0.9368360       0
```

```r
res.hcpc$desc.var$quanti
```

```
## $`1`
##                     v.test Mean in category Overall mean sd in category
## price             40.932240     27024.632715 2.115451e+04   9381.0573588
## tax               27.830312       146.821321 1.212355e+02     11.3658133
## inconsistencies   -8.477275         0.000000 1.840364e-02      0.0000000
## mpg              -34.469252        47.460725 5.327583e+01      9.8840356
## mileage          -51.377950      7199.759171 2.207848e+04   5763.4762452
## years_after_sell -56.679449         3.125162 4.697477e+00      0.6832679
##                    Overall sd       p.value
## price            9.563766e+03  0.000000e+00
## tax              6.130954e+01 1.864767e-170
## inconsistencies  1.447753e-01  2.305291e-17
## mpg              1.125053e+01 2.318103e-260
## mileage          1.931238e+04  0.000000e+00
## years_after_sell 1.849955e+00  0.000000e+00
##
## $`2`
##                     v.test Mean in category Overall mean sd in category
## years_after_sell  30.932744         6.019245 4.697477e+00       1.407261
## mileage           28.774442     34914.136195 2.207848e+04   16419.405018
## tax               27.227152       159.792746 1.212355e+02      45.637129
## inconsistencies   -5.503432         0.000000 1.840364e-02       0.000000
## price            -14.955909     17850.682457 2.115451e+04    5643.571818
##                    Overall sd       p.value
## years_after_sell 1.849955e+00 4.335711e-210
## mileage          1.931238e+04 4.480561e-182
## tax              6.130954e+01 3.099087e-163
## inconsistencies  1.447753e-01  3.724674e-08
## price            9.563766e+03  1.425219e-50
##
## $`3`
##                   v.test Mean in category Overall mean sd in category
## inconsistencies 67.302317         1.085366 1.840364e-02   2.794253e-01
## mileage          3.842198     30203.785906 2.207848e+04   2.117783e+04
## tax             -5.034070        87.439024 1.212355e+02   7.656594e+01
##                   Overall sd      p.value
## inconsistencies 1.447753e-01 0.000000e+00
## mileage         1.931238e+04 1.219376e-04
## tax             6.130954e+01 4.801731e-07
```

```
##
## $`4`
##                      v.test Mean in category Overall mean sd in category
## mpg                38.66466         64.90074 5.327583e+01        6.399705
## years_after_sell  33.92697          6.37477 4.697477e+00        1.115960
## mileage            29.37915      37241.21087 2.207848e+04    17565.981586
## inconsistencies   -4.75671          0.00000 1.840364e-02        0.000000
## price             -32.56909      12830.41344 2.115451e+04     3635.221807
## tax               -61.03467         21.23389 1.212355e+02        9.820985
##                    Overall sd       p.value
## mpg               1.125053e+01  0.000000e+00
## years_after_sell 1.849955e+00 2.666666e-252
## mileage           1.931238e+04 1.014216e-189
## inconsistencies   1.447753e-01  1.967740e-06
## price             9.563766e+03 1.123883e-232
## tax               6.130954e+01  0.000000e+00
```

We want to know now which variables are associated with the quantitative variables.

- Cluster 1
  - Cluster 1 groups cars with higher prices than the overall mean. The mean price in this first cluster is 27024 while the overall mean sits at 21154. It also contains cars that pay higher taxes and cars that have higher fuel consumption (low mpg). Finally, the cluster contains cars with a very low mileage (7200 on avg) compared to the global mean of 22000 and the cars where sold recently, as they average year after sell is 3 for the cluster and 4.7 for the global mean.
- Cluster 2 +Cluster 2 contains cars that where sold earlier than other cars, as they average year after sell is 6 for the cluster and 4.7 for the global mean. It also contains cars with a higher mileage and higher tax. Finally it contains cheaper cars than the global mean, 17850 dollars for the cluster vs 21100 for the overall mean.
- Cluster 3
  - In this cluster we can see that the variable mileage is higher than the overall mean while the taxes are lower. This can indicate us that they are old or very used cars. The most important trait for this cluster is that it groups observations that only have inconsistencies. The price is not highlighted so it must be arround the global mean.
- Cluster 4
  - Cars in cluster 4 have the biggest mileage mean, being it 37241.21087. Its cars spend also less fuel than the overall mean. It is very important to highlight that the cars in cluster 4 are very cheap and have very low taxes.

5 CA analysis

CA analysis for your data should contain your factor version of the numeric target (previous) in K= 7 (maximum 10) levels and 2 factors.

5.1 CA: f.price vs f.tax

We start the CA with the creation of a contingency table and a Pearson's chi2 test to test for independence.

```r
tt <- table(df[, c("f.price", "f.tax")])
tt
```

```
##                      f.tax
## f.price               f.tax-(125,145] f.tax-(145,155] f.tax-(155,580]
##   super cheap                     150              63              90
##   cheap                           276             105              91
##   expensive                       560             123             129
##   very expensive                  763             107             102
##   extremely expensive             819             109              66
##                      f.tax
## f.price               f.tax-[0,125]
##   super cheap                   697
##   cheap                         530
##   expensive                     187
##   very expensive                 32
##   extremely expensive             1
```

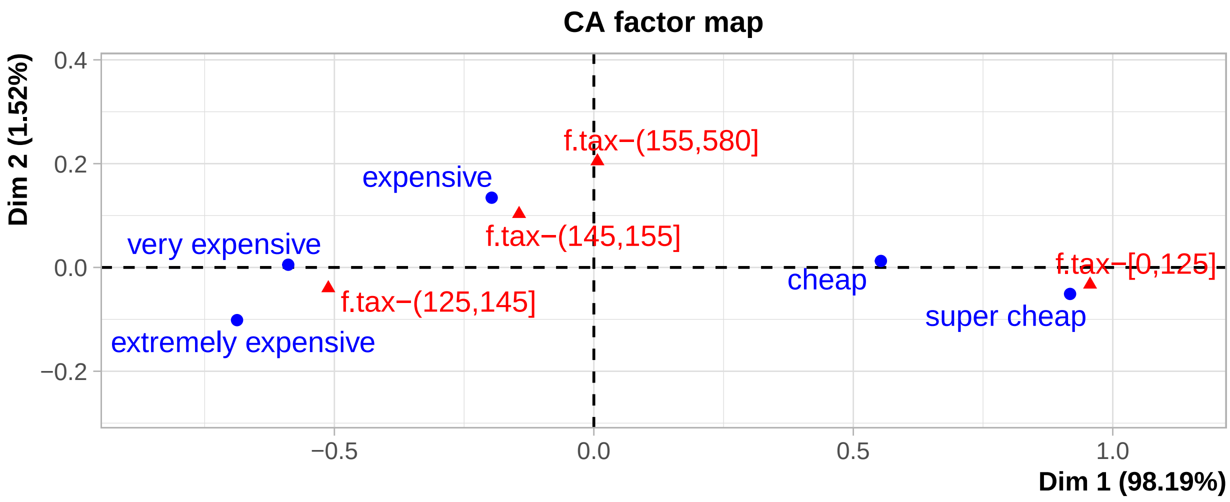
```r
chisq.test(tt)  #to see if variables are independents. H0: Variables are independent
```

```
##
##  Pearson's Chi-squared test
##
## data:  tt
## X-squared = 2043.3, df = 12, p-value < 2.2e-16
```

We get a p-value lower than 0.05 so we can reject the H0. In our sample, the row and the column variables are statistically significantly associated.

We are now going to take a look to the simple correspondences.

```r
res.ca <- CA(tt)
```

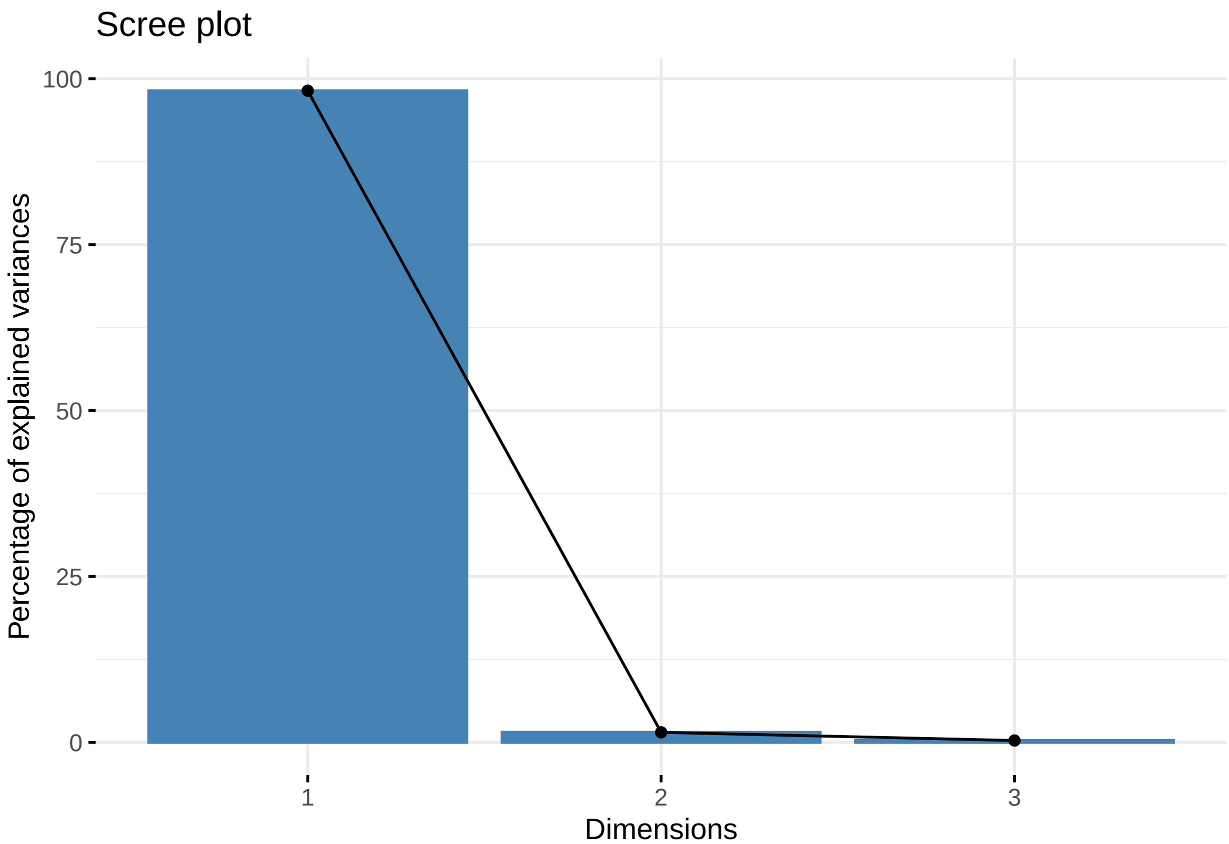



```
summary(res.ca, dig = 2)
```

```
##
## Call:
## CA(X = tt)
##
## The chi square of independence between the two variables is equal to 2043.265 (p-value =
##
## Eigenvalues
##                        Dim.1   Dim.2   Dim.3
## Variance               0.401   0.006   0.001
## % of var.             98.186   1.521   0.293
## Cumulative % of var.  98.186  99.707 100.000
##
## Rows
##                      Iner*1000     Dim.1     ctr    cos2     Dim.2     ctr
## super cheap         |  169.271 |   0.918  41.982   0.995 |  -0.051   8.379
## cheap               |   61.934 |   0.553  15.284   0.990 |   0.012   0.502
## expensive           |   11.350 |  -0.197   1.929   0.682 |   0.134  58.056
## very expensive      |   69.874 |  -0.589  17.356   0.997 |   0.005   0.088
## extremely expensive |   96.223 |  -0.688  23.448   0.978 |  -0.101  32.975
##                       cos2     Dim.3     ctr    cos2
## super cheap          0.003 |  -0.039  25.092   0.002 |
## cheap                0.001 |   0.054  48.214   0.009 |
## expensive            0.318 |  -0.002   0.095   0.000 |
## very expensive       0.000 |  -0.034  19.053   0.003 |
## extremely expensive  0.021 |   0.021   7.546   0.001 |
##
## Columns
```

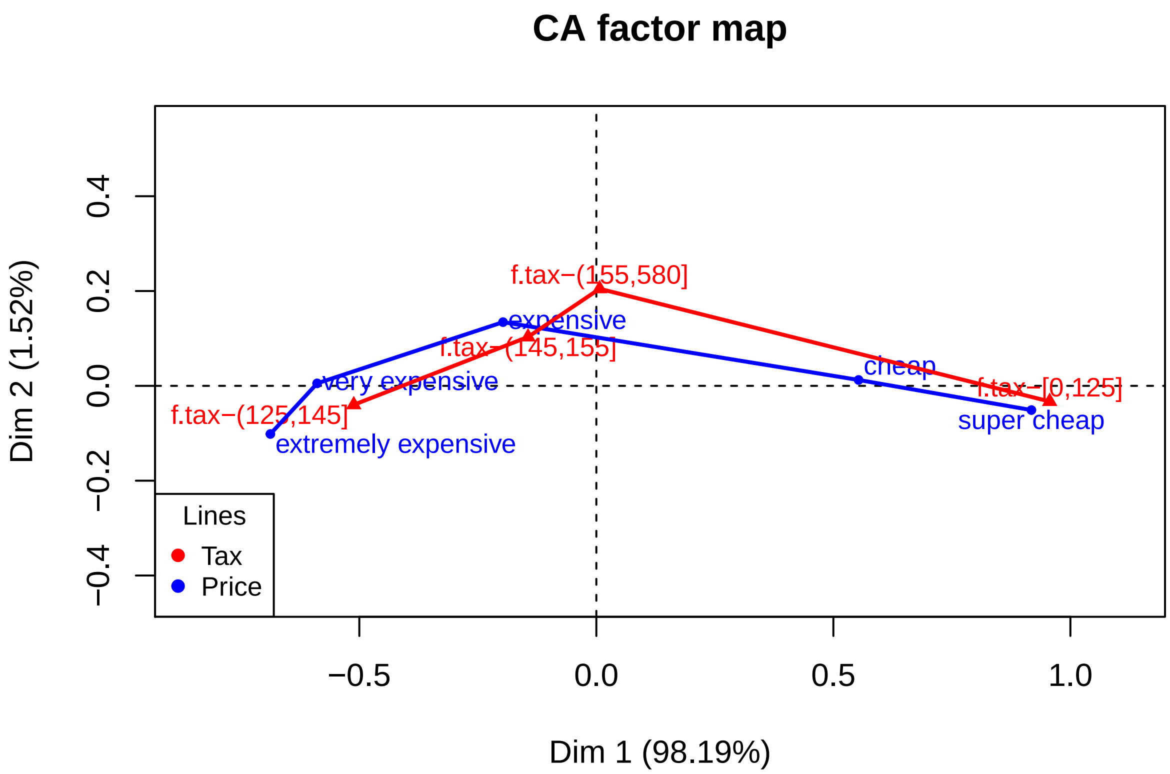
```
##                    Iner*1000    Dim.1    ctr   cos2    Dim.2    ctr
## f.tax-(125,145]  |   135.317 | -0.512 33.515  0.994 | -0.040 13.156
## f.tax-(145,155]  |     4.044 | -0.144  0.525  0.520 |  0.103 17.352
## f.tax-(155,580]  |     4.322 |  0.007  0.001  0.001 |  0.205 64.441
## f.tax-[0,125]    |   264.970 |  0.956 65.959  0.999 | -0.033  5.051
##                    cos2    Dim.3    ctr   cos2
## f.tax-(125,145]   0.006 | -0.007  1.969  0.000 |
## f.tax-(145,155]   0.267 |  0.092 71.984  0.213 |
## f.tax-(155,580]   0.927 | -0.057 25.998  0.072 |
## f.tax-[0,125]     0.001 | -0.001  0.049  0.000 |
```

```
fviz_eig(res.ca)  #Visualize the eigenvalues
```



According to the Kaiser criteria, we have to consider the first dimension. In this correspondence analysis almost all the variance is explained in the 1st dimension (96.74%).

```
plot(res.ca, cex = 0.8, graph.type = "classic")
lines(res.ca$row$coord[, 1], res.ca$row$coord[, 2], col = "blue", lwd = 2)
lines(res.ca$col$coord[, 1], res.ca$col$coord[, 2], col = "red", lwd = 2)
legend("bottomleft", title = "Lines", legend = c("Tax", "Price"), col = c("red",
    "blue"), pch = 19, cex = 0.8)
```



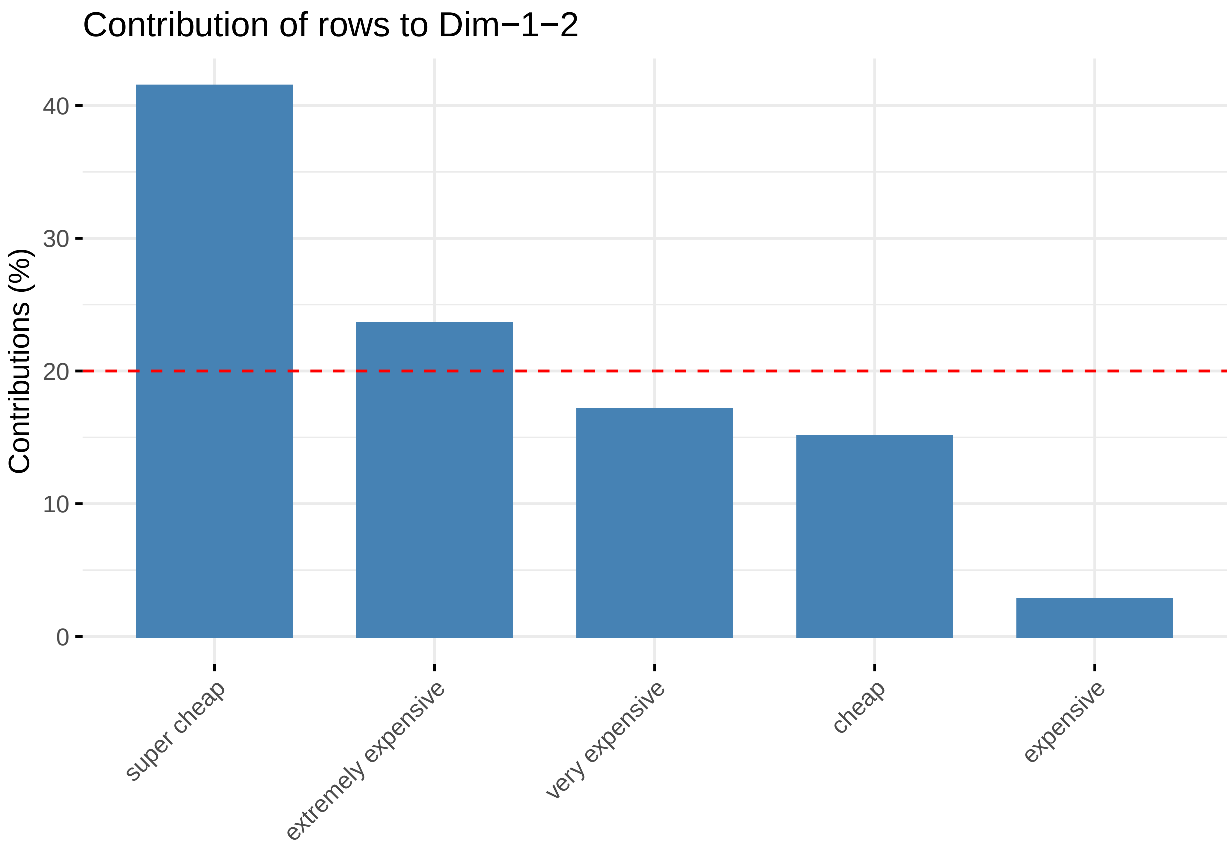
This graph shows that with mildly high taxes come cars with prices that are average or high. With cheap cars come very low taxes. Very high taxes come with cars that are between expensive and cheap. With extremely expensive cars we get mildly high taxes.

5.1.1 Contribution of rows to the dimensions

```
res.ca$row$contrib
```

```
##                          Dim 1       Dim 2       Dim 3
## super cheap         41.982346  8.37876423 25.09239026
## cheap               15.284268  0.50187751 48.21402393
## expensive            1.928968 58.05563985  0.09485374
## very expensive      17.356309  0.08838777 19.05263010
## extremely expensive 23.448109 32.97533064  7.54610196
```

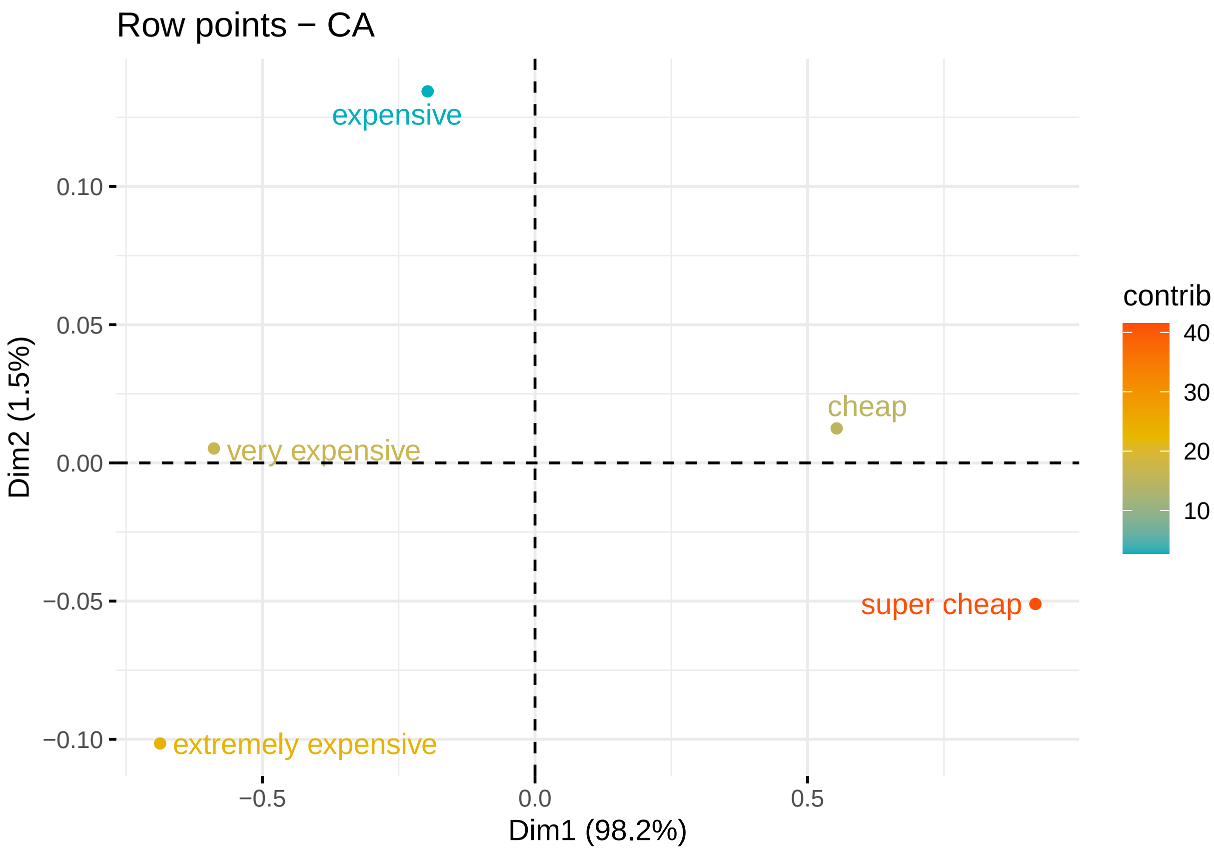
```
fviz_contrib(res.ca, choice = "row", axes = 1:2, top = 10)
```



The row variables with the larger value, contribute the most to the definition of the dimensions. Rows that contribute the most to Dim.1 are the most important in explaining the variability in the data set. In our case, the most important row categories for price are super cheap & extremely expensive cars. Rows that do not contribute much to any dimension or that contribute to the last dimensions (Dim 2) are less important. In our case these are cheap & expensive cars.

The most important (or, contributing) row points can be highlighted on the scatter plot as follow:

```
fviz_ca_row(res.ca, col.row = "contrib", gradient.cols = c("#00AFBB", "#E7B800",
    "#FC4E07"), repel = TRUE)
```



The scatter plot gives an idea of what pole of the dimensions the row categories are actually contributing to. It is evident that the row category "super cheap" has an important contribution to the positive pole of the first dimension, while the category "extremely expensive" has a major contribution to the negative pole of the first dimension.

In other words, dimension 1 is mainly defined by the opposition of super cheap cars (positive pole), and extremely expensive cars (negative pole).

5.2 CA: f.price vs fuelType

We start the CA with the creation of a contingency table and a Pearson's chi2 test to test for independence.

```
tt <- table(df[, c("f.price", "fuelType")])
tt
```

```
##                        fuelType
## f.price                 f.Fuel-Diesel f.Fuel-Petrol f.Fuel-Hybrid
##   super cheap                     476           521             3
##   cheap                           615           369            18
##   expensive                       590           394            15
##   very expensive                  572           421            11
##   extremely expensive             607           369            19
```

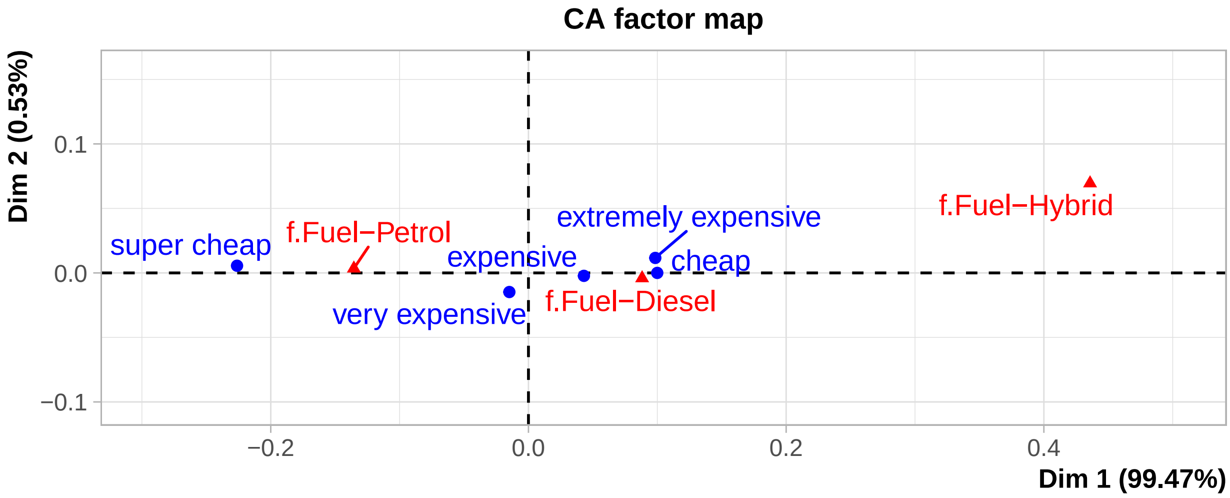
```
chisq.test(tt)  #to see if variables are independents. H0: Variables are independent
```

```
##
##  Pearson's Chi-squared test
##
## data:  tt
## X-squared = 73.263, df = 8, p-value = 1.098e-12
```

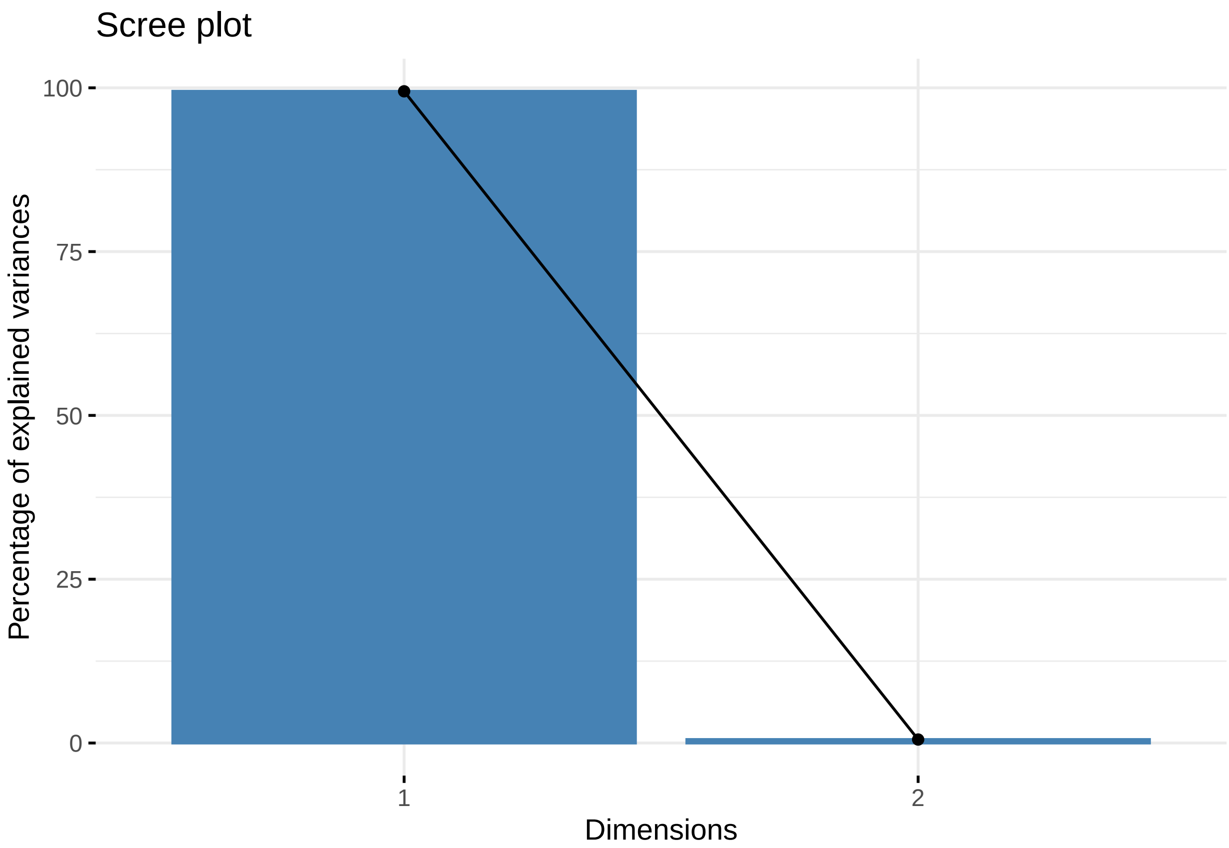
We get a p-value lower than 0.05 so we can reject the H0. In our sample, the row and the column variables are statistically significantly associated.

We are now going to take a look to the simple correspondences.

```
res.ca <- CA(tt)
```




```
fviz_eig(res.ca)  #Same outputs as PCA
```

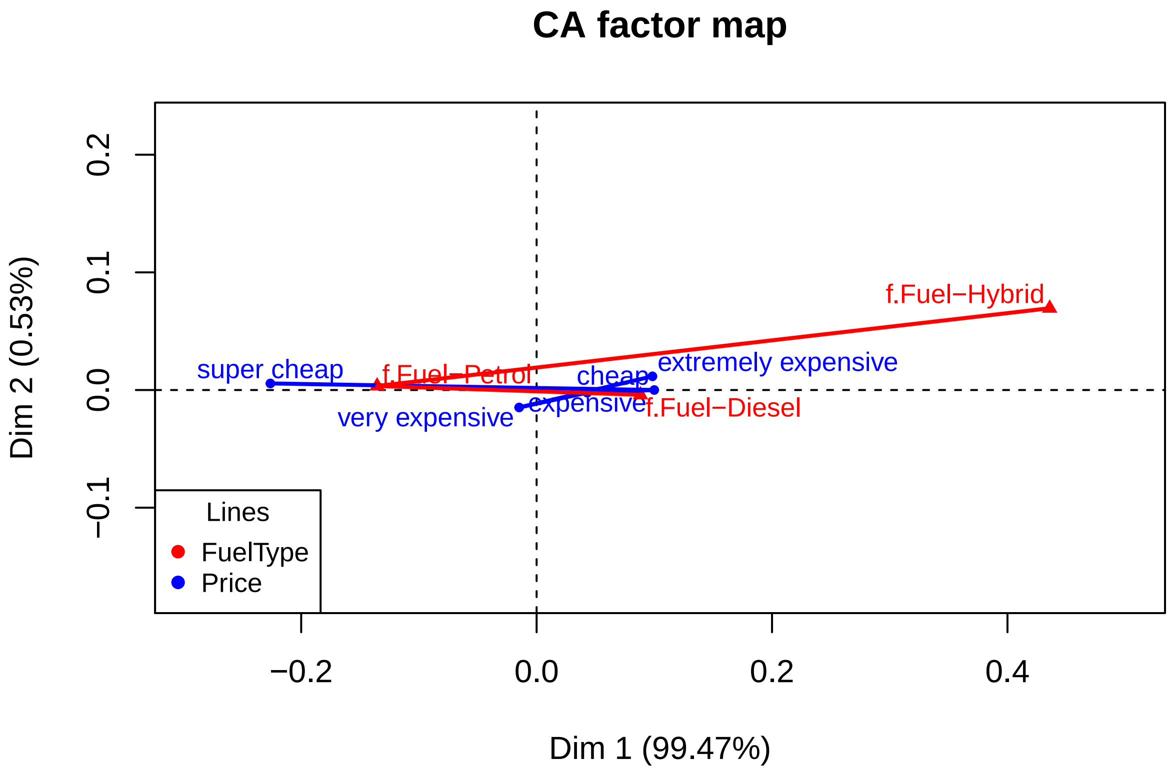



```r
summary(res.ca, dig = 2)
```

```
##
## Call:
## CA(X = tt)
##
## The chi square of independence between the two variables is equal to 73.26298 (p-value =
##
## Eigenvalues
##                        Dim.1   Dim.2
## Variance               0.015   0.000
## % of var.             99.466   0.534
## Cumulative % of var.  99.466 100.000
##
## Rows
##                       Iner*1000    Dim.1    ctr   cos2    Dim.2    ctr   cos2
## super cheap         |    10.237 | -0.226 70.195  0.999 |  0.006  7.956  0.001 |
## cheap               |     2.002 |  0.100 13.738  1.000 |  0.000  0.000  0.000 |
## expensive           |     0.371 |  0.043  2.539  0.997 | -0.002  1.324  0.003 |
## very expensive      |     0.088 | -0.015  0.304  0.501 | -0.015 56.346  0.499 |
## extremely expensive |     1.954 |  0.098 13.224  0.986 |  0.012 34.374  0.014 |
##
## Columns
##                       Iner*1000    Dim.1    ctr   cos2    Dim.2    ctr   cos2
## f.Fuel-Diesel       |     4.460 |  0.088 30.536  0.998 | -0.004 12.264  0.002 |
## f.Fuel-Petrol       |     7.621 | -0.136 52.259  0.999 |  0.003  6.261  0.001 |
## f.Fuel-Hybrid       |     2.571 |  0.436 17.206  0.975 |  0.069 81.474  0.025 |
```

According to the Kaiser criteria, we have to consider the first dimension. In this correspondence analysis almost all the variance is explained in the 1st dimension (97.1%).

```r
plot(res.ca, cex = 0.8, graph.type = "classic")
lines(res.ca$row$coord[, 1], res.ca$row$coord[, 2], col = "blue", lwd = 2)
lines(res.ca$col$coord[, 1], res.ca$col$coord[, 2], col = "red", lwd = 2)
legend("bottomleft", title = "Lines", legend = c("FuelType", "Price"),
    col = c("red", "blue"), pch = 19, cex = 0.8)
```



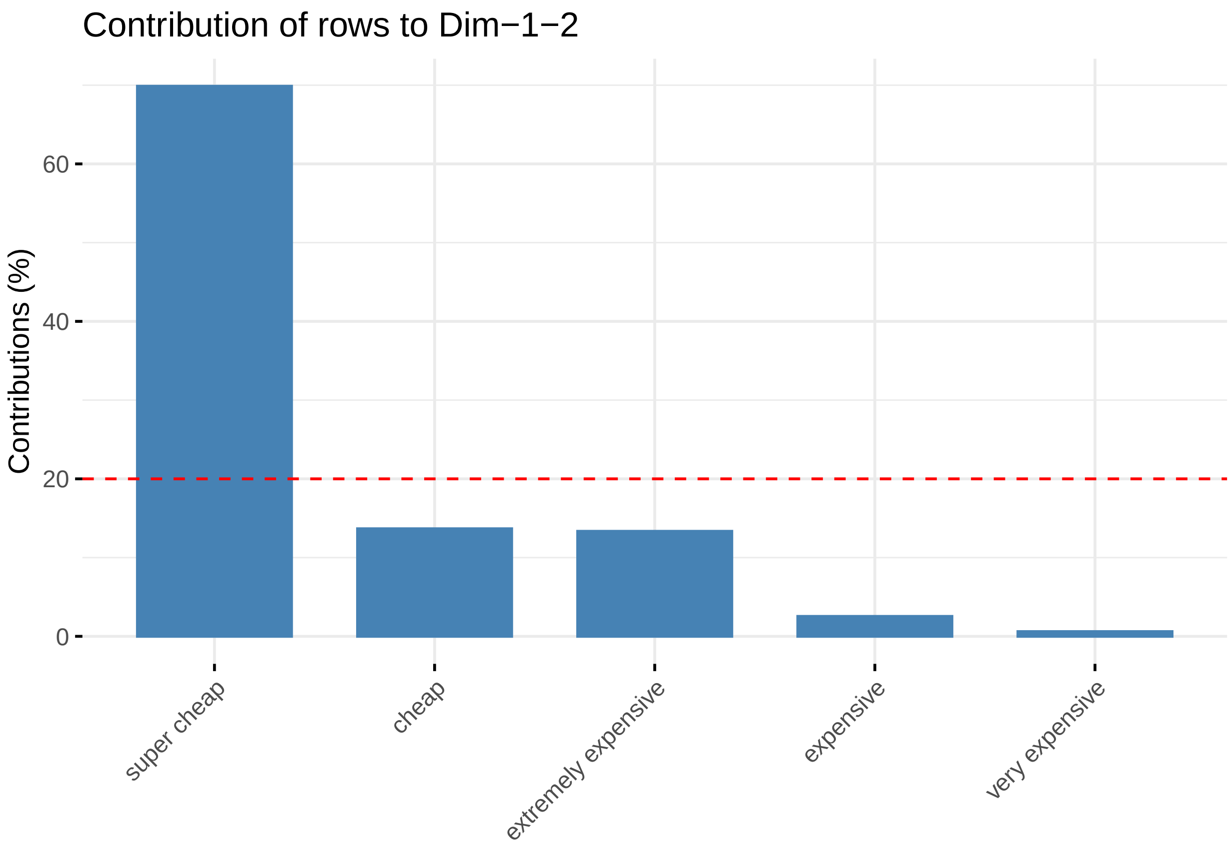
Cars that use petrol fuel are priced in an undetermined matter, because its point in the CA is placed between super cheap and very expensive cars. The same happens to diesel cars because they are placed between cheap and extremely expensive cars. Finally hybrid cars appear to be very far from all other cars. In conclusion, this graph does not provide very conclusive information to correlate the price of the cars with its type of fuel.

5.2.1 Contribution of rows to the dimensions

```r
res.ca$row$contrib
```

```
##                          Dim 1        Dim 2
## super cheap         70.1953635 7.955940e+00
## cheap               13.7377360 7.255159e-06
## expensive            2.5394905 1.324097e+00
## very expensive       0.3035202 5.634643e+01
## extremely expensive 13.2238898 3.437353e+01
```

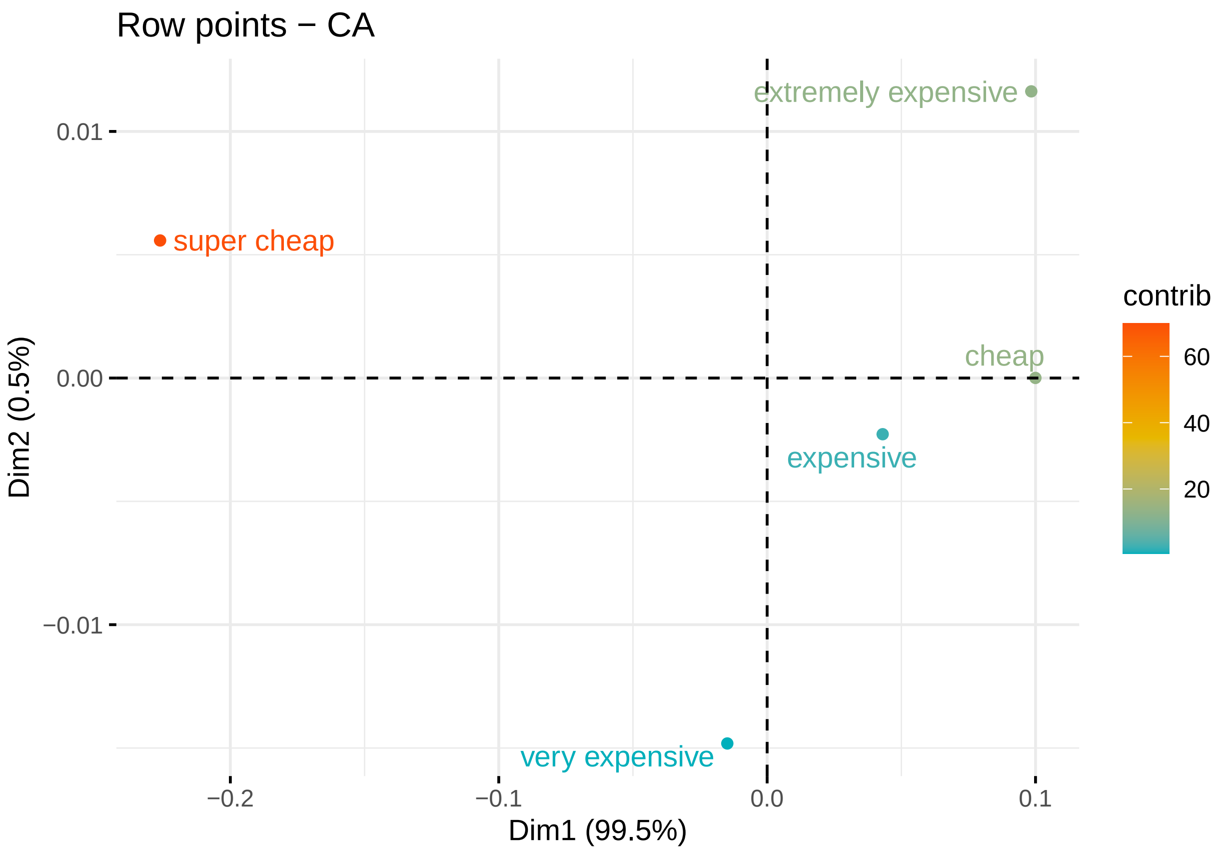
```r
fviz_contrib(res.ca, choice = "row", axes = 1:2, top = 10)
```



The row variables with the larger value, contribute the most to the definition of the dimensions. Rows that contribute the most to Dim.1 are the most important in explaining the variability in the data set. In our case, the most important row categories for price are super cheap cars & cheap cars. Rows that do not contribute much to any dimension or that contribute to the last dimensions (Dim 2) are less important. In our case these are expensive & very expensive cars.

The most important (or, contributing) row points can be highlighted on the scatter plot as follow:

```
fviz_ca_row(res.ca, col.row = "contrib", gradient.cols = c("#00AFBB", "#E7B800",
    "#FC4E07"), repel = TRUE)
```



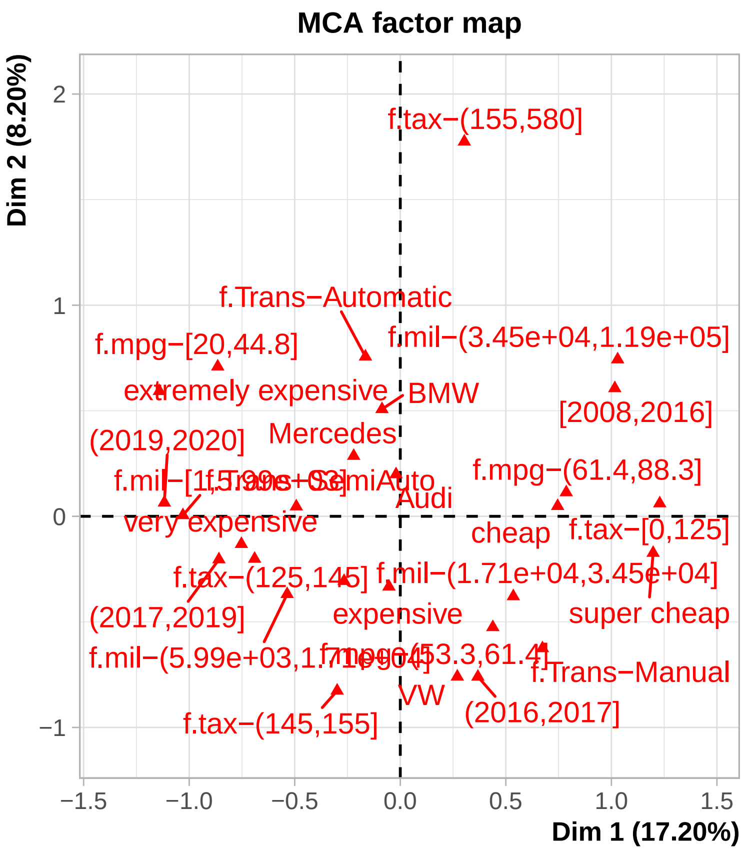
The scatter plot gives an idea of what pole of the dimensions the row categories are actually contributing to. It is evident that the row category "cheap" has an important contribution to the positive pole of the first dimension, while the category "super cheap" has a major contribution to the negative pole of the first dimension.

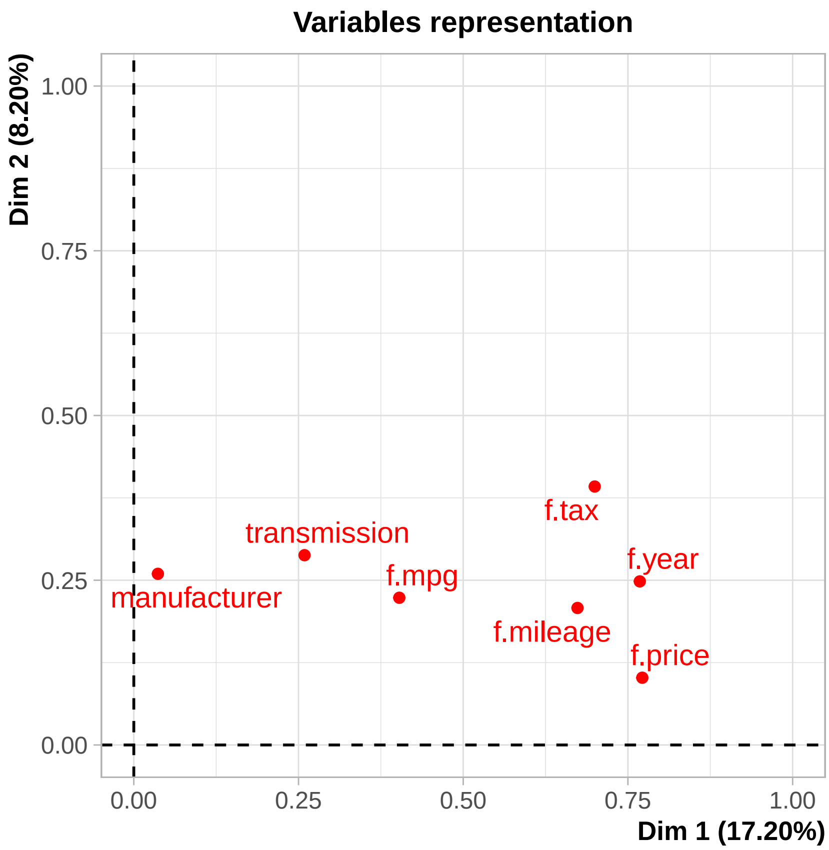
In other words, dimension 1 is mainly defined by the opposition of cheap cars (positive pole), and super cheap cars (negative pole).

6 MCA Analysis

```
par(mfrow = c(1, 1))
vars_dis <- c("model", "transmission", "fuelType", "engineSize", "manufacturer",
    "f.price", "f.tax", "f.mileage", "f.mpg", "f.year")
res.mca <- MCA(df[, c(vars_dis[c(2, 5:10)])])
```

```
## Warning: ggrepel: 1 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```






```r
# summary(res.mca,nbelements=50, nbind=0)
```

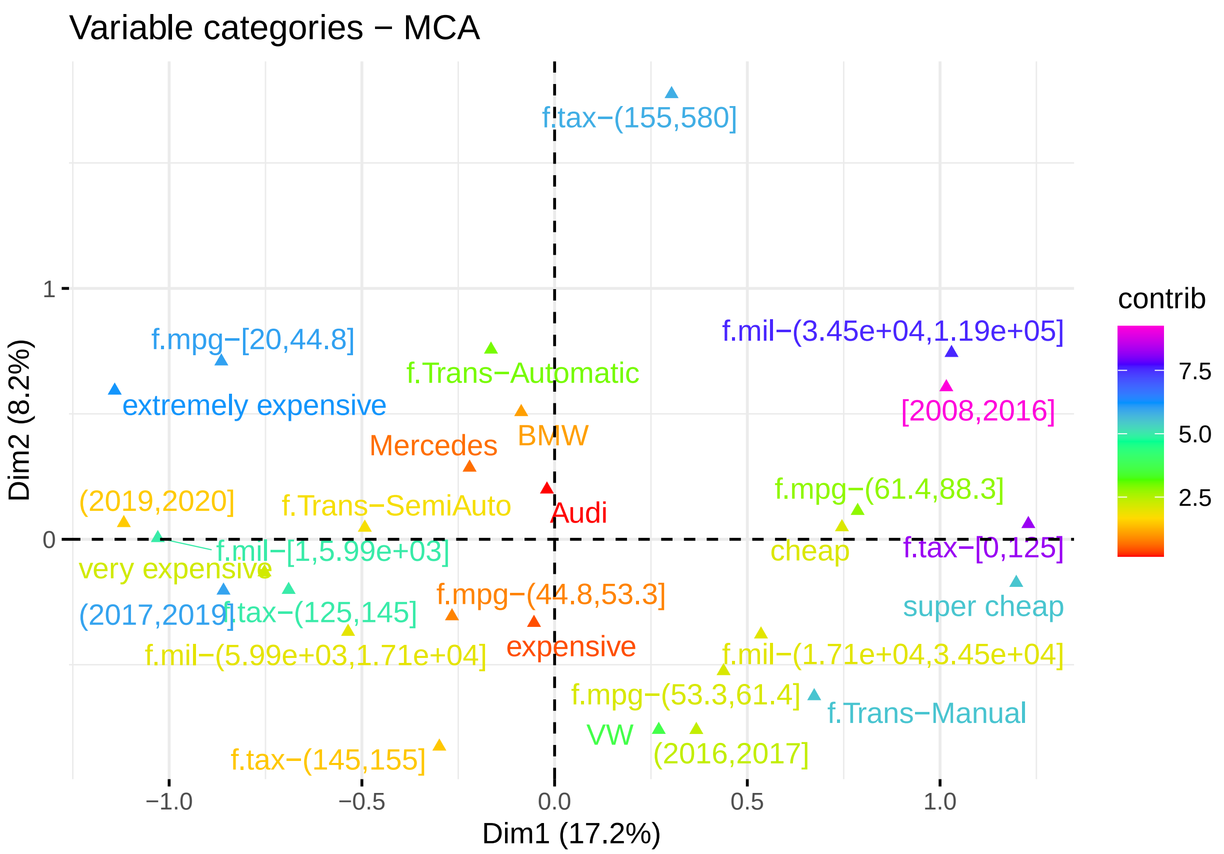
6.1 Interpreting the axes association to factor map

6.1.1 Variables representation graph (3rd graph from MCA)

Correlation between variables and principal dimensions: From the Variables representation map, we can see that all variables are placed on the first quadrant of the first factorial plane. From a more detailed look, the manufacturer variable has no importance on the 1st dimension but explains the 2nd dimension a bit, so cars will be scattered according to its manufacturer alongside the 2nd dimension. Transmission explains the 1st and 2nd dimension in a equal fashion, so the cars will be scattered alongside the 2 dimensions. Variables f.mpg, f.mileage, f.price, f.year and f.tax have a high representation on the positive side of the 1st dimension, so cars will be scattered alongside the 1st dimension according to these variables.

6.1.2 MCA factor map

```r
fviz_mca_var(res.mca, col.var="contrib",
             gradient.cols =rainbow(7) ,
             repel = TRUE, # avoid text overlapping (slow)
             ggtheme = theme_minimal()
             )
```



From this graph we can conclude that, Audi, Mercedes and BMW cars do not have a strong correlation with any factor variable because they are more or less centered and far away from the center of any other variable. As BMW and Mercedes are close together we could say that they have similar qualities. The most relevant groups are: * Very expensive cars with cars sold between 2017-2019 and taxes between 125-145. * VW and cars sold between 2016-2017 * Cars sold in 2008-2016 and very high mileage.

6.2 Eigenvalues and dominant axes analysis

How many axes we have to consider for next Hierarchical Classification stage?

```r
mean(res.mca$eig[, 1])
```

```
## [1] 0.1428571
```

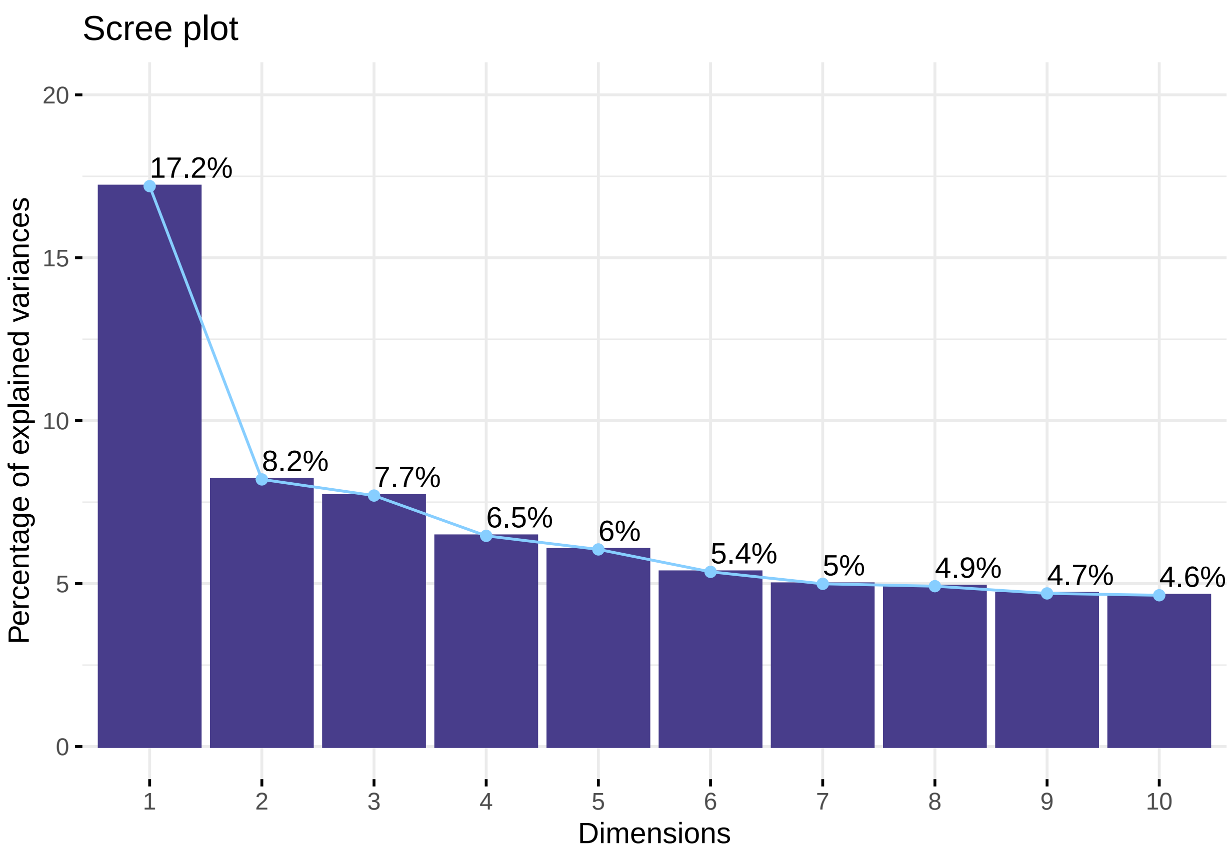
```r
head(get_eigenvalue(res.mca), 10)
```

```
##        eigenvalue variance.percent cumulative.variance.percent
## Dim.1   0.5159460        17.198199                    17.19820
## Dim.2   0.2459281         8.197604                    25.39580
## Dim.3   0.2310719         7.702395                    33.09820
## Dim.4   0.1939765         6.465885                    39.56408
## Dim.5   0.1814838         6.049460                    45.61354
## Dim.6   0.1608479         5.361598                    50.97514
## Dim.7   0.1498000         4.993333                    55.96847
## Dim.8   0.1476286         4.920954                    60.88943
## Dim.9   0.1409795         4.699318                    65.58875
## Dim.10  0.1392720         4.642401                    70.23115
```

We consider, according to the generalized Kaiser theorem, all those dimensions such that their eigenvalue is greater than the mean. We see that the average gives us 0.1428571. Therefore, we will take up to dimension 8, which represents the 60.88943% of the sample.

We can also visualize the percentages of inertia explained by each MCA dimensions:

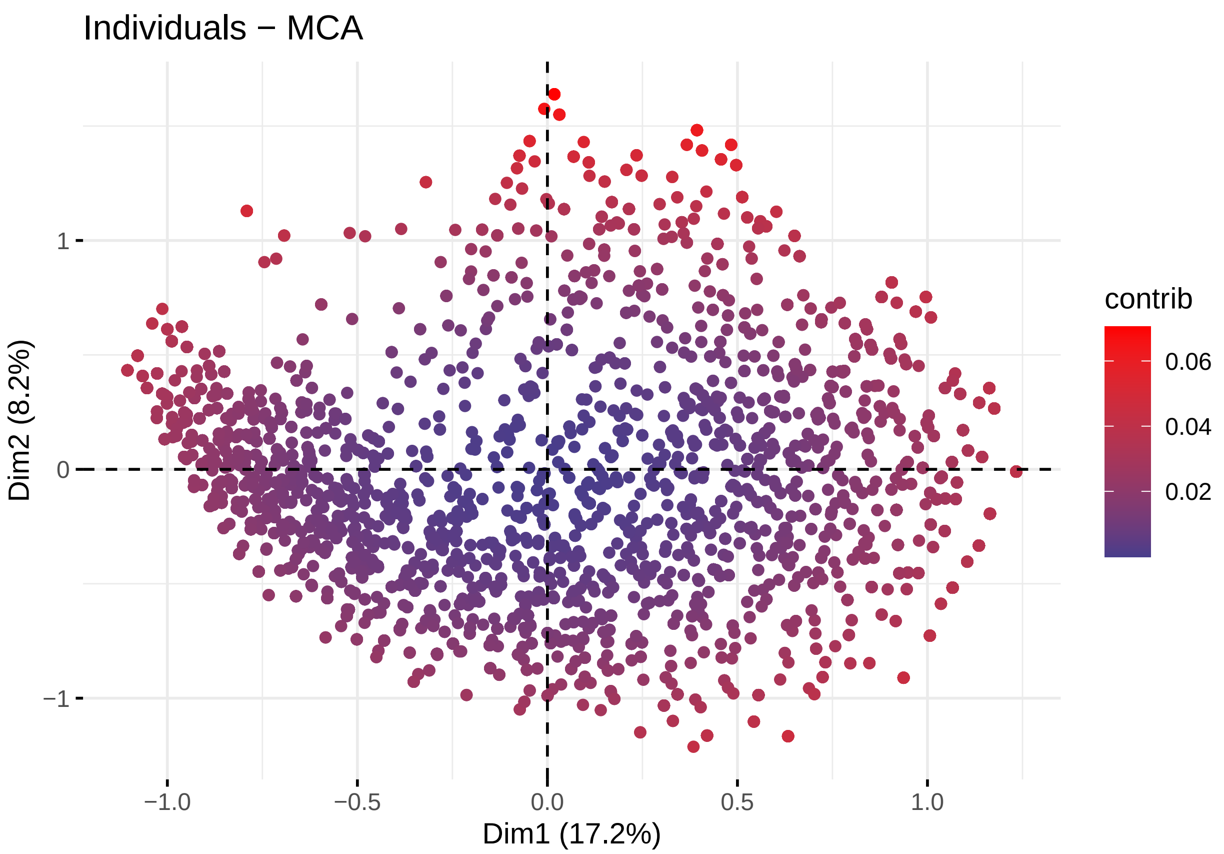
```r
fviz_screeplot(res.mca, addlabels = TRUE, ylim = c(0, 20), barfill = "darkslateblue",
    barcolor = "darkslateblue", linecolor = "skyblue1")
```



6.3 Individuals point of view

Are they any individuals "too contributive"?

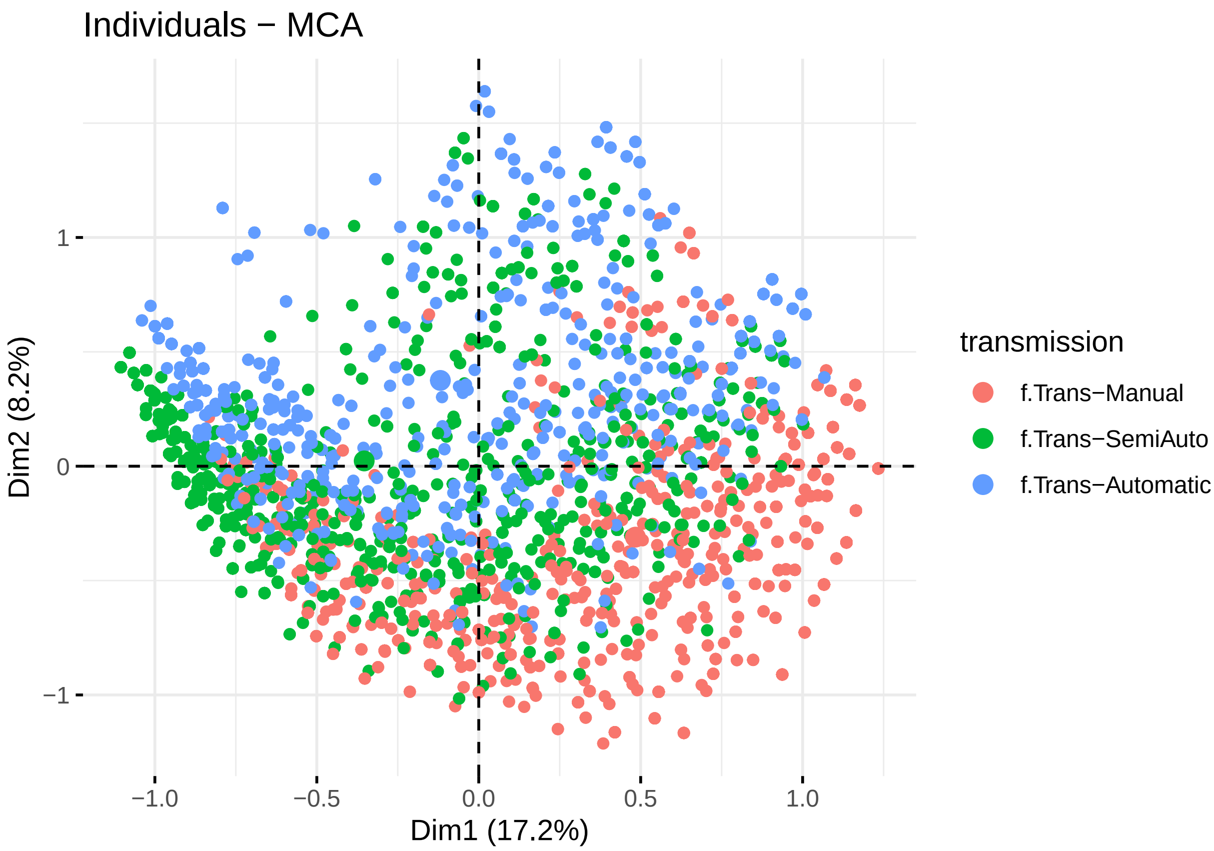
```
fviz_mca_ind(res.mca, geom = c("point"), col.ind = "contrib", gradient.cols = c("darkslatebl
    "red"))
```



We can see that individuals located in the periphery are the most contributive.

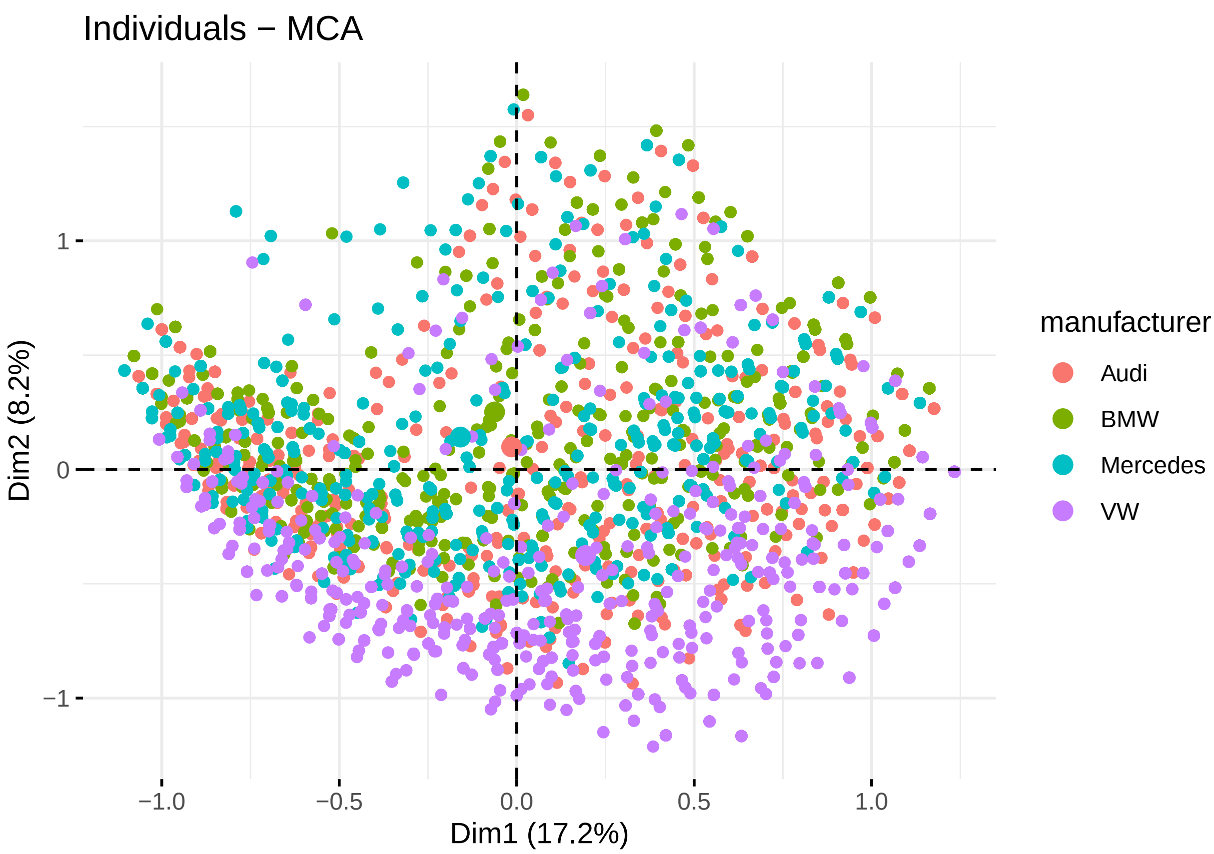
6.3.1 Groups of individuals

```
fviz_mca_ind(res.mca, label = "none", habillage = "transmission")
```



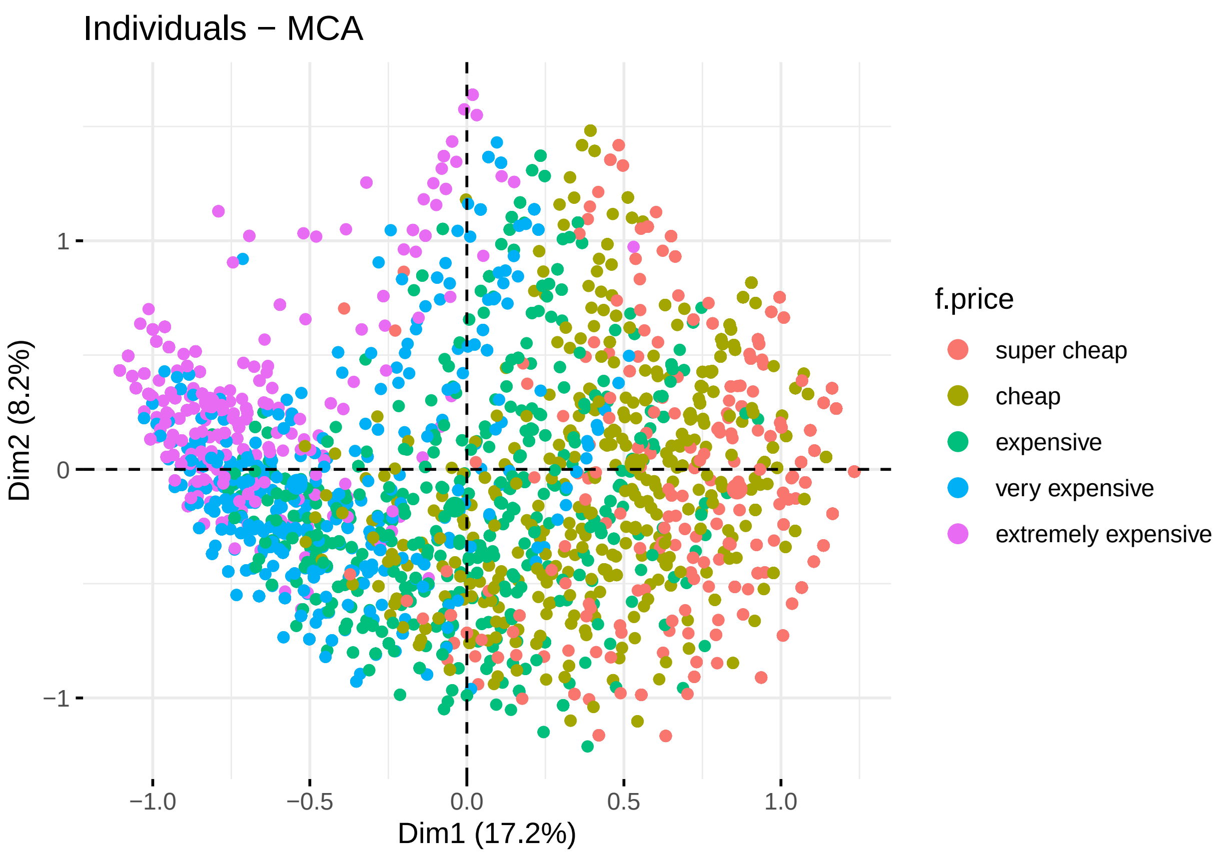
Automatic cars are located on the positive side of the 2nd dimension where manual cars are on the nagative side of the 2nd dimension.

```
fviz_mca_ind(res.mca, label = "none", habillage = "manufacturer")
```



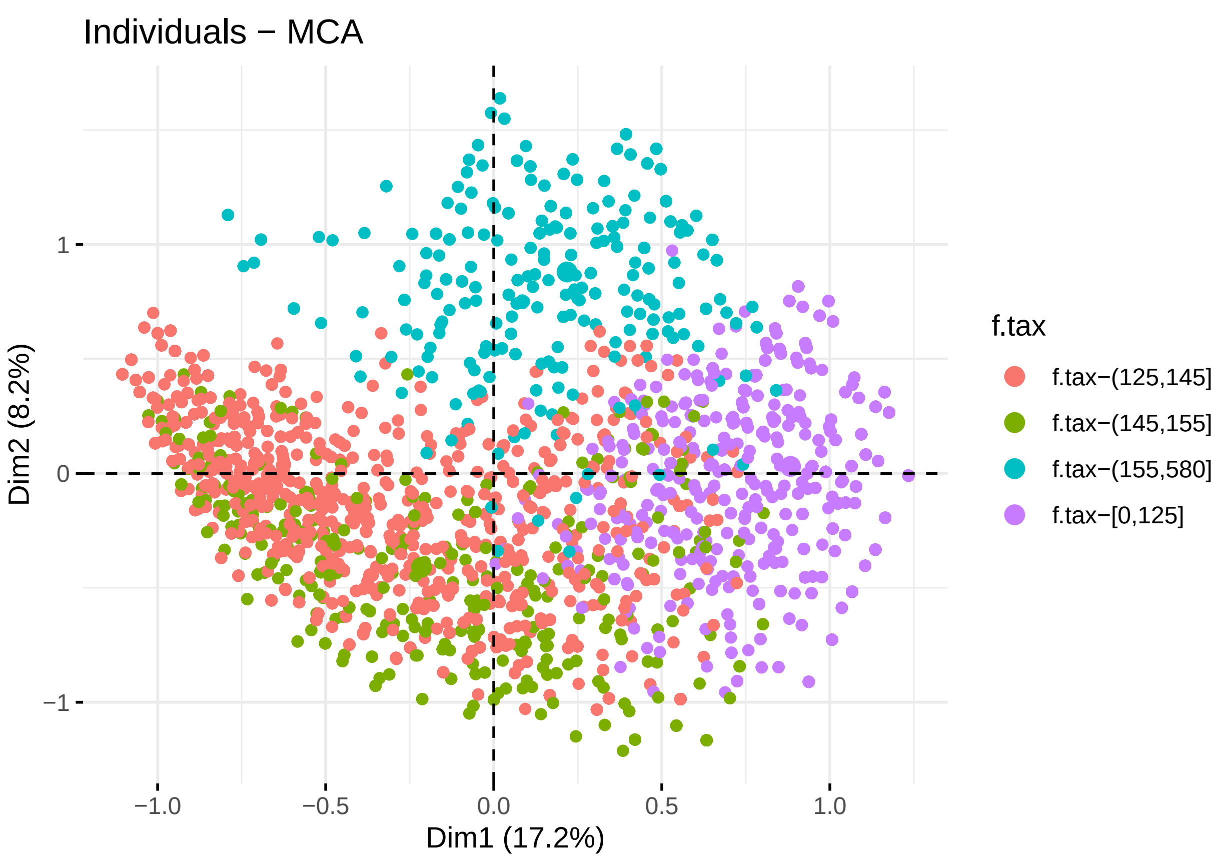
We cannot see any significant difference in the position of observations.

```r
fviz_mca_ind(res.mca, label = "none", habillage = "f.price")
```



Individuals are distributed across the 1st dimension: from extremely expensive cars on the negative side to super cheap on the positive side.

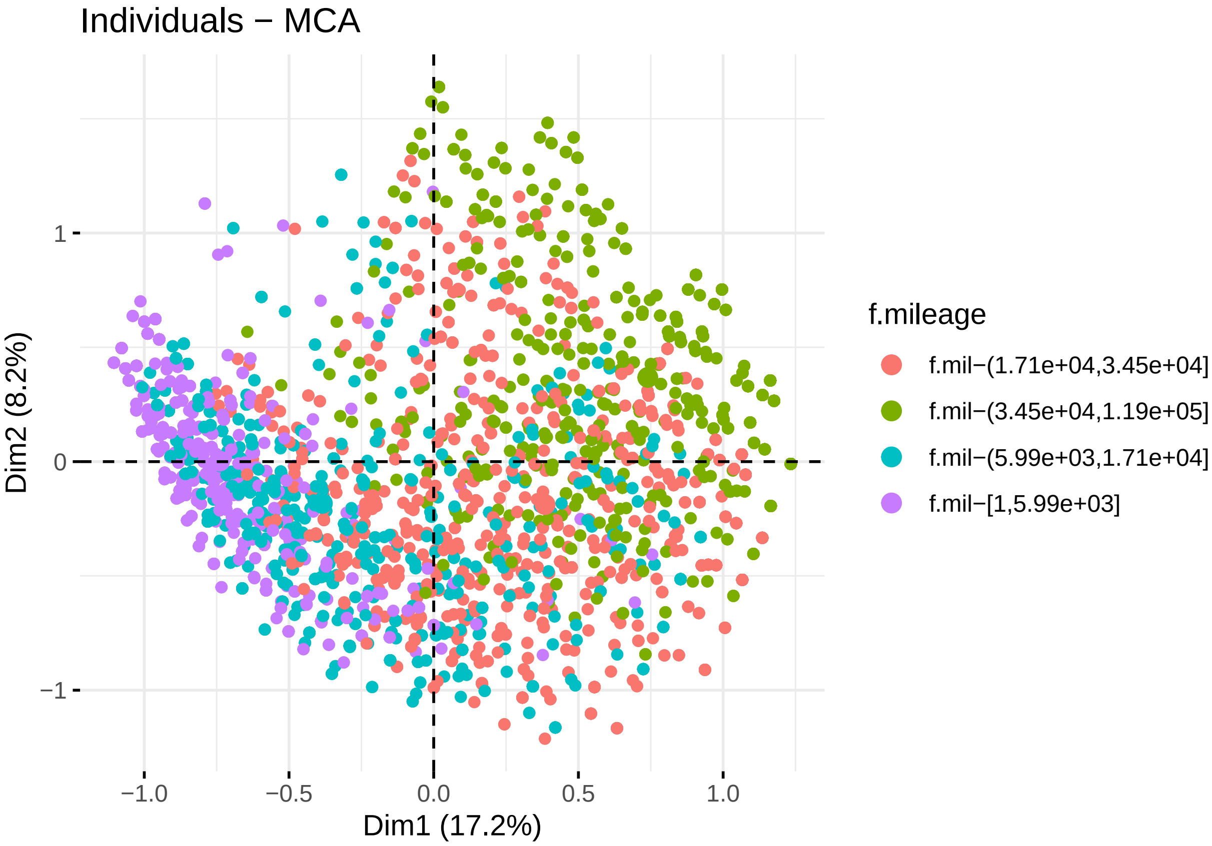
```r
fviz_mca_ind(res.mca, label = "none", habillage = "f.tax")
```



We can only

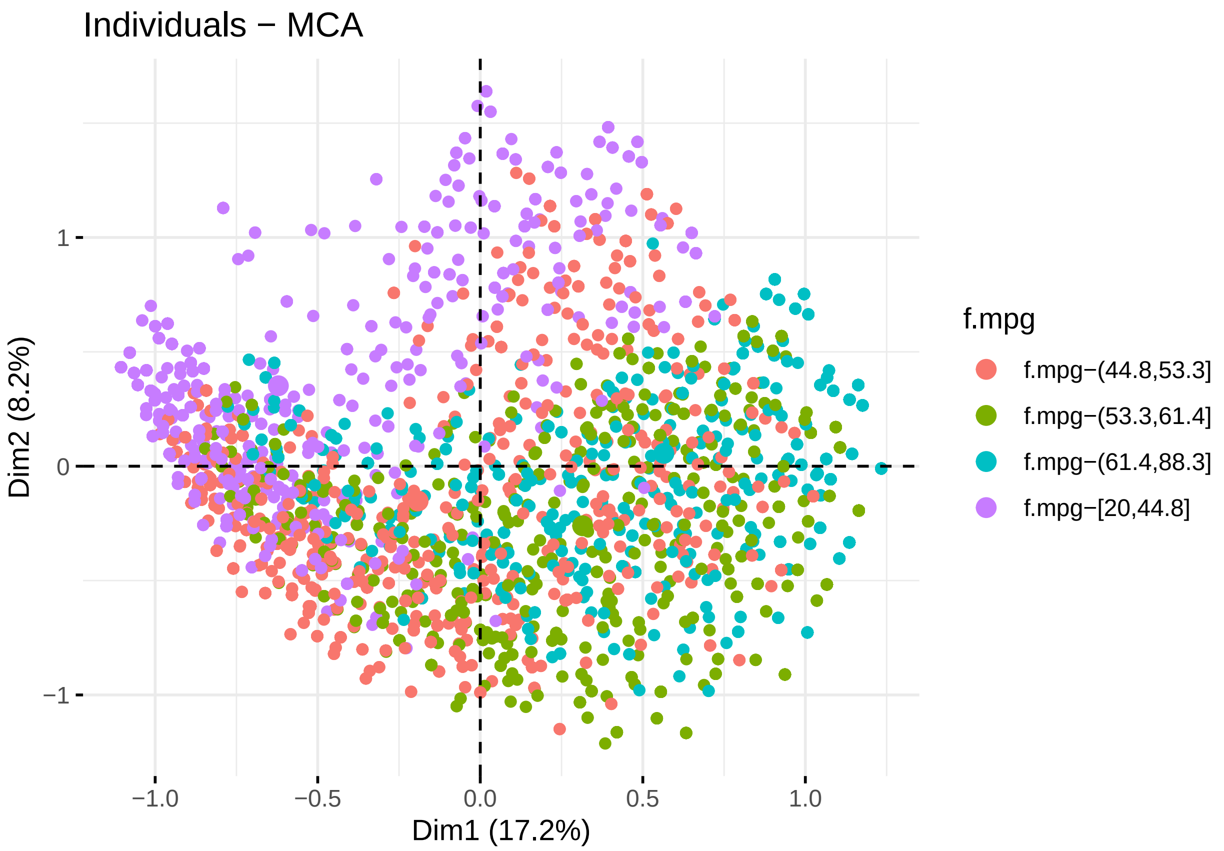
appreciate that observations with very high taxes are located on the positive side of the 2nd dimension.

```
fviz_mca_ind(res.mca, label = "none", habillage = "f.mileage")
```



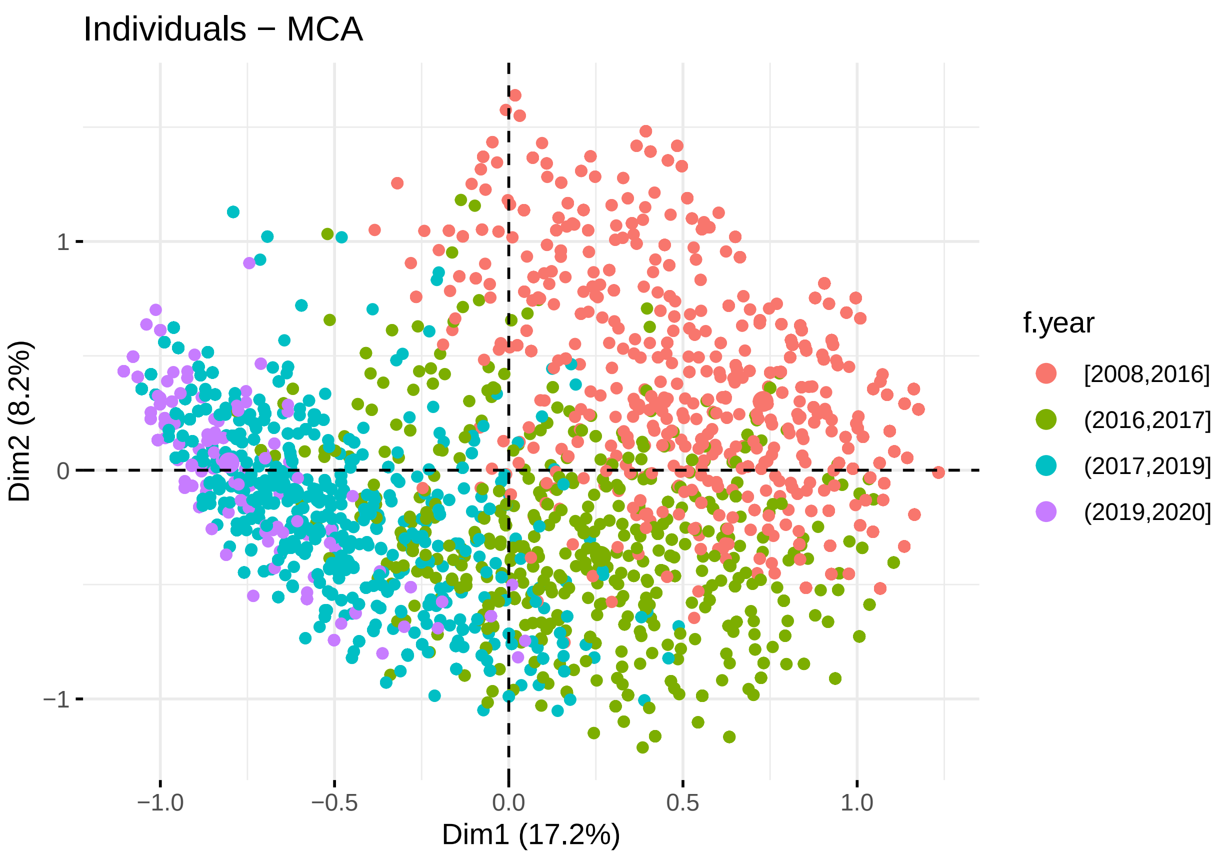
Individuals are distributed across the 1st dimension: from very low mileage on the negative side to very high mileage on the positive side.

```
fviz_mca_ind(res.mca, label = "none", habillage = "f.mpg")
```



We cannot see any significant difference in the position of observations.

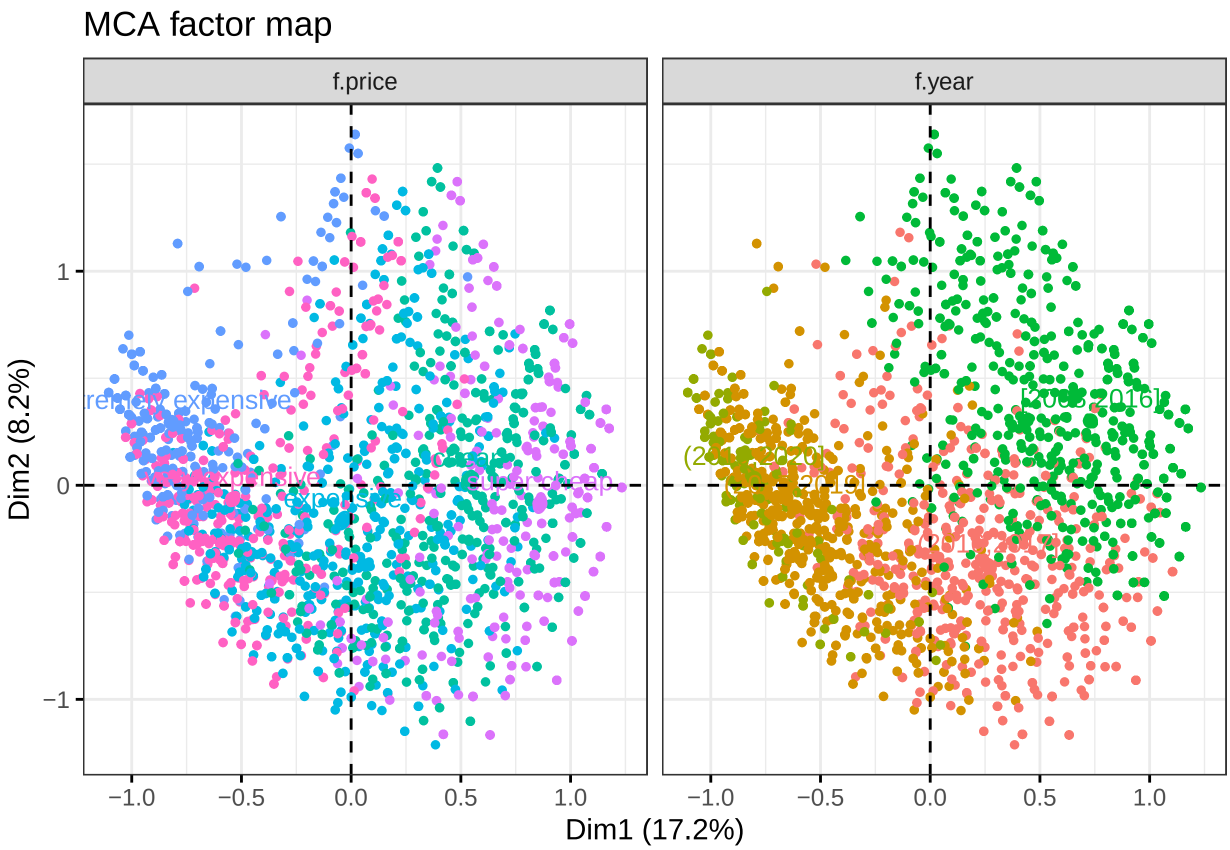
```
fviz_mca_ind(res.mca, label = "none", habillage = "f.year")
```



Individuals are distributed across the 1st dimension: from an early time of selling (high year) on the negative side to very old year of selling (low year) on the positive side.

```r
fviz_ellipses(res.mca, c("f.price", "f.year"), geom = "point")
```

```
## Warning: `gather_()` was deprecated in tidyr 1.2.0.
## Please use `gather()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```



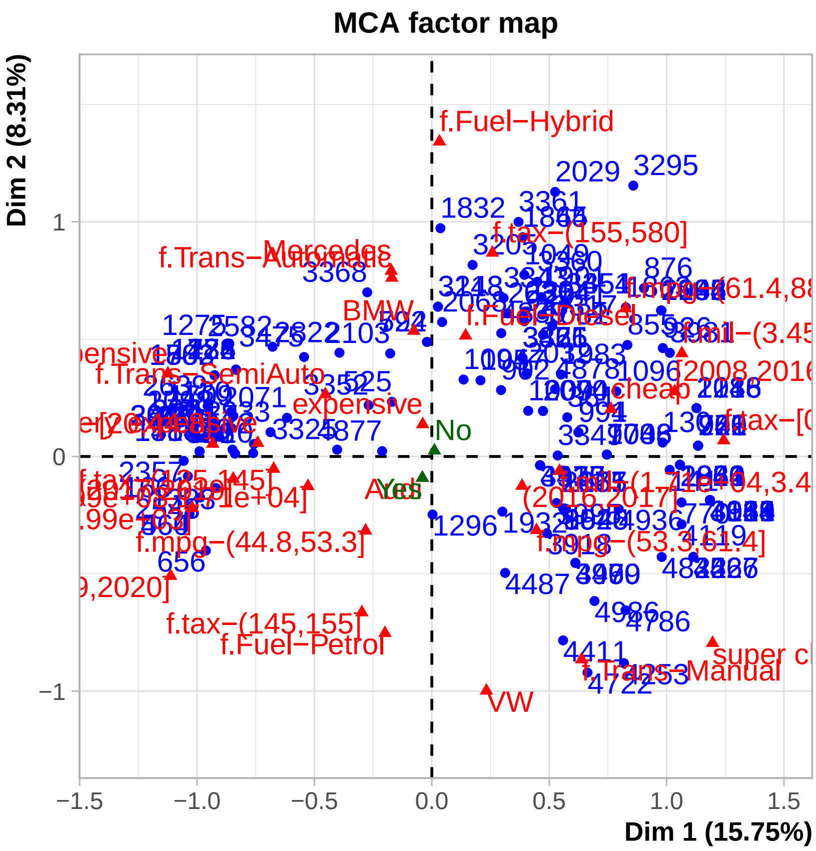
6.4 MCA using multivariant

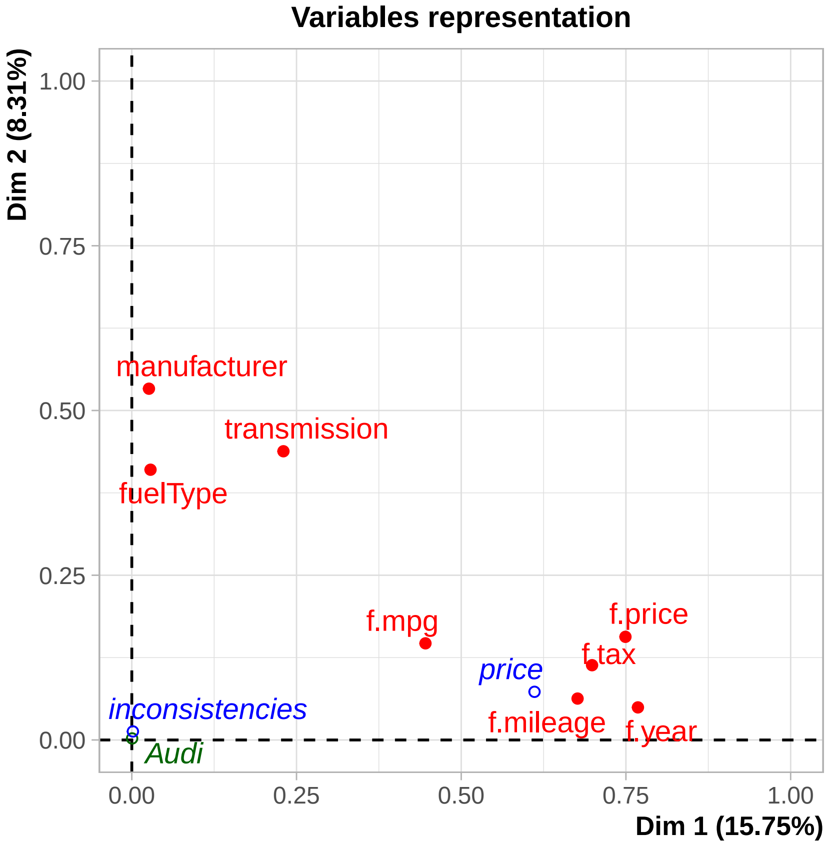
How do supplementary variables enhance the axis interpretation?

```r
llvout<-which(df$mout=="MvOut.Yes");length(llvout) #Multivariate outliers
```

```
## [1] 164
```

```r
res.mca<-MCA(df[,c(vars_dis[c(2:3,5:10)],"price", "inconsistencies", "Audi")], quali.sup=c(1
```





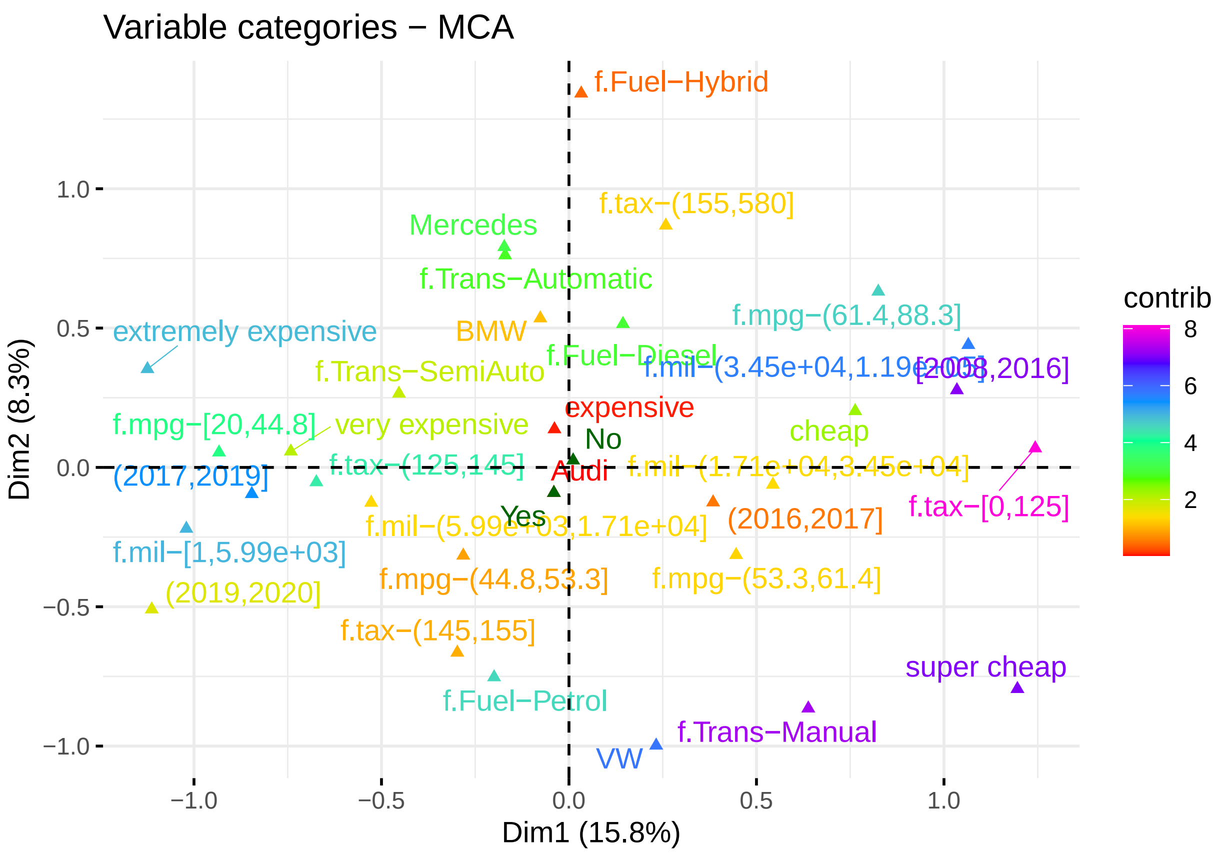

```r
mean(res.mca$eig[,1])
```

```
## [1] 0.125
```

```r
head(get_eigenvalue(res.mca), 10) #keep 9 dimensions
```

```
##         eigenvalue variance.percent cumulative.variance.percent
## Dim.1    0.4528704        15.752014                    15.75201
## Dim.2    0.2387697         8.305035                    24.05705
## Dim.3    0.2019698         7.025038                    31.08209
## Dim.4    0.1820983         6.333854                    37.41594
## Dim.5    0.1576542         5.483626                    42.89957
## Dim.6    0.1474280         5.127930                    48.02750
## Dim.7    0.1338591         4.655968                    52.68347
## Dim.8    0.1305733         4.541681                    57.22515
## Dim.9    0.1282831         4.462020                    61.68717
## Dim.10   0.1227017         4.267886                    65.95505
```

```r
fviz_mca_var(res.mca, col.var="contrib",
             gradient.cols =rainbow(7) ,
             repel = TRUE, # avoid text overlapping (slow)
             ggtheme = theme_minimal()
             )
```



From the supplementary quantitative variables graph we can see that variable price will explain a lot of the negative 1st dimension, so cars with a low price will be located on the positive part of the 1st dimension and expensive cars on the negative side of the dimension. In addition, when we add the multivariate outliers as supplementary individuals, they get taken out of the mca computation so they do not disrupt the calculus with their extreme values. Now, when we draw a variable factor map, we can also see the position of our supplementary variables relative to our original categorical variables and see how they correlate to each other. For example, from the previous graph we can see that our binary target variable "Audi", does not correlate with any variable whatsoever, as its 2 main values "Yes" and "No" are located on the center of the graph. We also saw this in the previous MCA analysis, when the Audi point of variable manufacturer were located on the center of the graph.

7 Hierarchical Clustering from MCA

```
res.hcmc <- HCPC(res.mca, nb.clust = -1, order = TRUE)
```

```
res.hcmc$call$t$within[1:15]
```

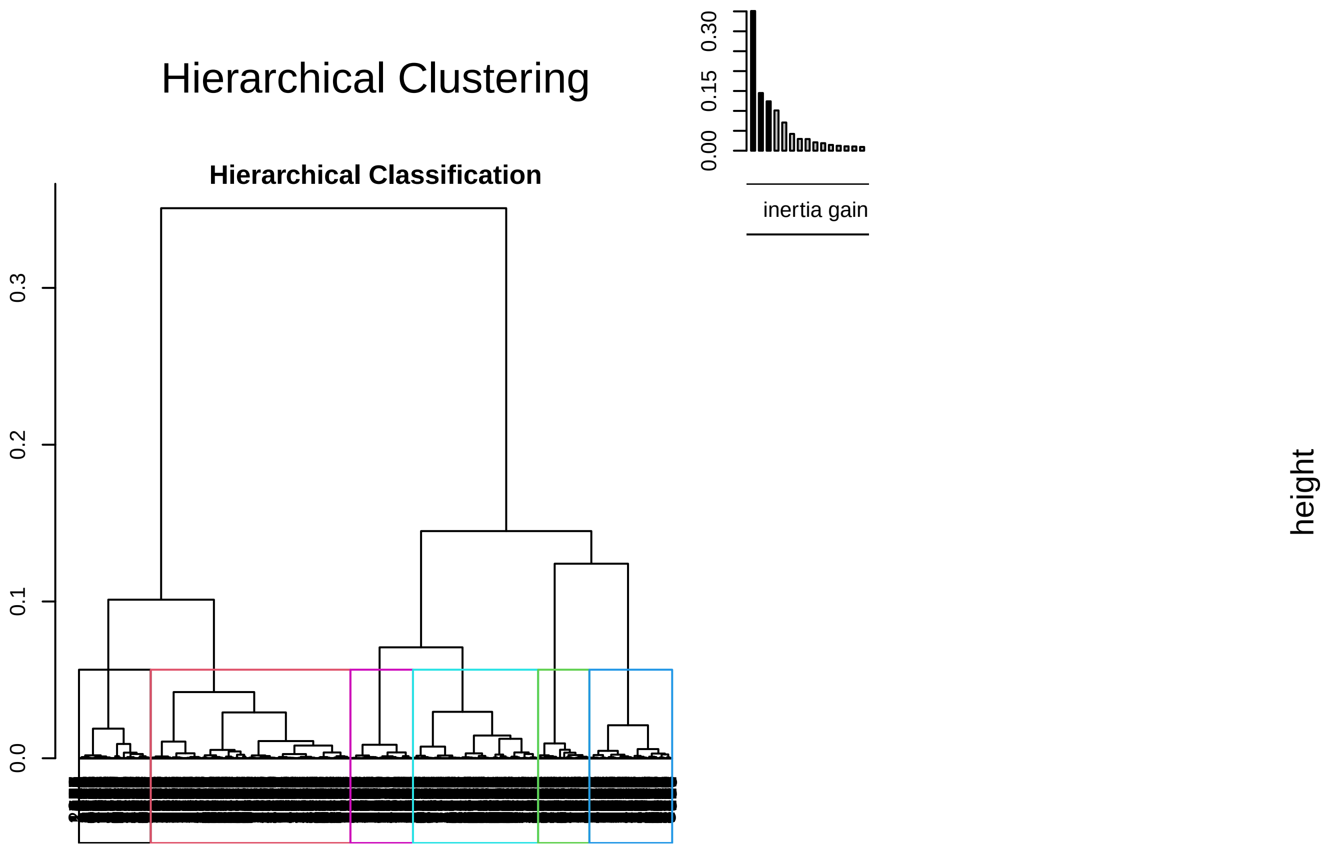
```
##  [1] 1.2333626 0.8825231 0.7376257 0.6135225 0.5123818 0.4416251 0.3993994
##  [8] 0.3697817 0.3405457 0.3194989 0.3006113 0.2861040 0.2736856 0.2627169
## [15] 0.2520782
```

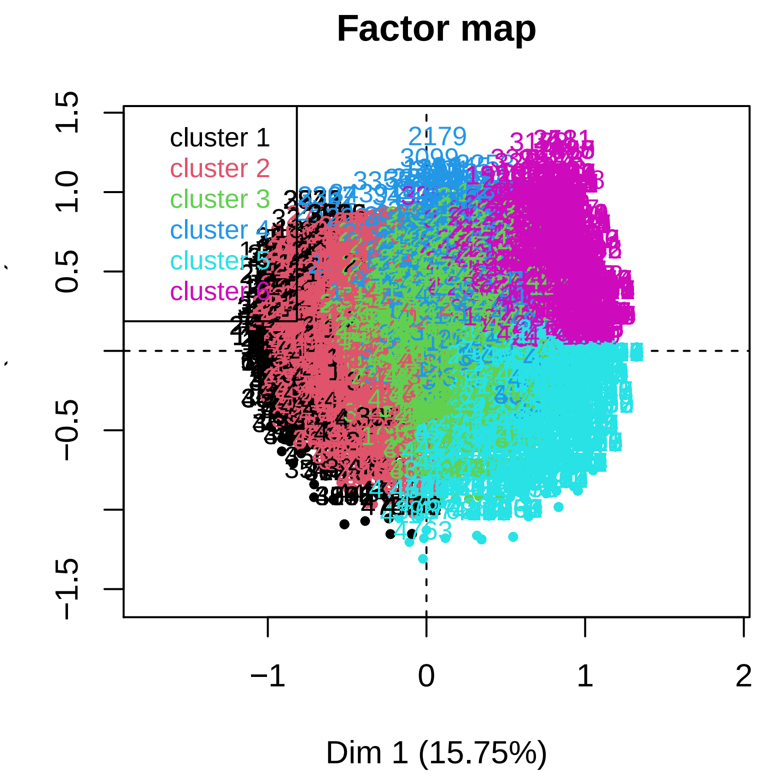
```
(res.hcmc$call$t$within[1] - res.hcmc$call$t$within[1:10])/res.hcmc$call$t$within[1]
```

```
##  [1] 0.0000000 0.2844577 0.4019393 0.5025611 0.5845651 0.6419341 0.6761703
##  [8] 0.7001841 0.7238884 0.7409529
```

We consider that 6 is the better number of cluster to have because it contains the 64.2% of the variability and having more clusters doesn't increases significantly the amount of variability.

```
res.hcmc <- HCPC(res.mca, nb.clust = 6, order = TRUE)
```

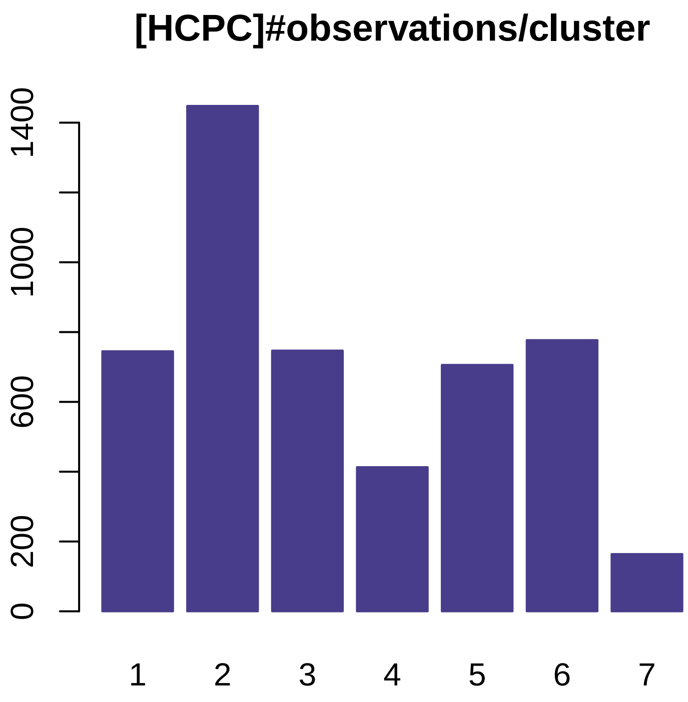




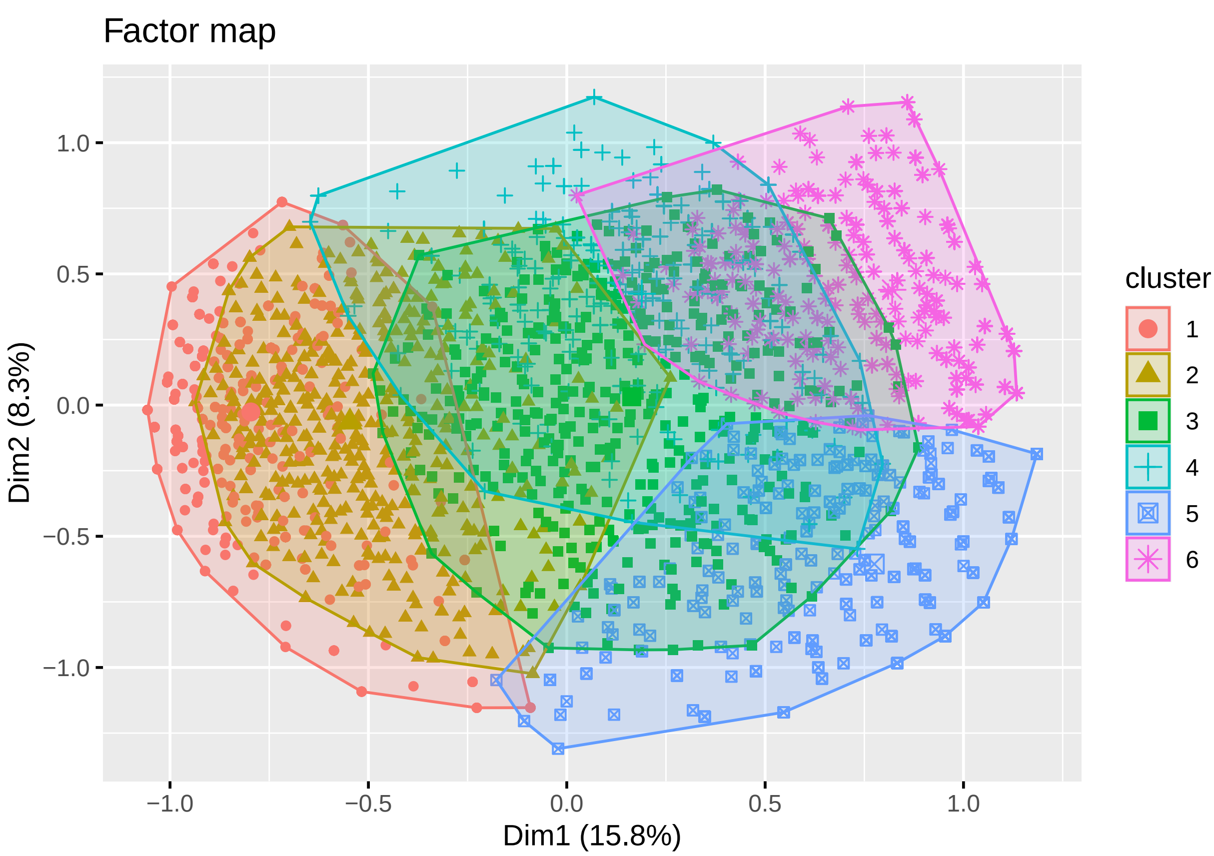

```
df$claHCMC <- 7
df[row.names(res.hcmc$data.clust), "claHCMC"] <- res.hcmc$data.clust$clust
df$claHCMC <- factor(df$claHCMC)
table(df$claHCMC)
```

```
##
##    1    2    3    4    5    6    7
##  745 1448  747  413  706  777  164
```

```
# Multivariate outliers will be in cluster 7

# Interpret clustering results
barplot(table(df$claHCMC), col = "darkslateblue", border = "darkslateblue",
    main = "[HCPC]#observations/cluster")
```




```r
# Individuals factor map
fviz_cluster(res.hcmc, geom = "point", main = "Factor map")
```



7.1 Description of clusters by categorical variables

We proceed to explain the data obtained.

```
res.hcmc$desc.var$test.chi2
```

```
##                   p.value df
## f.price       0.000000e+00 20
## f.tax         0.000000e+00 15
## f.mileage     0.000000e+00 15
## f.mpg         0.000000e+00 15
## f.year        0.000000e+00 15
## transmission 1.536482e-282 10
## manufacturer 1.367309e-249 15
## fuelType     1.437572e-177 10
## Audi          2.679468e-15  5
```

We can see the intensity of the variables, in our case the categorical variables that affect more to the clustering are **f.price**, **f.tax**, **f.mileage**, **f.mpg** and **f.year** because are those ones with the smallest p.value. Next, we want to see for each cluster which are the categories that characterize them.

```
res.hcmc$desc.var$category
```

```
## $`1`
##                                         Cla/Mod    Mod/Cla    Global
## f.mileage=f.mil-[1,5.99e+03]         58.5385878 95.7046980 25.186104
## f.price=extremely expensive          58.5805085 74.2281879 19.520265
## f.year=(2019,2020]                   98.0582524 40.6711409  6.389578
## f.mpg=f.mpg-[20,44.8]                37.9949452 60.5369128 24.545079
## f.tax=f.tax-(125,145]                26.0301109 88.1879195 52.191894
## transmission=f.Trans-SemiAuto        22.4210526 57.1812081 39.288668
## f.year=(2017,2019]                   21.8625498 58.9261745 41.521919
## Audi=Yes                             20.5348615 28.8590604 21.650124
## manufacturer=Audi                    20.5348615 28.8590604 21.650124
## fuelType=f.Fuel-Petrol               17.2069825 46.3087248 41.459884
## manufacturer=BMW                     17.6413255 24.2953020 21.215881
## f.mpg=f.mpg-(44.8,53.3]              12.9672006 22.8187919 27.109181
## fuelType=f.Fuel-Diesel               13.9089595 51.6778523 57.237386
## Audi=No                              13.9878596 71.1409396 78.349876
## manufacturer=VW                      10.9152542 21.6107383 30.500414
## f.mpg=f.mpg-(53.3,61.4]               7.4789916 11.9463087 24.607113
## f.tax=f.tax-(155,580]                 1.4251781  0.8053691  8.705542
## transmission=f.Trans-Manual           6.5320665 14.7651007 34.822167
## f.mpg=f.mpg-(61.4,88.3]               3.0487805  4.6979866 23.738627
## f.price=expensive                     2.1105528  2.8187919 20.574855
## f.price=super cheap                   1.5334064  1.8791946 18.879239
## f.price=cheap                         1.1190234  1.4765101 20.326716
## f.year=(2016,2017]                    0.3460208  0.4026846 17.928040
## f.mileage=f.mil-(5.99e+03,1.71e+04]   1.2997563  2.1476510 25.454921
## f.mileage=f.mil-(1.71e+04,3.45e+04]   1.0425020  1.7449664 25.785773
## f.mileage=f.mil-(3.45e+04,1.19e+05]   0.2631579  0.4026846 23.573201
```

```
## f.tax=f.tax-[0,125]                  0.0000000  0.0000000 28.908189
## f.year=[2008,2016]                   0.0000000  0.0000000 34.160463
##                                          p.value     v.test
## f.mileage=f.mil-[1,5.99e+03]         0.000000e+00        Inf
## f.price=extremely expensive         4.117081e-294  36.652881
## f.year=(2019,2020]                  3.939040e-262  34.587172
## f.mpg=f.mpg-[20,44.8]               3.615531e-118  23.110684
## f.tax=f.tax-(125,145]               1.862956e-113  22.637253
## transmission=f.Trans-SemiAuto        6.995837e-27  10.734683
## f.year=(2017,2019]                   2.611729e-25  10.395042
## Audi=Yes                             4.429160e-07   5.049522
## manufacturer=Audi                    4.429160e-07   5.049522
## fuelType=f.Fuel-Petrol               3.633016e-03   2.908384
## manufacturer=BMW                     2.718493e-02   2.208852
## f.mpg=f.mpg-(44.8,53.3]              3.745563e-03  -2.898831
## fuelType=f.Fuel-Diesel               8.956296e-04  -3.321413
## Audi=No                              4.429160e-07  -5.049522
## manufacturer=VW                      4.205414e-09  -5.875904
## f.mpg=f.mpg-(53.3,61.4]              1.656877e-20  -9.282410
## f.tax=f.tax-(155,580]                4.316844e-24 -10.124157
## transmission=f.Trans-Manual          5.533705e-40 -13.234673
## f.mpg=f.mpg-(61.4,88.3]              1.413070e-51 -15.108981
## f.price=expensive                    1.763976e-52 -15.245493
## f.price=super cheap                  2.406332e-53 -15.375057
## f.price=cheap                        5.632234e-63 -16.750320
## f.year=(2016,2017]                   1.157032e-64 -16.979890
## f.mileage=f.mil-(5.99e+03,1.71e+04]  9.052368e-79 -18.790383
## f.mileage=f.mil-(1.71e+04,3.45e+04]  3.122310e-84 -19.446465
## f.mileage=f.mil-(3.45e+04,1.19e+05]  7.831754e-90 -20.097047
## f.tax=f.tax-[0,125]                 1.067790e-122 -23.556858
## f.year=[2008,2016]                  7.886437e-151 -26.158508
##
## $`2`
##                                        Cla/Mod     Mod/Cla    Global
## f.year=(2017,2019]                  69.72111554 96.68508287 41.521919
## f.mileage=f.mil-(5.99e+03,1.71e+04] 73.43623071 62.43093923 25.454921
## f.tax=f.tax-(125,145]               50.67353407 88.32872928 52.191894
## f.price=very expensive              64.43556444 44.54419890 20.698925
## f.mpg=f.mpg-(44.8,53.3]             44.46987033 40.26243094 27.109181
## f.mpg=f.mpg-[20,44.8]               39.67986521 32.52762431 24.545079
## transmission=f.Trans-SemiAuto       36.42105263 47.79005525 39.288668
## f.mileage=f.mil-[1,5.99e+03]        38.58784893 32.45856354 25.186104
## fuelType=f.Fuel-Petrol              34.61346633 47.92817680 41.459884
## Audi=No                             31.82897862 83.28729282 78.349876
## f.price=expensive                   36.48241206 25.06906077 20.574855
## manufacturer=Mercedes               35.24844720 31.35359116 26.633581
## f.tax=f.tax-(145,155]               33.87423935 11.53314917 10.194376
## fuelType=f.Fuel-Hybrid              15.87301587  0.69060773  1.302730
## fuelType=f.Fuel-Diesel              26.87861272 51.38121547 57.237386
## Audi=Yes                            23.11365807 16.71270718 21.650124
## manufacturer=Audi                   23.11365807 16.71270718 21.650124
## f.mpg=f.mpg-(53.3,61.4]             18.99159664 15.60773481 24.607113
## transmission=f.Trans-Manual         21.14014252 24.58563536 34.822167
```

```
## f.mpg=f.mpg-(61.4,88.3]            14.63414634 11.60220994 23.738627
## f.year=(2019,2020]                  0.97087379  0.20718232  6.389578
## f.price=cheap                      12.10579858  8.21823204 20.326716
## f.tax=f.tax-(155,580]               0.23752969  0.06906077  8.705542
## f.year=(2016,2017]                  4.15224913  2.48618785 17.928040
## f.price=super cheap                 1.64293538  1.03591160 18.879239
## f.mileage=f.mil-(1.71e+04,3.45e+04]  4.49077787  3.86740331 25.785773
## f.mileage=f.mil-(3.45e+04,1.19e+05]  1.57894737  1.24309392 23.573201
## f.tax=f.tax-[0,125]                 0.07153076  0.06906077 28.908189
## f.year=[2008,2016]                  0.54479419  0.62154696 34.160463
##                                          p.value     v.test
## f.year=(2017,2019]                  0.000000e+00        Inf
## f.mileage=f.mil-(5.99e+03,1.71e+04] 2.672430e-310  37.655369
## f.tax=f.tax-(125,145]              1.216746e-261  34.554576
## f.price=very expensive             2.080427e-146  25.766974
## f.mpg=f.mpg-(44.8,53.3]             1.178919e-39  13.177725
## f.mpg=f.mpg-[20,44.8]               1.134397e-16   8.289799
## transmission=f.Trans-SemiAuto       3.561546e-15   7.869467
## f.mileage=f.mil-[1,5.99e+03]        6.321487e-14   7.501244
## fuelType=f.Fuel-Petrol              2.720752e-09   5.947616
## Audi=No                             2.961595e-08   5.543692
## f.price=expensive                   6.165333e-07   4.985965
## manufacturer=Mercedes               1.526548e-06   4.807729
## f.tax=f.tax-(145,155]               4.620395e-02   1.993525
## fuelType=f.Fuel-Hybrid              1.074246e-02  -2.550967
## fuelType=f.Fuel-Diesel              8.032469e-08  -5.366398
## Audi=Yes                            2.961595e-08  -5.543692
## manufacturer=Audi                   2.961595e-08  -5.543692
## f.mpg=f.mpg-(53.3,61.4]             1.301156e-22  -9.785372
## transmission=f.Trans-Manual         2.973381e-23  -9.933595
## f.mpg=f.mpg-(61.4,88.3]             3.513147e-42 -13.609568
## f.year=(2019,2020]                  1.097573e-44 -14.024906
## f.price=cheap                       3.222580e-48 -14.590620
## f.tax=f.tax-(155,580]               3.992830e-67 -17.309439
## f.year=(2016,2017]                  1.165346e-96 -20.862843
## f.price=super cheap                4.808812e-132 -24.451856
## f.mileage=f.mil-(1.71e+04,3.45e+04] 9.842729e-144 -25.527180
## f.mileage=f.mil-(3.45e+04,1.19e+05] 2.926318e-172 -27.979005
## f.tax=f.tax-[0,125]                2.654675e-263 -34.665003
## f.year=[2008,2016]                 4.003071e-307 -37.460827
##
## $`3`
##                                        Cla/Mod    Mod/Cla    Global
## f.year=(2016,2017]                 66.0899654 76.7068273 17.928040
## f.mileage=f.mil-(1.71e+04,3.45e+04] 44.9077787 74.9665328 25.785773
## f.price=expensive                  36.8844221 49.1298527 20.574855
## f.tax=f.tax-(145,155]              33.8742394 22.3560910 10.194376
## f.price=cheap                      25.4323499 33.4672021 20.326716
## f.mpg=f.mpg-(53.3,61.4]            23.6134454 37.6171352 24.607113
## transmission=f.Trans-SemiAuto      18.7368421 47.6572959 39.288668
## fuelType=f.Fuel-Diesel             17.5216763 64.9263722 57.237386
## Audi=Yes                           19.8662846 27.8447122 21.650124
## manufacturer=Audi                  19.8662846 27.8447122 21.650124
```

```
## f.mpg=f.mpg-(61.4,88.3]            18.9024390 29.0495315 23.738627
## manufacturer=Mercedes              17.3136646 29.8527443 26.633581
## transmission=f.Trans-Manual        13.5391924 30.5220884 34.822167
## transmission=f.Trans-Automatic     13.0191693 21.8206158 25.889165
## Audi=No                            14.2253893 72.1552878 78.349876
## fuelType=f.Fuel-Petrol             12.6184539 33.8688086 41.459884
## f.tax=f.tax-[0,125]                11.4449213 21.4190094 28.908189
## f.price=very expensive              9.2907093 12.4497992 20.698925
## manufacturer=VW                     9.9661017 19.6787149 30.500414
## f.mileage=f.mil-(5.99e+03,1.71e+04]  8.5296507 14.0562249 25.454921
## f.year=(2019,2020]                  0.3236246  0.1338688  6.389578
## f.mileage=f.mil-(3.45e+04,1.19e+05]  6.8421053 10.4417671 23.573201
## f.tax=f.tax-(155,580]               1.4251781  0.8032129  8.705542
## f.price=extremely expensive         3.1779661  4.0160643 19.520265
## f.mpg=f.mpg-[20,44.8]               2.8643639  4.5515395 24.545079
## f.year=[2008,2016]                  4.4794189  9.9062918 34.160463
## f.price=super cheap                 0.7667032  0.9370817 18.879239
## f.year=(2017,2019]                  4.9302789 13.2530120 41.521919
## f.mileage=f.mil-[1,5.99e+03]        0.3284072  0.5354752 25.186104
##                                          p.value     v.test
## f.year=(2016,2017]                  0.000000e+00        Inf
## f.mileage=f.mil-(1.71e+04,3.45e+04] 2.310709e-214  31.248899
## f.price=expensive                   4.484896e-83  19.309314
## f.tax=f.tax-(145,155]               2.891683e-27  10.815985
## f.price=cheap                       2.843815e-20   9.224690
## f.mpg=f.mpg-(53.3,61.4]             5.589123e-18   8.640656
## transmission=f.Trans-SemiAuto       4.416130e-07   5.050085
## fuelType=f.Fuel-Diesel              3.296065e-06   4.651452
## Audi=Yes                            1.288626e-05   4.362028
## manufacturer=Audi                   1.288626e-05   4.362028
## f.mpg=f.mpg-(61.4,88.3]             2.702791e-04   3.642237
## manufacturer=Mercedes               3.188174e-02   2.145890
## transmission=f.Trans-Manual         6.924598e-03  -2.700449
## transmission=f.Trans-Automatic      5.196992e-03  -2.794563
## Audi=No                             1.288626e-05  -4.362028
## fuelType=f.Fuel-Petrol              3.904096e-06  -4.616424
## f.tax=f.tax-[0,125]                 5.108796e-07  -5.022181
## f.price=very expensive              2.263573e-10  -6.342302
## manufacturer=VW                     5.283027e-13  -7.217813
## f.mileage=f.mil-(5.99e+03,1.71e+04]  2.856700e-16  -8.179227
## f.year=(2019,2020]                  2.911776e-22  -9.703553
## f.mileage=f.mil-(3.45e+04,1.19e+05]  5.510568e-23  -9.871909
## f.tax=f.tax-(155,580]               3.540452e-24 -10.143537
## f.price=extremely expensive         2.356218e-40 -13.298670
## f.mpg=f.mpg-[20,44.8]               7.329743e-56 -15.745886
## f.year=[2008,2016]                  2.382099e-61 -16.526061
## f.price=super cheap                 1.418372e-62 -16.695286
## f.year=(2017,2019]                  1.928791e-73 -18.127635
## f.mileage=f.mil-[1,5.99e+03]        7.830633e-96 -20.771540
##
## $`4`
##                                       Cla/Mod    Mod/Cla    Global
## f.tax=f.tax-(155,580]              96.9121140 98.7893462  8.705542
```

```
## f.year=[2008,2016]                    21.6707022 86.6828087 34.160463
## f.mpg=f.mpg-[20,44.8]                 19.3765796 55.6900726 24.545079
## f.mileage=f.mil-(3.45e+04,1.19e+05] 19.2105263 53.0266344 23.573201
## f.mpg=f.mpg-(44.8,53.3]               13.9588101 44.3099274 27.109181
## transmission=f.Trans-Automatic        14.0575080 42.6150121 25.889165
## manufacturer=BMW                      13.2553606 32.9297821 21.215881
## f.price=expensive                     12.8643216 30.9927361 20.574855
## f.mileage=f.mil-(1.71e+04,3.45e+04] 11.8684844 35.8353511 25.785773
## Audi=Yes                              10.5062082 26.6343826 21.650124
## manufacturer=Audi                     10.5062082 26.6343826 21.650124
## fuelType=f.Fuel-Hybrid                 1.5873016  0.2421308  1.302730
## manufacturer=Mercedes                  6.9875776 21.7917676 26.633581
## Audi=No                                7.9968329 73.3656174 78.349876
## f.price=super cheap                    5.8050383 12.8329298 18.879239
## f.price=extremely expensive            5.8262712 13.3171913 19.520265
## f.year=(2016,2017]                     4.7289504  9.9273608 17.928040
## manufacturer=VW                        5.2203390 18.6440678 30.500414
## f.year=(2019,2020]                     0.0000000  0.0000000  6.389578
## f.mileage=f.mil-(5.99e+03,1.71e+04]  3.3306255  9.9273608 25.454921
## transmission=f.Trans-Manual            4.2161520 17.1912833 34.822167
## f.tax=f.tax-(145,155]                  0.2028398  0.2421308 10.194376
## f.mileage=f.mil-[1,5.99e+03]           0.4105090  1.2106538 25.186104
## f.mpg=f.mpg-(61.4,88.3]                0.0000000  0.0000000 23.738627
## f.mpg=f.mpg-(53.3,61.4]                0.0000000  0.0000000 24.607113
## f.tax=f.tax-[0,125]                    0.0000000  0.0000000 28.908189
## f.year=(2017,2019]                     0.6972112  3.3898305 41.521919
## f.tax=f.tax-(125,145]                  0.1584786  0.9685230 52.191894
##                                            p.value     v.test
## f.tax=f.tax-(155,580]                  0.000000e+00        Inf
## f.year=[2008,2016]                    5.164070e-119  23.194583
## f.mpg=f.mpg-[20,44.8]                  6.111933e-46  14.228335
## f.mileage=f.mil-(3.45e+04,1.19e+05]  3.935950e-42  13.601260
## f.mpg=f.mpg-(44.8,53.3]                3.841220e-15   7.860004
## transmission=f.Trans-Automatic         1.042175e-14   7.734002
## manufacturer=BMW                       6.783607e-09   5.796196
## f.price=expensive                      1.717133e-07   5.227612
## f.mileage=f.mil-(1.71e+04,3.45e+04]  2.274445e-06   4.727372
## Audi=Yes                               1.179631e-02   2.518180
## manufacturer=Audi                      1.179631e-02   2.518180
## fuelType=f.Fuel-Hybrid                 2.768604e-02  -2.201707
## manufacturer=Mercedes                  1.825161e-02  -2.360474
## Audi=No                                1.179631e-02  -2.518180
## f.price=super cheap                    6.574377e-04  -3.406740
## f.price=extremely expensive            5.633847e-04  -3.448656
## f.year=(2016,2017]                     2.537414e-06  -4.705100
## manufacturer=VW                        1.265092e-08  -5.690714
## f.year=(2019,2020]                     4.004354e-13  -7.255414
## f.mileage=f.mil-(5.99e+03,1.71e+04]  1.732099e-16  -8.239308
## transmission=f.Trans-Manual            1.321563e-16  -8.271615
## f.tax=f.tax-(145,155]                  3.325687e-19  -8.957352
## f.mileage=f.mil-[1,5.99e+03]           1.285488e-46 -14.336969
## f.mpg=f.mpg-(61.4,88.3]                6.991681e-52 -15.155280
## f.mpg=f.mpg-(53.3,61.4]                4.627529e-54 -15.481473
```

```
## f.tax=f.tax-[0,125]                    2.910961e-65 -17.060690
## f.year=(2017,2019]                     2.220349e-78 -18.742708
## f.tax=f.tax-(125,145]                 1.119949e-132 -24.511278
##
## $`5`
##                                          Cla/Mod    Mod/Cla    Global
## f.price=super cheap                   64.2935378 83.1444759 18.879239
## transmission=f.Trans-Manual           37.7672209 90.0849858 34.822167
## manufacturer=VW                       37.6949153 78.7535411 30.500414
## f.tax=f.tax-[0,125]                   37.8397711 74.9291785 28.908189
## f.year=[2008,2016]                    30.6295400 71.6713881 34.160463
## fuelType=f.Fuel-Petrol                26.2842893 74.6458924 41.459884
## f.mpg=f.mpg-(53.3,61.4]               32.0168067 53.9660057 24.607113
## f.mileage=f.mil-(3.45e+04,1.19e+05]   25.5263158 41.2181303 23.573201
## f.mileage=f.mil-(1.71e+04,3.45e+04]   21.3311949 37.6770538 25.785773
## Audi=No                               15.4658221 83.0028329 78.349876
## Audi=Yes                              11.4613181 16.9971671 21.650124
## manufacturer=Audi                     11.4613181 16.9971671 21.650124
## f.price=cheap                         11.1902340 15.5807365 20.326716
## fuelType=f.Fuel-Hybrid                 0.0000000  0.0000000  1.302730
## f.mpg=f.mpg-(44.8,53.3]               10.6788711 19.8300283 27.109181
## f.mileage=f.mil-(5.99e+03,1.71e+04]   10.1543461 17.7053824 25.454921
## f.year=(2019,2020]                     0.6472492  0.2832861  6.389578
## f.tax=f.tax-(155,580]                  0.0000000  0.0000000  8.705542
## manufacturer=BMW                       1.7543860  2.5495751 21.215881
## f.price=expensive                      0.9045226  1.2747875 20.574855
## f.mileage=f.mil-[1,5.99e+03]           1.9704433  3.3994334 25.186104
## f.price=extremely expensive            0.0000000  0.0000000 19.520265
## transmission=f.Trans-Automatic         1.3578275  2.4079320 25.889165
## fuelType=f.Fuel-Diesel                 6.4667630 25.3541076 57.237386
## f.price=very expensive                 0.0000000  0.0000000 20.698925
## manufacturer=Mercedes                  0.9316770  1.6997167 26.633581
## f.mpg=f.mpg-[20,44.8]                  0.0842460  0.1416431 24.545079
## transmission=f.Trans-SemiAuto          2.7894737  7.5070822 39.288668
## f.tax=f.tax-(125,145]                  4.4770206 16.0056657 52.191894
## f.year=(2017,2019]                     2.6394422  7.5070822 41.521919
##                                             p.value     v.test
## f.price=super cheap                    0.000000e+00        Inf
## transmission=f.Trans-Manual           2.443854e-244  33.382704
## manufacturer=VW                       2.035270e-184  28.961081
## f.tax=f.tax-[0,125]                   1.759975e-169  27.749640
## f.year=[2008,2016]                    1.431003e-108  22.135754
## fuelType=f.Fuel-Petrol                 1.361886e-83  19.370776
## f.mpg=f.mpg-(53.3,61.4]                4.079119e-75  18.338498
## f.mileage=f.mil-(3.45e+04,1.19e+05]    7.840458e-30  11.345136
## f.mileage=f.mil-(1.71e+04,3.45e+04]    4.110691e-14   7.557443
## Audi=No                                9.154294e-04   3.315306
## Audi=Yes                               9.154294e-04  -3.315306
## manufacturer=Audi                      9.154294e-04  -3.315306
## f.price=cheap                          5.188700e-04  -3.470821
## fuelType=f.Fuel-Hybrid                 4.486956e-05  -4.080859
## f.mpg=f.mpg-(44.8,53.3]                1.386724e-06  -4.826903
## f.mileage=f.mil-(5.99e+03,1.71e+04]    1.341044e-07  -5.273147
```

```
## f.year=(2019,2020]                    1.977128e-19  -9.014532
## f.tax=f.tax-(155,580]                 4.977984e-31 -11.583816
## manufacturer=BMW                      6.481774e-54 -15.459782
## f.price=expensive                     2.746442e-62 -16.655801
## f.mileage=f.mil-[1,5.99e+03]          5.996660e-63 -16.746590
## f.price=extremely expensive           2.083683e-73 -18.123386
## transmission=f.Trans-Automatic        8.542509e-74 -18.172371
## fuelType=f.Fuel-Diesel                3.387255e-77 -18.597170
## f.price=very expensive                2.109092e-78 -18.745443
## manufacturer=Mercedes                 1.560795e-83 -19.363756
## f.mpg=f.mpg-[20,44.8]                 7.770899e-93 -20.437441
## transmission=f.Trans-SemiAuto         4.246431e-94 -20.578849
## f.tax=f.tax-(125,145]                6.334671e-103 -21.541695
## f.year=(2017,2019]                   7.220435e-105 -21.747970
##
## $`6`
##                                         Cla/Mod    Mod/Cla    Global
## f.tax=f.tax-[0,125]                  50.6437768 91.1196911 28.908189
## f.year=[2008,2016]                   42.6755448 90.7335907 34.160463
## f.mpg=f.mpg-(61.4,88.3]              47.3867596 70.0128700 23.738627
## f.mileage=f.mil-(3.45e+04,1.19e+05] 46.5789474 68.3397683 23.573201
## fuelType=f.Fuel-Diesel               26.0476879 92.7927928 57.237386
## f.price=cheap                        41.7090539 52.7670528 20.326716
## manufacturer=Mercedes                24.9223602 41.3127413 26.633581
## f.price=super cheap                  25.9583790 30.5019305 18.879239
## transmission=f.Trans-Automatic       22.9233227 36.9369369 25.889165
## manufacturer=BMW                     23.0019493 30.3732304 21.215881
## fuelType=f.Fuel-Hybrid               44.4444444  3.6036036  1.302730
## f.mpg=f.mpg-(53.3,61.4]              17.8991597 27.4131274 24.607113
## f.price=expensive                    10.7537688 13.7709138 20.574855
## f.year=(2016,2017]                    7.9584775  8.8803089 17.928040
## transmission=f.Trans-SemiAuto        10.8947368 26.6409266 39.288668
## f.tax=f.tax-(145,155]                 2.4340771  1.5444015 10.194376
## f.year=(2019,2020]                    0.0000000  0.0000000  6.389578
## f.tax=f.tax-(155,580]                 0.0000000  0.0000000  8.705542
## f.price=very expensive                2.2977023  2.9601030 20.698925
## manufacturer=VW                       4.6101695  8.7516088 30.500414
## f.mileage=f.mil-(5.99e+03,1.71e+04]  3.2493907  5.1480051 25.454921
## f.price=extremely expensive           0.0000000  0.0000000 19.520265
## f.mpg=f.mpg-(44.8,53.3]               1.5255530  2.5740026 27.109181
## f.mileage=f.mil-[1,5.99e+03]          0.1642036  0.2574003 25.186104
## f.mpg=f.mpg-[20,44.8]                 0.0000000  0.0000000 24.545079
## fuelType=f.Fuel-Petrol                1.3965087  3.6036036 41.459884
## f.tax=f.tax-(125,145]                 2.2583201  7.3359073 52.191894
## f.year=(2017,2019]                    0.1494024  0.3861004 41.521919
##                                            p.value     v.test
## f.tax=f.tax-[0,125]                   0.000000e+00        Inf
## f.year=[2008,2016]                   1.088450e-288  36.310910
## f.mpg=f.mpg-(61.4,88.3]              1.672907e-205  30.589798
## f.mileage=f.mil-(3.45e+04,1.19e+05] 4.286891e-193  29.642107
## fuelType=f.Fuel-Diesel               2.215152e-126  23.913529
## f.price=cheap                        1.797663e-111  22.434887
## manufacturer=Mercedes                 1.958198e-22   9.743936
```

```
## f.price=super cheap                    8.344025e-18   8.594758
## transmission=f.Trans-Automatic         9.857764e-14   7.442794
## manufacturer=BMW                       4.771915e-11   6.577884
## fuelType=f.Fuel-Hybrid                 9.911325e-08   5.328342
## f.mpg=f.mpg-(53.3,61.4]                4.931026e-02   1.965899
## f.price=expensive                      1.089075e-07  -5.311197
## f.year=(2016,2017]                     1.687156e-14  -7.672470
## transmission=f.Trans-SemiAuto          8.959241e-16  -8.040336
## f.tax=f.tax-(145,155]                  8.559289e-25 -10.281269
## f.year=(2019,2020]                     4.334417e-25 -10.346638
## f.tax=f.tax-(155,580]                  2.248083e-34 -12.226353
## f.price=very expensive                 2.418034e-54 -15.523170
## manufacturer=VW                        7.641501e-56 -15.743251
## f.mileage=f.mil-(5.99e+03,1.71e+04]    6.228315e-58 -16.044676
## f.price=extremely expensive            1.731310e-81 -19.119691
## f.mpg=f.mpg-(44.8,53.3]                3.721718e-86 -19.672344
## f.mileage=f.mil-[1,5.99e+03]          2.438775e-104 -21.692049
## f.mpg=f.mpg-[20,44.8]                 7.549333e-106 -21.851335
## fuelType=f.Fuel-Petrol                3.943994e-153 -26.359979
## f.tax=f.tax-(125,145]                 1.445204e-186 -29.131214
## f.year=(2017,2019]                    1.723753e-196 -29.904423
```

- Cluster 1
  - We can see that cars in the first cluster are extremely expensive and sold in the year 2020. Moreover, they have low mpg and mileage. Audi and BMW cars are overrepresented.
- Cluster 2
  - Cars in cluster 2 use to have been sold between 2018 and 2019. Very expensive cars and with low mileage are overrepresented. The avegare amount of Mercedes cars is a little bit larger than the global average (overrepresented).
- Cluster 3
  - Cars in cluster 3 use to have been sold in 2017. They have a mileage close to the mean and half of them are expensive cars (they are overrepresented).
- Cluster 4
  - Cars in cluster 4 use to have been sold between 2008 and 2016. They have higher taxes and mileage than the mean, as well as lower miles per galon. Automatic and BMW cars are overrepresented.
- Cluster 5
  - Cars in cluster 5 use to have been sold between 2008 and 2016. They are characterized for having manual transmission and using petrol. Besides, they have low taxes and are super cheap cars, and use to be VW.
- Cluster 6
  - Cars in cluster 6 use to have been sold between 2008 and 2016. They are characterized for using diesel, having high miles per galon and very high mileage. Besides, they have low taxes and cheap cars are overrepresented.

We now proceed to see the quantitative variables that characterizes the clusters.

```r
res.hcmc$desc.var$quanti.var
```

```
##                       Eta2      P-value
## price           0.54092336 0.000000e+00
## inconsistencies 0.00713398 1.815299e-06
```

The quantitative variable that only characterizes the clusters is **price**, as variable "inconsistencies" has a very low eta2 value. It has to be told that they are the only quantitative variables included in the MCA as supplementary.

```r
res.hcmc$desc.var$quanti
```

```
## $`1`
##         v.test Mean in category Overall mean sd in category Overall sd
## price 36.76681         33004.64     21154.51       9306.978   9563.766
##           p.value
## price 6.2641e-296
##
## $`2`
##                    v.test Mean in category Overall mean sd in category
## price           17.978281     2.493690e+04 2.115451e+04   7.705950e+03
## inconsistencies -3.610123     6.906077e-03 1.840364e-02   8.281536e-02
##                   Overall sd      p.value
## price           9563.7656718 2.882955e-72
## inconsistencies    0.1447753 3.060520e-04
##
## $`3`
##          v.test Mean in category Overall mean sd in category Overall sd
## price -6.324677         19119.26     21154.51       4613.844   9563.766
##            p.value
## price 2.537626e-10
##
## $`4`
## NULL
##
## $`5`
##          v.test Mean in category Overall mean sd in category Overall sd
## price -31.41406         10704.24     21154.51       2672.409   9563.766
##             p.value
## price 1.300486e-216
##
## $`6`
##                     v.test Mean in category Overall mean sd in category
## inconsistencies   5.057476     4.247104e-02 1.840364e-02      0.2199756
## price           -20.891475     1.458703e+04 2.115451e+04   3388.2026118
##                   Overall sd      p.value
## inconsistencies    0.1447753 4.248413e-07
## price           9563.7656718 6.401227e-97
```

- Cluster 1

    - High price, 11846 dollars more than the average price.
- Cluster 2
    - Price 3782 dollars higher than the average price.
- Cluster 3
    - Price 2035 dollars lower than the average price.
- Cluster 4
    - Price very similar to the average price.
- Cluster 5
    - Price 10450 dollars lower than the average price.
- Cluster 6
    - Price 6567 dollars lower than the average price.

7.2 C. The description of the clusters by the individuals

```
res.hcmc$desc.ind$para
```

```
## Cluster: 1
##      3451      4851      4968        38      4127
## 0.1005731 0.2012397 0.2012397 0.2249695 0.2337012
## ------------------------------------------------------------
## Cluster: 2
##     3789     3922     4148     4156     4157
## 0.131819 0.131819 0.131819 0.131819 0.131819
## ------------------------------------------------------------
## Cluster: 3
##      3938      4663       184       476      4556
## 0.2044787 0.2044787 0.2210476 0.2210476 0.2336239
## ------------------------------------------------------------
## Cluster: 4
##      4858      4860      4863      4842      4874
## 0.1262139 0.1262139 0.1262139 0.1494844 0.1494844
## ------------------------------------------------------------
## Cluster: 5
##      3603      3874      4171      4174       689
## 0.2115991 0.2115991 0.2115991 0.2115991 0.2776652
## ------------------------------------------------------------
## Cluster: 6
##      3191      2225      2320      2491      2568
## 0.1627976 0.1773470 0.1773470 0.1773470 0.1773470
```

This command allow us to see for each cluster the top 5 closest individuals to the cluster center. Below each individual it's the distance between each individual and the cluster center.

We are gonna see the values of the variables of these individuals and search for a correlation with the characteristics that describes our clusters.

Cluster 1:

```r
summary(df[c(3451, 4851, 4968, 38, 4127), ])
```

```
##                      model          year           price                        transmission
##  Audi- A5               :1   Min.   :2020   Min.   :24496   f.Trans-Manual   :0
##  Mercedes- A Class      :1   1st Qu.:2020   1st Qu.:25644   f.Trans-SemiAuto :3
##  VW- Passat             :1   Median :2020   Median :36999   f.Trans-Automatic:2
##  VW- Tiguan Allspace:1   Mean   :2020   Mean   :34318
##  VW- Touareg            :1   3rd Qu.:2020   3rd Qu.:39454
##  Audi- A1               :0   Max.   :2020   Max.   :44999
##  (Other)                :0
##     mileage               fuelType        tax             mpg            engineSize
##  Min.   :1005   f.Fuel-Diesel:4   Min.   :145   Min.   :34.50   2        :3
##  1st Qu.:2983   f.Fuel-Petrol:1   1st Qu.:145   1st Qu.:38.70   1.3      :1
##  Median :6000   f.Fuel-Hybrid:0   Median :145   Median :45.60   3        :1
##  Mean   :5183                     Mean   :145   Mean   :45.08   1        :0
##  3rd Qu.:7925                     3rd Qu.:145   3rd Qu.:53.30   1.2      :0
##  Max.   :8000                     Max.   :145   Max.   :53.30   1.4      :0
##                                                                 (Other):0
##    manufacturer                  f.price   Audi   years_after_sell
##  Audi     :1     super cheap         :0   No :4   Min.   :2
##  BMW      :0     cheap               :0   Yes:1   1st Qu.:2
##  Mercedes:1     expensive           :0           Median :2
##  VW       :3     very expensive      :2           Mean   :2
##                  extremely expensive:3           3rd Qu.:2
##                                                  Max.   :2
##
##     f.tax              f.mileage            f.mpg                 f.year
##  Length:5           Length:5           Length:5           [2008,2016]:0
##  Class :character   Class :character   Class :character   (2016,2017]:0
##  Mode  :character   Mode  :character   Mode  :character   (2017,2019]:0
##                                                           (2019,2020]:5
##
##
##
##  inconsistencies         mout    kmeans_clust HCPC_clust claHCMC
##  Min.   :0       MvOut.No :5    1:0          1:5        1:5
##  1st Qu.:0       MvOut.Yes:0    2:0          2:0        2:0
##  Median :0                      3:5          3:0        3:0
##  Mean   :0                      4:0          4:0        4:0
##  3rd Qu.:0                      5:0          5:0        5:0
##  Max.   :0                      6:0                     6:0
##                                                         7:0
```

We can see that all of them were sold in 2020. They use to have semiautomatic transmission and use diesel. Besides, they are or extremely expensive or expensive cars.

Cluster 2:

```r
summary(df[c(3789, 3922, 4148, 4156, 4157), ])
```

```
##          model          year           price                     transmission
##  VW- T-Cross:3   Min.   :2019   Min.   :17997   f.Trans-Manual   :0
```

```
##   VW- Golf   :2   1st Qu.:2019   1st Qu.:17999   f.Trans-SemiAuto :0
##   Audi- A1   :0   Median :2019   Median :19250   f.Trans-Automatic:5
##   Audi- A3   :0   Mean   :2019   Mean   :18846
##   Audi- A4   :0   3rd Qu.:2019   3rd Qu.:19393
##   Audi- A5   :0   Max.   :2019   Max.   :19590
##   (Other)    :0
##      mileage               fuelType       tax             mpg           engineSize
##   Min.   :2648   f.Fuel-Diesel:5   Min.   :145   Min.   :51.40   1.6    :4
##   1st Qu.:3435   f.Fuel-Petrol:0   1st Qu.:145   1st Qu.:51.40   2      :1
##   Median :3800   f.Fuel-Hybrid:0   Median :145   Median :51.40   1      :0
##   Mean   :3897                     Mean   :145   Mean   :51.76   1.2    :0
##   3rd Qu.:4800                     3rd Qu.:145   3rd Qu.:52.30   1.3    :0
##   Max.   :4803                     Max.   :145   Max.   :52.30   1.4    :0
##                                                                  (Other):0
##     manufacturer                  f.price    Audi    years_after_sell
##   Audi    :0     super cheap        :0    No :5    Min.   :3
##   BMW     :0     cheap              :0    Yes:0    1st Qu.:3
##   Mercedes:0     expensive          :5             Median :3
##   VW      :5     very expensive     :0             Mean   :3
##                  extremely expensive:0             3rd Qu.:3
##                                                    Max.   :3
##
##      f.tax              f.mileage             f.mpg                  f.year
##   Length:5            Length:5            Length:5            [2008,2016]:0
##   Class :character    Class :character    Class :character    (2016,2017]:0
##   Mode  :character    Mode  :character    Mode  :character    (2017,2019]:5
##                                                               (2019,2020]:0
##
##
##
##   inconsistencies         mout     kmeans_clust HCPC_clust claHCMC
##   Min.   :0       MvOut.No :5    1:0          1:5        1:0
##   1st Qu.:0       MvOut.Yes:0    2:0          2:0        2:5
##   Median :0                      3:5          3:0        3:0
##   Mean   :0                      4:0          4:0        4:0
##   3rd Qu.:0                      5:0          5:0        5:0
##   Max.   :0                      6:0                     6:0
##                                                          7:0
```

We can see that all of them were sold in 2019. They use to have automatic transmission and use diesel. It can be also highlighted that they have very low mileage, so they little-used cars. Besides, they are expensive cars and all of them are VW.

Cluster 3:

```
summary(df[c(3938, 4663, 184, 476, 4556), ])
```

```
##         model          year          price                     transmission
##   Audi- Q3    :2   Min.   :2017   Min.   :17360   f.Trans-Manual    :0
##   VW- Tiguan:2   1st Qu.:2017   1st Qu.:17695   f.Trans-SemiAuto :3
##   VW- Golf    :1   Median :2017   Median :18058   f.Trans-Automatic:2
##   Audi- A1    :0   Mean   :2017   Mean   :18351
##   Audi- A3    :0   3rd Qu.:2017   3rd Qu.:18899
```

```
##   Audi- A4   :0   Max.   :2017   Max.   :19741
##   (Other)    :0
##     mileage              fuelType        tax            mpg           engineSize
##   Min.   :28458   f.Fuel-Diesel:5   Min.   :145   Min.   :49.60   2       :5
##   1st Qu.:32878   f.Fuel-Petrol:0   1st Qu.:145   1st Qu.:55.40   1       :0
##   Median :33670   f.Fuel-Hybrid:0   Median :145   Median :55.40   1.2     :0
##   Mean   :35381                     Mean   :145   Mean   :55.14   1.3     :0
##   3rd Qu.:35035                     3rd Qu.:145   3rd Qu.:57.60   1.4     :0
##   Max.   :46862                     Max.   :145   Max.   :57.70   1.5     :0
##                                                                   (Other):0
##    manufacturer                   f.price    Audi    years_after_sell
##   Audi    :2     super cheap        :0    No :3   Min.   :5
##   BMW     :0     cheap              :1    Yes:2   1st Qu.:5
##   Mercedes:0     expensive          :4            Median :5
##   VW      :3     very expensive     :0            Mean   :5
##                  extremely expensive:0            3rd Qu.:5
##                                                   Max.   :5
##
##     f.tax             f.mileage            f.mpg                    f.year
##   Length:5           Length:5           Length:5           [2008,2016]:0
##   Class :character   Class :character   Class :character   (2016,2017]:5
##   Mode  :character   Mode  :character   Mode  :character   (2017,2019]:0
##                                                            (2019,2020]:0
##
##
##
##   inconsistencies        mout    kmeans_clust HCPC_clust claHCMC
##   Min.   :0       MvOut.No :5   1:0          1:0        1:0
##   1st Qu.:0       MvOut.Yes:0   2:0          2:5        2:0
##   Median :0                     3:0          3:0        3:5
##   Mean   :0                     4:5          4:0        4:0
##   3rd Qu.:0                     5:0          5:0        5:0
##   Max.   :0                     6:0                     6:0
##                                                         7:0
```

We can see that all of them were sold in 2017. They have automatic or semiautomatic transmission and use diesel. It can be also highlighted that they have high mileage, so they are very used cars. Besides, they are expensive cars.

Cluster 4:

```
summary(df[c(4858, 4860, 4863, 4842, 4874), ])
```

```
##           model          year          price                   transmission
##   VW- Touareg:5   Min.   :2015   Min.   :18191   f.Trans-Manual   :0
##   Audi- A1    :0   1st Qu.:2015   1st Qu.:19990   f.Trans-SemiAuto :3
##   Audi- A3    :0   Median :2015   Median :20490   f.Trans-Automatic:2
##   Audi- A4    :0   Mean   :2015   Mean   :21332
##   Audi- A5    :0   3rd Qu.:2016   3rd Qu.:22495
##   Audi- A6    :0   Max.   :2016   Max.   :25495
##   (Other)     :0
##     mileage              fuelType        tax            mpg          engineSize
##   Min.   :21200   f.Fuel-Diesel:5   Min.   :235   Min.   :42.8   3       :5
```

```
##  1st Qu.:33891   f.Fuel-Petrol:0   1st Qu.:235   1st Qu.:42.8   1       :0
##  Median :35000   f.Fuel-Hybrid:0   Median :235   Median :42.8   1.2     :0
##  Mean   :43167                     Mean   :235   Mean   :42.8   1.3     :0
##  3rd Qu.:59757                     3rd Qu.:235   3rd Qu.:42.8   1.4     :0
##  Max.   :65987                     Max.   :235   Max.   :42.8   1.5     :0
##                                                                 (Other):0
##    manufacturer                   f.price    Audi    years_after_sell
##  Audi    :0     super cheap        :0    No :5    Min.   :6.0
##  BMW     :0     cheap              :0    Yes:0    1st Qu.:6.0
##  Mercedes:0     expensive          :3             Median :7.0
##  VW      :5     very expensive     :2             Mean   :6.6
##                 extremely expensive:0             3rd Qu.:7.0
##                                                   Max.   :7.0
##
##     f.tax             f.mileage             f.mpg                   f.year
##  Length:5           Length:5           Length:5           [2008,2016]:5
##  Class :character   Class :character   Class :character   (2016,2017]:0
##  Mode  :character   Mode  :character   Mode  :character   (2017,2019]:0
##                                                           (2019,2020]:0
##
##
##
##  inconsistencies        mout    kmeans_clust HCPC_clust claHCMC
##  Min.   :0       MvOut.No :5   1:5          1:0        1:0
##  1st Qu.:0       MvOut.Yes:0   2:0          2:5        2:0
##  Median :0                     3:0          3:0        3:0
##  Mean   :0                     4:0          4:0        4:5
##  3rd Qu.:0                     5:0          5:0        5:0
##  Max.   :0                     6:0                     6:0
##                                                        7:0
```

We can see that all of them were sold in 2015 or 2016. They have automatic or semiautomatic transmission and use diesel. It can be also highlighted that they have high mileage, so they are very used cars. Besides, they are expensive or very expensive cars, and all of them are VW.

Cluster 5:

```r
summary(df[c(3603, 3874, 4171, 4174, 689), ])
```

```
##       model         year          price                   transmission
##  VW- Golf:2   Min.   :2015   Min.   :10495   f.Trans-Manual   :1
##  VW- Polo:2   1st Qu.:2015   1st Qu.:10991   f.Trans-SemiAuto :4
##  Audi- A1:1   Median :2016   Median :10998   f.Trans-Automatic:0
##  Audi- A3:0   Mean   :2016   Mean   :11535
##  Audi- A4:0   3rd Qu.:2016   3rd Qu.:12495
##  Audi- A5:0   Max.   :2016   Max.   :12695
##  (Other) :0
##     mileage                fuelType       tax           mpg           engineSize
##  Min.   :11180   f.Fuel-Diesel:0   Min.   :20   Min.   :55.40   1.4     :4
##  1st Qu.:26278   f.Fuel-Petrol:5   1st Qu.:20   1st Qu.:56.50   1.2     :1
##  Median :27704   f.Fuel-Hybrid:0   Median :30   Median :56.50   1       :0
##  Mean   :25934                     Mean   :26   Mean   :57.98   1.3     :0
##  3rd Qu.:30552                     3rd Qu.:30   3rd Qu.:60.10   1.5     :0
```

```
##  Max.   :33958                      Max.   :30   Max.   :61.40   1.6    :0
##                                                                  (Other):0
##    manufacturer                   f.price    Audi    years_after_sell
##  Audi    :1      super cheap         :5    No :4    Min.   :6.0
##  BMW     :0      cheap               :0    Yes:1    1st Qu.:6.0
##  Mercedes:0      expensive           :0             Median :6.0
##  VW      :4      very expensive      :0             Mean   :6.4
##                  extremely expensive:0              3rd Qu.:7.0
##                                                     Max.   :7.0
##
##     f.tax             f.mileage            f.mpg                  f.year
##  Length:5           Length:5           Length:5           [2008,2016]:5
##  Class :character   Class :character   Class :character   (2016,2017]:0
##  Mode  :character   Mode  :character   Mode  :character   (2017,2019]:0
##                                                           (2019,2020]:0
##
##
##
##  inconsistencies         mout    kmeans_clust HCPC_clust claHCMC
##  Min.   :0        MvOut.No :5    1:0          1:0        1:0
##  1st Qu.:0        MvOut.Yes:0    2:0          2:0        2:0
##  Median :0                       3:0          3:0        3:0
##  Mean   :0                       4:1          4:5        4:0
##  3rd Qu.:0                       5:4          5:0        5:5
##  Max.   :0                       6:0                     6:0
##                                                          7:0
```

We can see that all of them were sold in 2015 or 2016. They use to have semiautomatic transmission and use petrol. It can be also highlighted that they have very low taxes. Besides, they are super cheap cars.

Cluster 6:

```
summary(df[c(3191, 2225, 2320, 2491, 2568), ])
```

```
##                   model        year           price                  transmission
##  Mercedes- E Class   :2   Min.   :2013   Min.   :12499   f.Trans-Manual   :0
##  Mercedes- A Class   :1   1st Qu.:2015   1st Qu.:14991   f.Trans-SemiAuto :4
##  Mercedes- C Class   :1   Median :2016   Median :15995   f.Trans-Automatic:1
##  Mercedes- GLA Class:1    Mean   :2015   Mean   :15387
##  Audi- A1            :0   3rd Qu.:2016   3rd Qu.:16452
##  Audi- A3            :0   Max.   :2016   Max.   :16998
##  (Other)             :0
##     mileage               fuelType       tax            mpg            engineSize
##  Min.   :18061   f.Fuel-Diesel:5   Min.   :125   Min.   :56.60   2.1    :5
##  1st Qu.:43883   f.Fuel-Petrol:0   1st Qu.:125   1st Qu.:58.90   1      :0
##  Median :52485   f.Fuel-Hybrid:0   Median :125   Median :58.90   1.2    :0
##  Mean   :46888                     Mean   :125   Mean   :58.68   1.3    :0
##  3rd Qu.:59080                     3rd Qu.:125   3rd Qu.:58.90   1.4    :0
##  Max.   :60931                     Max.   :125   Max.   :60.10   1.5    :0
##                                                                  (Other):0
##    manufacturer                   f.price    Audi    years_after_sell
##  Audi    :0      super cheap         :1    No :5    Min.   :6.0
##  BMW     :0      cheap               :4    Yes:0    1st Qu.:6.0
```

```
##  Mercedes:5     expensive          :0           Median :6.0
##  VW      :0     very expensive     :0           Mean   :6.8
##                 extremely expensive:0           3rd Qu.:7.0
##                                                 Max.   :9.0
##
##     f.tax            f.mileage           f.mpg                    f.year
##  Length:5           Length:5           Length:5           [2008,2016]:5
##  Class :character   Class :character   Class :character   (2016,2017]:0
##  Mode  :character   Mode  :character   Mode  :character   (2017,2019]:0
##                                                           (2019,2020]:0
##
##
##
##  inconsistencies        mout   kmeans_clust HCPC_clust claHCMC
##  Min.   :0       MvOut.No :5   1:5          1:0        1:0
##  1st Qu.:0       MvOut.Yes:0   2:0          2:5        2:0
##  Median :0                     3:0          3:0        3:0
##  Mean   :0                     4:0          4:0        4:0
##  3rd Qu.:0                     5:0          5:0        5:0
##  Max.   :0                     6:0                     6:5
##                                                        7:0
```

We can see that all of them were sold in 2013, 2015 or 2016. They use to have semiautomatic transmission and use diesel. It can be also highlighted that they have high mileage, so they are very used cars. Besides, they use to be cheap cars, and all of them are Mercedes.

7.3 Comparison of clusters obtained after K-Means (based on PCA)

In general, we get similar results from our Kmeans using PCA and from our HCMC. From our HCMC we get 5 clusters characterized by the price, year of selling and mileage of the cars where in the Kmeans we get 3 clusters where those variables are also a main quality. From our HCMC we also get a cluster characterized by high taxes and mileage and low mpg (cluster 4). Low miles per gallon means high consumption so therefore that is why we could see higher taxes in those cars. In contrast we did not see such cluster in the Kmeans, but we did see a cluster, for example, that was characterized by its transmission, fuel and type of engine (cluster 2).