

Entrega Final de ADEI

Compilación de los entregables 1,2 y 3

Alejandro Alarcón

10/19/2021

Contents

1	Introducción al dataset	5
1.1	Transformación de variables categóricas a factores	6
1.2	Transformación de variables numéricas a factores	7
1.3	Exploración de las variables	7
1.3.1	Factores	7
1.3.2	Variables numéricas	9
2	Por cada variable	13
2.1	Conteo de missings	13
2.2	Conteo de outliers	15
2.3	Conteo de errores	18
3	Imputación	19
3.1	Imputación de variables numéricas	20
3.2	Imputación de factores	22
4	Discretización de variables numéricas en factores	23
4.1	Mileage	24
4.2	Tax	25
4.3	Mpg	26
4.3.1	Age	27
4.3.2	Price	28
4.4	Generación del target categórico Audi	29
5	Identificación los outliers multivariantes	30
6	Profiling	33
6.1	Target numérico (Price)	33
6.2	Target factor (AUDI)	34
7	Análisis de Componentes Principales	38
7.1	Análisis según los individuos	41
7.2	Análisis según las variables	43
7.2.1	Variables numéricas	43
7.2.2	Targets	45
7.2.3	Factores	47

8 Hierachical Clustering	48
8.1 Análisis según las variables	51
8.1.1 Factores	51
8.1.2 Variables numéricas	52
8.2 Análisis según los inividuos	53
9 K-means Clustering desde ACP	56
10 Correspondence Analysis	61
11 Multiple Correspondence Analysis	66
11.1 Análisis segun las variables	69
11.1.1 Factores	69
11.1.2 Variables numéricas	71
12 Clustering Jerárquico desde MCA	73
12.1 Análisis según las variables	79
12.1.1 Factores	79
12.1.2 Variables numéricas	79
12.2 Análisis según las componentes del MCA	81
12.3 Análisis según individuos	84
13 K-Means Clustering desde MCA	86
13.1 Profiling de la clusterización	87
13.2 Variables explicativas numéricas	90
14 Modelo de regresión lineal	90
14.1 Variables numéricas	90
14.2 Factores	109
14.3 Interacciones	113
14.3.1 Interacciones entre factores	113
14.3.2 Interacciones Factor-Numéricas	114
15 Modelo de regresión Binaria	119
15.1 Variables numéricas	120
15.2 Factores	121
15.3 Interacciones	128
15.4 Diagnóstico	133
15.5 Bondad del ajuste y capacidad de predicción	139
15.6 Matriz de confusión	142

```
Loading required package: effects

Loading required package: carData

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

Loading required package: FactoMineR

Loading required package: car

Loading required package: factoextra

Loading required package: ggplot2

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

Loading required package: RColorBrewer

Loading required package: dplyr

Attaching package: 'dplyr'

The following object is masked from 'package:car':
  recode

The following objects are masked from 'package:stats':
  filter, lag

The following objects are masked from 'package:base':
  intersect, setdiff, setequal, union

Loading required package: ggmap

Google's Terms of Service: https://cloud.google.com/maps-platform/terms/.

Please cite ggmap if you use it! See citation("ggmap") for details.

Loading required package: ggthemes

Loading required package: missMDA

Loading required package: epiDisplay

Loading required package: foreign

Loading required package: survival

Loading required package: MASS

Attaching package: 'MASS'

The following object is masked from 'package:dplyr':
  select
```

```
Loading required package: nnet

Attaching package: 'epiDisplay'

The following object is masked from 'package:ggplot2':

alpha
```

```
Loading required package: games

Loading required package: maxLik

Loading required package: miscTools
```

Please cite the 'maxLik' package as:

Henningsen, Arne and Toomet, Ott (2011). maxLik: A package for maximum likelihood estimation in R. Compu

If you have questions, suggestions, or comments regarding the 'maxLik' package, please use a forum or 't
<https://r-forge.r-project.org/projects/maxlik/>

```
Loading required package: Formula
```

1 Introducción al dataset

Echamos un vistazo al dataset

```
summary( df )
```

```
model                  year          price      transmission
Length:5000      Min.   :2001   Min.   : 899  Length:5000
Class :character  1st Qu.:2016  1st Qu.:13995 Class  :character
Mode  :character  Median :2017  Median :19498 Mode   :character
                           Mean   :2017  Mean   :21207
                           3rd Qu.:2019 3rd Qu.:25980
                           Max.  :2020  Max.  :109495

mileage        fuelType          tax      mpg
Min.   : 1   Length:5000      Min.   : 0.0  Min.   : 1.10
1st Qu.: 5815  Class :character  1st Qu.:125.0 1st Qu.: 45.60
Median :17731  Mode  :character  Median :145.0  Median : 53.30
Mean   :23590                           Mean   :122.8  Mean   : 53.93
3rd Qu.:34130                           3rd Qu.:145.0 3rd Qu.: 61.40
Max.  :178000                          Max.  :580.0  Max.  :470.80

engineSize     manufacturer
Min.   :0.000  Length:5000
1st Qu.:1.500  Class :character
Median :2.000  Mode  :character
Mean   :1.909
3rd Qu.:2.000
Max.  :5.500
```

```
names( df )
```

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

1.1 Transformación de variables categóricas a factores

```
#Model
```

```
df$model <- factor(paste0(df$manufacturer, "-", df$model))  
head(levels(df$model)) #Algunos de los valores para el factor modelo
```

```
[1] "Audi- A1" "Audi- A3" "Audi- A4" "Audi- A5" "Audi- A6" "Audi- A7"
```

```
#Transmission
```

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

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

```
df$transmission <- factor( df$transmission, levels = c("Manual","Semi-Auto","Automatic"),labels = paste0
```

```
head( df )
```

	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize
1	Audi- A1	2017	12500	f.Trans-Manual	15735	Petrol	150	55.4	1.4
6	Audi- A1	2016	13900	f.Trans-Automatic	32260	Petrol	30	58.9	1.4
9	Audi- A3	2015	10200	f.Trans-Manual	46112	Petrol	20	60.1	1.4
23	Audi- A5	2017	22500	f.Trans-Automatic	21649	Diesel	145	58.9	3.0
25	Audi- Q5	2016	20000	f.Trans-Automatic	23789	Diesel	200	47.1	2.0
38	Audi- A6	2016	19400	f.Trans-Automatic	34030	Diesel	125	58.9	2.0

manufacturer

	manufacturer
1	Audi
6	Audi
9	Audi
23	Audi
25	Audi
38	Audi

```
#FuelType
```

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

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

```
df$fuelType <- factor( df$fuelType, levels = c("Diesel","Petrol","Hybrid"), labels = paste0("f.Fuel-",c
```

```
#Manufacturer
```

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

```
[1] "Audi"      "BMW"       "Mercedes"  "VW"
```

1.2 Transformación de variables numéricas a factores

```
#Year + Age
```

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

```
Min. 1st Qu. Median Mean 3rd Qu. Max.  
2001 2016 2017 2017 2019 2020
```

```
df$age <- 2021 - df$year
```

```
df$year<-factor(df$year)
```

```
summary(df$age)
```

```
Min. 1st Qu. Median Mean 3rd Qu. Max.  
1.000 2.000 4.000 3.843 5.000 20.000
```

```
#EngineSize
```

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

```
Min. 1st Qu. Median Mean 3rd Qu. Max.  
0.000 1.500 2.000 1.909 2.000 5.500
```

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

```
table(df$engineSize)
```

```
0 1 1.2 1.3 1.4 1.5 1.6 1.8 1.9 2 2.1 2.2 2.3 2.5 2.7 2.9  
9 365 147 63 310 554 345 38 1 2142 412 20 4 7 1 10  
3 3.2 3.5 3.7 4 4.2 4.4 4.7 5 5.5  
512 3 2 1 36 4 7 2 1 4
```

1.3 Exploración de las variables

1.3.1 Factores

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

```
tab1(df$year)
```

```
df$year :  
Frequency Percent Cum. percent  
2001 3 0.1 0.1  
2002 1 0.0 0.1  
2003 1 0.0 0.1  
2004 5 0.1 0.2  
2005 3 0.1 0.3  
2006 9 0.2 0.4  
2007 7 0.1 0.6  
2008 5 0.1 0.7  
2009 9 0.2 0.9  
2010 14 0.3 1.1  
2011 16 0.3 1.5  
2012 38 0.8 2.2  
2013 129 2.6 4.8  
2014 217 4.3 9.1  
2015 424 8.5 17.6  
2016 885 17.7 35.3  
2017 895 17.9 53.2  
2018 460 9.2 62.4  
2019 1563 31.3 93.7  
2020 316 6.3 100.0  
Total 5000 100.0 100.0
```

```
tab1(df$engineSize)
```

df\$engineSize :

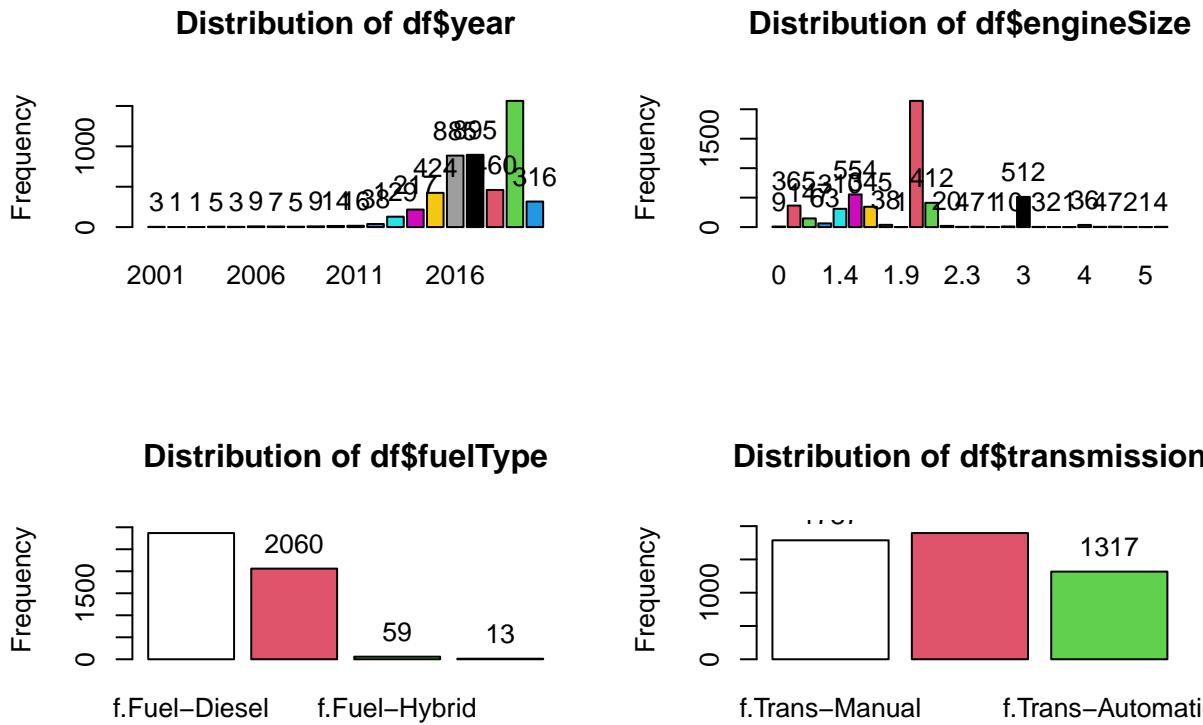
	Frequency	Percent	Cum. percent
0	9	0.2	0.2
1	365	7.3	7.5
1.2	147	2.9	10.4
1.3	63	1.3	11.7
1.4	310	6.2	17.9
1.5	554	11.1	29.0
1.6	345	6.9	35.9
1.8	38	0.8	36.6
1.9	1	0.0	36.6
2	2142	42.8	79.5
2.1	412	8.2	87.7
2.2	20	0.4	88.1
2.3	4	0.1	88.2
2.5	7	0.1	88.3
2.7	1	0.0	88.4
2.9	10	0.2	88.6
3	512	10.2	98.8
3.2	3	0.1	98.9
3.5	2	0.0	98.9
3.7	1	0.0	98.9
4	36	0.7	99.6
4.2	4	0.1	99.7
4.4	7	0.1	99.9
4.7	2	0.0	99.9
5	1	0.0	99.9
5.5	4	0.1	100.0
Total	5000	100.0	100.0

```
tab1(df$fuelType)
```

df\$fuelType :

	Frequency	%(NA+)	%(NA-)
f.Fuel-Diesel	2868	57.4	57.5
f.Fuel-Petrol	2060	41.2	41.3
f.Fuel-Hybrid	59	1.2	1.2
NA's	13	0.3	0.0
Total	5000	100.0	100.0

```
tab1(df$transmission)
```



```
df$transmission :
      Frequency Percent Cum. percent
f.Trans-Manual      1787    35.7      35.7
f.Trans-SemiAuto   1896    37.9      73.7
f.Trans-Automatic  1317    26.3     100.0
  Total            5000   100.0     100.0
```

1.3.2 Variables numéricas

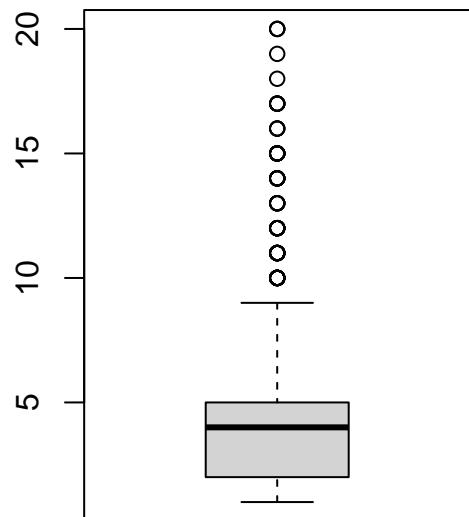
```
par(mfrow = c(1,2))
summary(df[,c("age", "price", "mileage", "tax", "mpg")])
```

age	price	mileage	tax
Min. : 1.000	Min. : 899	Min. : 1	Min. : 0.0
1st Qu.: 2.000	1st Qu.: 13995	1st Qu.: 5815	1st Qu.: 125.0
Median : 4.000	Median : 19498	Median : 17731	Median : 145.0
Mean : 3.843	Mean : 21207	Mean : 23590	Mean : 122.8
3rd Qu.: 5.000	3rd Qu.: 25980	3rd Qu.: 34130	3rd Qu.: 145.0
Max. : 20.000	Max. : 109495	Max. : 178000	Max. : 580.0

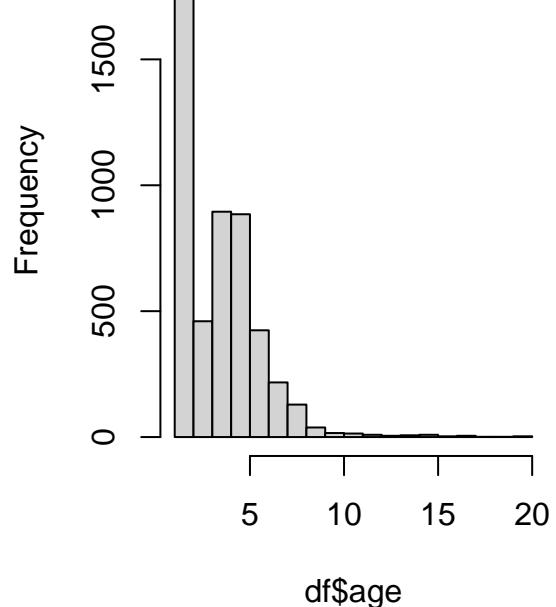
mpg
Min. : 1.10
1st Qu.: 45.60
Median : 53.30
Mean : 53.93
3rd Qu.: 61.40
Max. : 470.80

```
boxplot( df$age, main="Age" )
hist( df$age )
```

Age

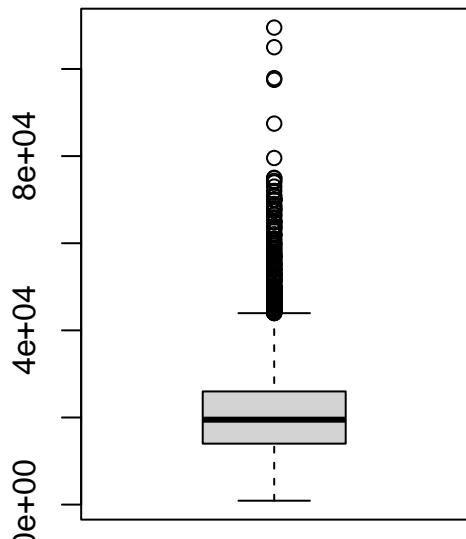


Histogram of df\$age

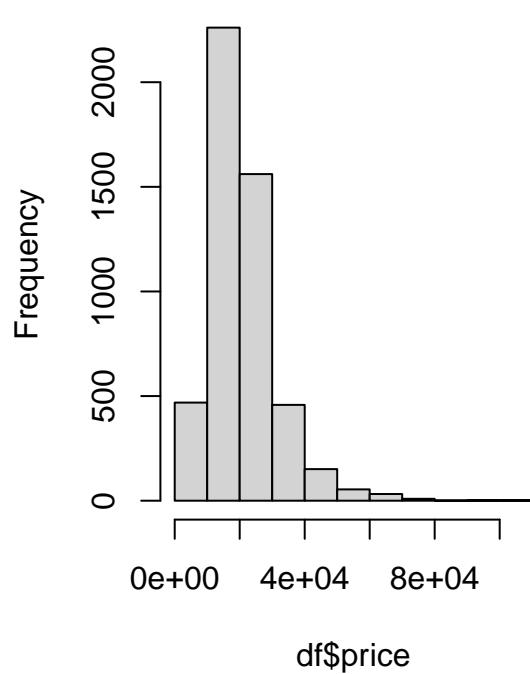


```
boxplot( df$price, main="price" )
hist( df$price )
```

price

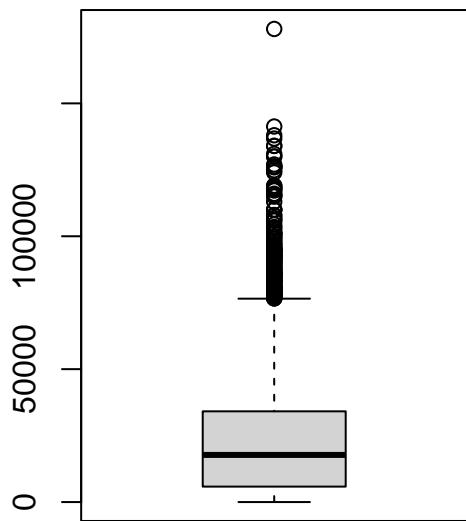


Histogram of df\$price

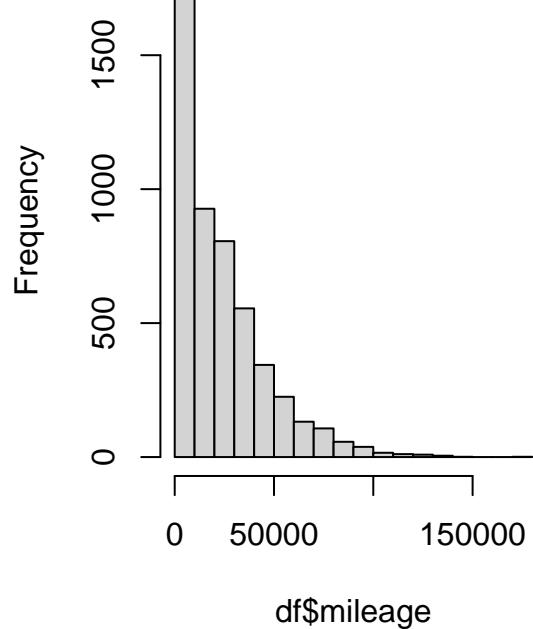


```
boxplot( df$mileage,main="mileage" )
hist( df$mileage )
```

mileage

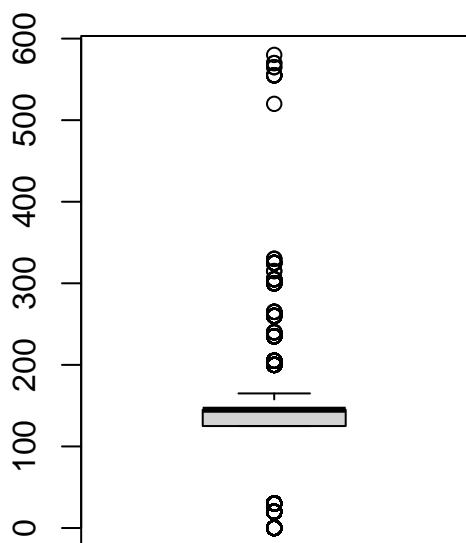


Histogram of df\$mileage

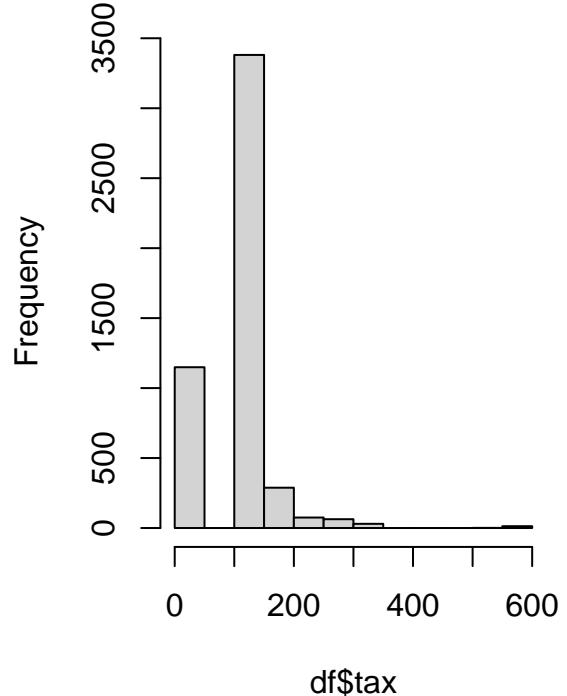


```
boxplot( df$tax, main="tax")
hist(df$tax)
```

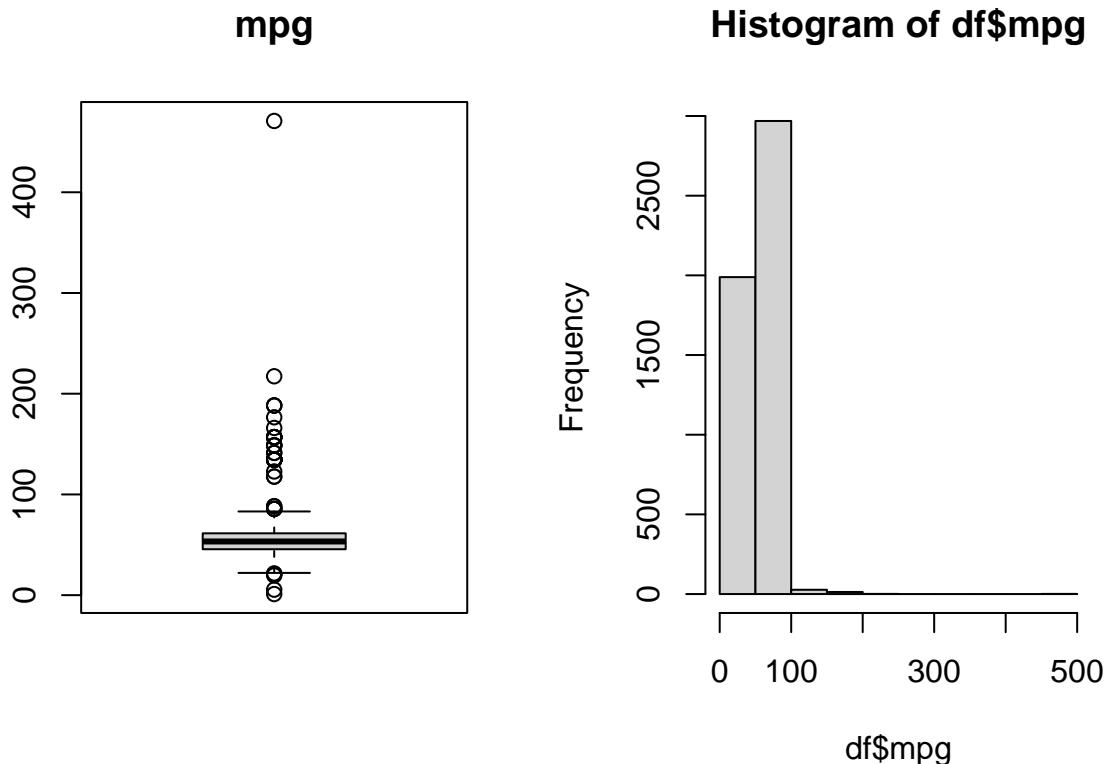
tax



Histogram of df\$tax



```
boxplot( df$mpg, main="mpg")
hist(df$mpg)
```



Funciones útiles

```

calcQ <- function(x) {
  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) {
  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=miss_x,mis_ind=miss_i) }

countX <- function(x,X) {
  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) }

```

2 Por cada variable

2.1 Conteo de missings

Iniciamos el recuento de missings creando dos estructuras de datos auxiliares y llamando a la función countNA. Como podemos apreciar, nos aparecen 13 individuos con valores de missing.

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

mis1<-countNA(df)
imis<-mis1$mis_ind
inds <- which(imis > 0)
inds
```

```
[1] 2123 2131 3487 4034 4056 4495 4496 4509 4682 4782 4783 4874 4875
```

Con el siguiente comando podemos ver todos los individuos que hemos seleccionado que contenían algún miss. En este caso, todos los miss están agrupados en la misma variable: fuelType. Podríamos pensar que esto podría ser debido a la existencia de coches eléctricos, pero como podemos apreciar, en algunos casos, el tipo de cambio es manual, hecho que no se aplica a los coches eléctricos.

```
df[inds,]
```

		model	year	price	transmission	mileage	fuelType	tax	mpg
21018		BMW- 3 Series	2017	15300	f.Trans-Automatic	39428	<NA>	0	148.7
21110		BMW- 3 Series	2017	16000	f.Trans-Automatic	47495	<NA>	135	134.5
34344		Mercedes- C Class	2020	40999	f.Trans-Automatic	400	<NA>	135	217.3
39938		VW- Golf	2019	16889	f.Trans-Manual	12954	<NA>	150	45.6
40147		VW- Golf	2016	13795	f.Trans-Automatic	24463	<NA>	30	53.3
44517		VW- Polo	2019	12889	f.Trans-Manual	13016	<NA>	145	48.7
44524		VW- Polo	2019	13649	f.Trans-Manual	5000	<NA>	145	48.7
44653		VW- Polo	2019	14995	f.Trans-Automatic	10763	<NA>	145	45.6
46378		VW- Tiguan	2019	24999	f.Trans-Automatic	8491	<NA>	145	36.2
47541		VW- Up	2015	6799	f.Trans-Manual	28291	<NA>	20	62.8
47543		VW- Up	2020	10899	f.Trans-Manual	5000	<NA>	145	54.3
48422		VW- Touareg	2014	20995	f.Trans-Automatic	30523	<NA>	300	39.2
48423		VW- Touareg	2015	19995	f.Trans-Automatic	59115	<NA>	235	42.8
		engineSize	manufacturer	age					
21018		2	BMW	4					
21110		2	BMW	4					
34344		2	Mercedes	1					
39938		1	VW	2					
40147		1.4	VW	5					
44517		1	VW	2					
44524		1	VW	2					
44653		1	VW	2					
46378		1.5	VW	2					
47541		1	VW	6					
47543		1	VW	1					
48422		3	VW	7					
48423		3	VW	6					

Por último, como podemos ver en el recuento de misses por variable, se puede apreciar como todos los misses se acumulan en la variable que hemos comentado anteriormente: fuelType.

```
mis1$mis_col # Number of missings for the current set of variables
```

	mis_x
model	0
year	0

```
price          0
transmission   0
mileage         0
fuelType        13
tax             0
mpg            0
engineSize      0
manufacturer    0
age             0
```

En conclusión, solo aparecen un total de 13 missing values en todo el dataframe. Si miramos por variables, estos 13 missings aparecen en la columna de fuelType. Por otro lado, y si miramos por individuos, podemos ver como para los 13 individuos que tienen missings, este está en la columna de fuelType.

2.2 Conteo de outliers

Iniciamos el recuento de outliers creando dos estructuras de datos auxiliares y una función que retornara los individuos con extreme outliers y mild outliers por separado.

```
iouts<-rep(0,nrow(df)) # rows - trips
jouts<-rep(0,2*ncol(df)) # columns - variables

# Funcion que recibe como parametro una columna y devuelve los ids de los individuos outlier
outliers_column <- function(x){
  outs <- NULL
  out_bounds <- calcQ(x)
  ex <- which((x<out_bounds$souti)|(x>out_bounds$souts))
  mild <- which((x<out_bounds$mouti)|(x>out_bounds$mouts))

  boxplot(x)
  abline(h=out_bounds$mouti,col="orange")
  abline(h=out_bounds$mouts,col="orange")
  abline(h=out_bounds$souti,col="red")
  abline(h=out_bounds$souts,col="red")

  outs <- rep(0,length(x))
  outs[mild] <- 1
  outs[ex] <- 2
  if(length(ex)==0){
    outs <- factor(outs, labels=c("Non-Outlier","Mild-Outlier"))
  }else if(length(mild)==0){
    outs <- factor(outs, labels=c("Non-Outlier","Extreme-Outlier"))
  }else{
    outs <- factor(outs, labels=c("Non-Outlier","Mild-Outlier","Extreme-Outlier"))
  }
  list(extreme=ex, mild=mild, outs=outs)
}
```

Aplicaremos la función definida previamente a nuestras columnas numéricas.

```
# Estos son los outliers tanto mild como extreme de las variables numéricas
par(mfrow = c(1, 5))
outs_price <- outliers_column(df$price)
length(outs_price$mild) + length(outs_price$extreme)
```

```
[1] 235
```

```
outs_mileage <- outliers_column(df$mileage)
length(outs_mileage$mild) + length(outs_mileage$extreme)
```

```
[1] 187
```

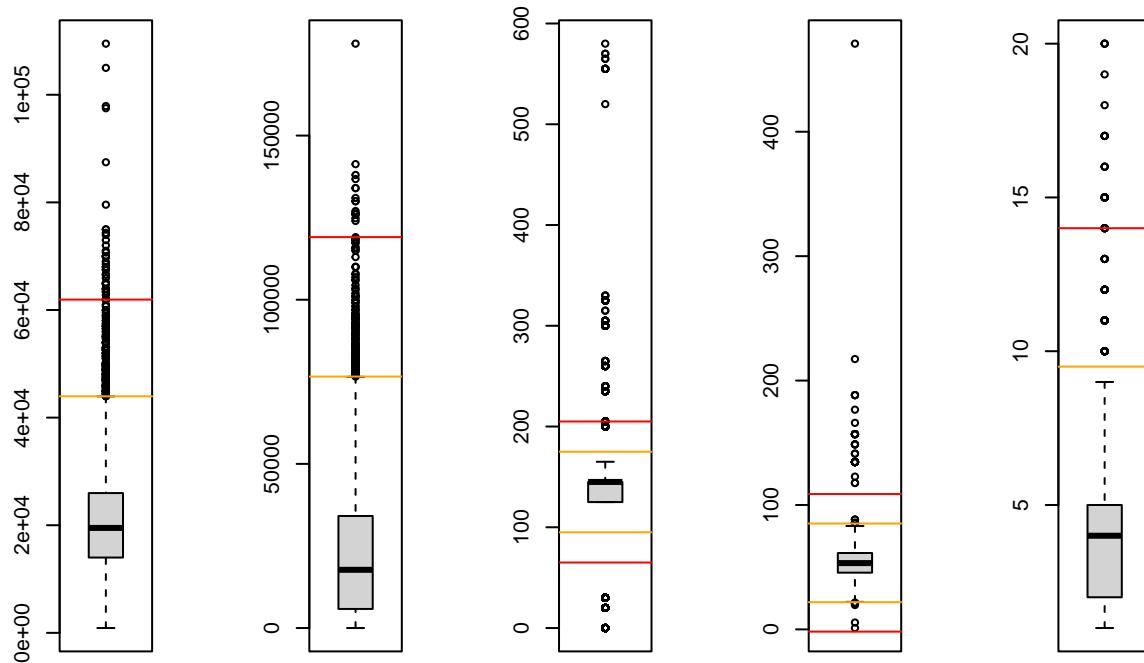
```
outs_tax <- outliers_column(df$tax)
length(outs_tax$mild) + length(outs_tax$extreme)
```

```
[1] 2756
```

```
outs_mpg <- outliers_column(df$mpg)
length(outs_mpg$mild)
```

```
[1] 54
```

```
outs_age <- outliers_column(df$age)
```



```
length(outs_age$mild) + length(outs_age$extreme)
```

```
[1] 95
```

Generamos la suma de todos los outliers.

```
df$outts <- rep(0,nrow(df))
df$outts[which(outs_mileage$outts!="Non-Outlier")] <- df$outts[which(outs_mileage$outts!="Non-Outlier")] + 1
df$outts[which(outs_price$outts!="Non-Outlier")] <- df$outts[which(outs_price$outts!="Non-Outlier")] + 1
df$outts[which(outs_tax$outts!="Non-Outlier")] <- df$outts[which(outs_tax$outts!="Non-Outlier")] + 1
df$outts[which(outs_mpg$outts!="Non-Outlier")] <- df$outts[which(outs_mpg$outts!="Non-Outlier")] + 1
df$outts[which(outs_age$outts!="Non-Outlier")] <- df$outts[which(outs_age$outts!="Non-Outlier")] + 1

summary(df$outts)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.00	0.00	0.00	0.39	1.00	4.00

```
# Conteo de extreme outliers
```

```
sum_extreme_outliers <- length(outs_price$extreme) + length(outs_mileage$extreme) + length(outs_tax$extreme)
sum_extreme_outliers
```

```
[1] 1419
```

```
# Conteo de mild outliers
```

```
sum_mild_outliers <- length(outs_price$mild) + length(outs_mileage$mild) + length(outs_tax$mild) + length(outs_mpg$mild)
sum_mild_outliers
```

```
[1] 1950
```

Como podemos apreciar, una vez realizado el sumatorio de los diferentes tipos de outliers para todo el dataframe, vemos que el número de extreme outliers es mayor al de mild outliers.

Ponemos los extreme outliers como missings.

```
df [outs_mileage$extreme, "mileage"] <-NA  
df [outs_tax$extreme, "tax"] <-NA  
df [outs_mpg$extreme, "mpg"] <-NA  
df [outs_age$extreme, "age"] <-NA
```

2.3 Conteo de errores

Procedemos al conteo de los errores.

Para este apartado, en la mayoría de los casos, simplemente comprobamos que los valores insertados no sean negativos o cero, aunque tal vez se podrían tener en cuenta otros casos si se examina el dataframe de manera técnica (i.e. consumos o distancias exageradamente elevados).

```
# Price
err_price <- which(df$price<0)
#df <- df[ -err_price, ]      #En este caso está vacío, así que no es necesario
length(err_price)
```

[1] 0

```
# Age
err_age <- which(df$age<0)
df[err_age, "age"] <- NA
length(err_age)
```

[1] 0

```
# Mileage
err_mileage <- which(df$mileage<0)
df[err_mileage, "mileage"] <- NA
length(err_mileage)
```

[1] 0

```
# mpg
err_mpg <- which(df$mpg<0)
df[err_mpg, "mpg"] <- NA
length(err_mpg)
```

[1] 0

```
# engineSize
err_engineSize <- which(df$engineSize == "0")
df[err_engineSize, "engineSize"] <- NA
length(err_engineSize)
```

[1] 9

```
# tax
err_tax <- which(df$tax<0)
df[err_tax, "tax"] <- NA
length(err_tax)
```

[1] 0

3 Imputación

A continuación, vamos a proceder a la imputación de los missings, errors y extreme outliers que hemos encontrado previamente.

En primer lugar, separaremos nuestras variables en numéricas y categóricas. También declaramos price y manufacturer como variables respuesta.

```
library(missMDA)
names(df)

[1] "model"          "year"           "price"          "transmission"   "mileage"
[6] "fuelType"        "tax"            "mpg"            "engineSize"     "manufacturer"
[11] "age"             "outs"

vars_num <- names(df)[c(5,7,8,11)]
vars_cat <- names(df)[c(1,2,4,6,9)]
vars_res <- names(df)[c(3,10)]
```

3.1 Imputación de variables numéricas

Podemos echar un pequeño vistazo a nuestras variables numéricas.

```
summary(df[,vars_num])
```

mileage	tax	mpg	age
Min. : 1	Min. :125	Min. : 1.10	Min. : 1.000
1st Qu.: 5800	1st Qu.:145	1st Qu.:45.60	1st Qu.: 2.000
Median : 17632	Median :145	Median :53.30	Median : 4.000
Mean : 23238	Mean :147	Mean :53.07	Mean : 3.786
3rd Qu.: 34000	3rd Qu.:145	3rd Qu.:61.40	3rd Qu.: 5.000
Max. :119000	Max. :205	Max. :88.30	Max. :14.000
NA's :16	NA's :1297	NA's :42	NA's :22

Aplicamos la función imputePCA con 2 componentes primarias. (Teniendo en cuenta el PCA que realizamos en laboratorios posteriores, dos componentes son suficientes para aglutinar más del 80% de la variabilidad).

```
res.impca <- imputePCA(df[,vars_num], ncp = 2)
summary(res.impca$completeObs)
```

mileage	tax	mpg	age
Min. : 1	Min. :125.0	Min. : 1.1	Min. : 1.000
1st Qu.: 5807	1st Qu.:145.0	1st Qu.:45.6	1st Qu.: 2.000
Median : 17706	Median :145.0	Median :53.3	Median : 4.000
Mean : 23318	Mean :146.9	Mean :53.1	Mean : 3.798
3rd Qu.: 34095	3rd Qu.:147.5	3rd Qu.:61.4	3rd Qu.: 5.000
Max. :119000	Max. :205.0	Max. :88.3	Max. :14.000

Vamos a ver para cada variable, el valor que acaban recibiendo los individuos que hemos inputado:

```
head(res.impca$completeObs[ outs_age$extreme, "age" ])
```

```
9880      10553     17953     19848     20506     20544
5.904202 8.309153 8.394285 2.174381 4.318005 6.577762
```

```
head(res.impca$completeObs[ outs_mileage$extreme, "mileage" ])
```

```
18967      18980     19848     19860     19884     20904
57572.874 59283.687 5550.296 37522.427 34222.870 47606.733
```

```
head(res.impca$completeObs[ outs_tax$extreme, "tax" ])
```

```
6         9        41        54        70        72
146.2654 148.8502 147.3746 146.4441 139.2268 144.8021
```

```
head(res.impca$completeObs[ outs_mpg$extreme, "mpg" ])
```

```
5148      6308     8876     11341     11561     14452
51.71888 51.10256 60.79850 55.06192 54.69687 59.37574
```

Sustituimos en el dataframe original:

```
df[, vars_num] <- res.impca$completeObs
```

Como podemos ver, en los valores que previamente teníamos NA, ahora aparecen los valores que se han obtenido de la imputación.

```
df[outs_age$extreme, "age"]
```

```
[1] 5.904202 8.309153 8.394285 2.174381 4.318005 6.577762 6.740687 7.283228  
[9] 8.193843 4.372187 7.933087 6.730148 6.096680 8.645744 9.221560 4.991256  
[17] 6.311744 8.904037 6.853158 9.699162 1.857385 2.500580
```

```
df[outs_mpg$extreme, "mpg"]
```

```
[1] 51.71888 51.10256 60.79850 55.06192 54.69687 59.37574 52.87664 52.07685  
[9] 54.03751 60.34226 54.44925 54.29924 53.68512 54.49085 50.50866 52.70174  
[17] 57.74811 56.76985 55.44259 55.47986 56.78424 55.18505 55.73419 55.05140  
[25] 61.91662 54.44355 56.01543 55.67018 55.08581 56.05532 53.34436 55.86386  
[33] 54.38946 58.29225 61.23032 51.08887 57.28788 57.63969 59.87276 58.42167  
[41] 56.60836 57.04605
```

3.2 Imputación de factores

Vamos a empezar echando un vistazo al summary de las variables categóricas.

Podemos apreciar que en el caso de fuelType, nos aparecen 13 valores NA (los missings que hemos encontrado).

En el caso de engineSize, podemos ver que aparecen 9 NAs (los errores que hemos encontrado).

```
summary(df[,vars_cat])
```

```
      model          year      transmission
VW- Golf       : 471    2019   :1563   f.Trans-Manual   :1787
Mercedes- C Class: 376    2017   : 895   f.Trans-SemiAuto :1896
VW- Polo        : 337    2016   : 885   f.Trans-Automatic:1317
BMW- 3 Series    : 274    2018   : 460
Mercedes- A Class: 260    2015   : 424
Mercedes- E Class: 201    2020   : 316
(Other)         :3081   (Other): 457

      fuelType      engineSize
f.Fuel-Diesel:2868     2       :2142
f.Fuel-Petrol:2060     1.5     : 554
f.Fuel-Hybrid: 59      3       : 512
NA's           : 13      2.1     : 412
                  1       : 365
                  (Other):1006
NA's           :      9
```

Procedemos con la imputación:

```
res.immca <- imputeMCA(df[,vars_cat],ncp=10)
summary(res.immca$completeObs)
```

```
      model          year      transmission
VW- Golf       : 471    2019   :1563   f.Trans-Manual   :1787
Mercedes- C Class: 376    2017   : 895   f.Trans-SemiAuto :1896
VW- Polo        : 337    2016   : 885   f.Trans-Automatic:1317
BMW- 3 Series    : 274    2018   : 460
Mercedes- A Class: 260    2015   : 424
Mercedes- E Class: 201    2020   : 316
(Other)         :3081   (Other): 457

      fuelType      engineSize
f.Fuel-Diesel:2875     2       :2149
f.Fuel-Petrol:2066     1.5     : 554
f.Fuel-Hybrid: 59      3       : 513
                  2.1     : 412
                  1       : 366
                  1.6     : 345
                  (Other): 661
```

Podemos ver como en el summary que acabamos de realizar, ya no aparecen valores NA para ninguna de las variables.

Sustituimos la salida de la imputación en nuestro dataframe:

```
df[,vars_cat] <- res.immca$completeObs
```

4 Discretización de variables numéricas en factores

Una vez hemos realizado la imputación, tanto de las variables numéricas como de los factores, vamos a proceder a la discretización de las variables numéricas.

En los casos de age o tax, esta discretización se ha hecho de manera pseudo-aleatoria, ya que la discretización por cuantiles no funcionaba bien debido a la distribución de estas variables.

```
vars_num
```

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

Como hemos podido ver con anterioridad, las distribuciones de estas variables son bastante diferentes entre sí.

En el caso de price o mpg, por ejemplo, podemos ver que la distribución se aproxima más a una normal.

Por otro lado, en la variable mileage, esta se podría aproximar mejor con una distribución de chi-cuadrado o exponencial.

Finalmente, para la variable tax, podemos ver que existe una gran acumulación en valores cercanos a 145.

A continuación lo estudiaremos con más profundidad.

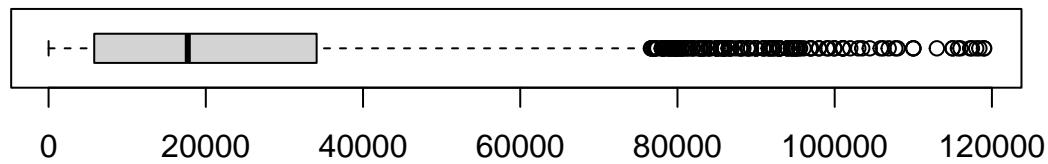
4.1 Mileage

En primer lugar vamos a echar un vistazo a esta variable.

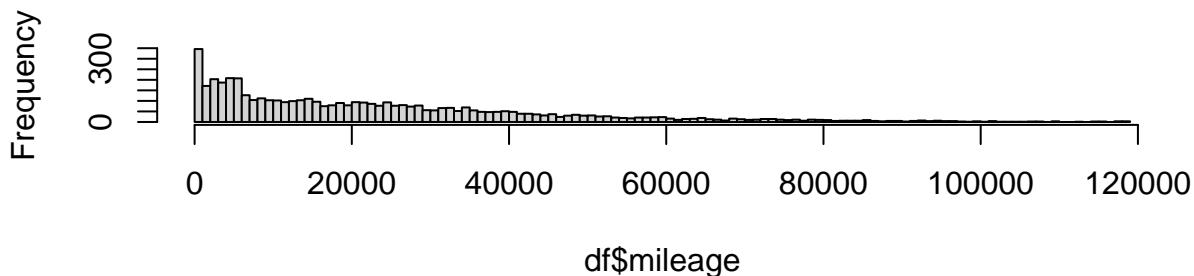
```
summary(df$mileage)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1	5807	17706	23318	34095	119000

```
par(mfrow = c(2,1))
boxplot(df$mileage, horizontal = TRUE)
hist(df$mileage, breaks=100)
```



Histogram of df\$mileage



Procedemos a la discretización según cuantiles, ya que en este caso si que era bastante representativa.

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

df$aux <- factor(cut(df$mileage, breaks=quants[1:5], include.lowest = TRUE))
df$f.miles<-factor(cut(df$mileage/1000,breaks=quants[1:5]/1000,include.lowest = TRUE ))
levels(df$f.miles)<-paste("f.miles-",levels(df$f.miles),sep="")
table(df$f.miles)
```

f.miles-[0.001,5.81]	f.miles-(5.81,17.7]	f.miles-(17.7,34.1]
1250	1251	1249
f.miles-(34.1,119]		
1250		

#Esto cuadra ya que estamos tomando como referencia los quantiles y podemos ver como en todos hay el mismo número de elementos.

Como podemos apreciar, aparecen 1250 elementos en cada cuantil, que es lo que esperaríamos en condiciones ideales.

4.2 Tax

Para la variables tax la cosa se complica un poco.

Si echamos un vistazo a algunos gráficos, podemos ver como hay una gran acumulación en valores entre 140 y 155. Esto hace que, al generar los cuantiles, el q1 y el q2 tengan el mismo valor, de modo que vamos a proceder con una discretización alternativa.

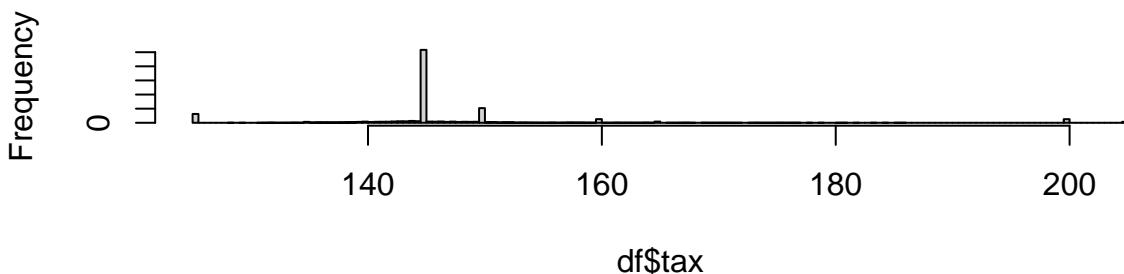
```
summary(df$tax)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
125.0	145.0	145.0	146.9	147.5	205.0

```
par(mfrow = c(2, 1))
boxplot(df$tax, horizontal = TRUE)
hist(df$tax, breaks=200)
```



Histogram of df\$tax



```
df$aux <- factor(cut(df$tax, breaks=c(0,144,145,155,205), include.lowest = TRUE))
table(df$aux)
```

[0,144]	(144,145]	(145,155]	(155,205]
918	2640	974	468

En la tabla podemos ver el gran pico en el valor 145, que acumula 2647 elementos, más de la mitad del tamaño de la muestra.

Por último, generamos una variable factor con etiquetas para los diferentes intervalos que hemos definido.

```
df$f.tax<-factor(cut(df$tax,breaks=c(0,144,145,155,205),include.lowest = TRUE ))
levels(df$f.tax)<-paste("f.tax-",levels(df$f.tax),sep="")
table(df$f.tax)
```

f.tax-[0,144]	f.tax-(144,145]	f.tax-(145,155]	f.tax-(155,205]
918	2640	974	468

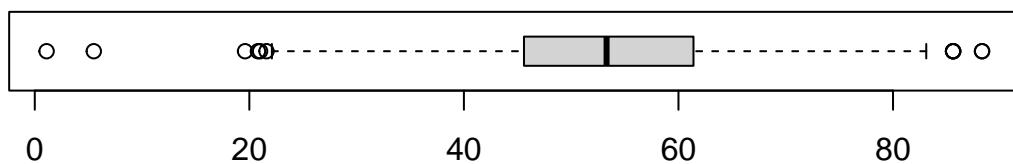
4.3 Mpg

En el caso de la variable mpg, vamos a proceder con una discretización por cuantiles, ya que su distribución lo permite y el resultado es bastante explicativo junto con las etiquetas que vamos a añadir al factor.

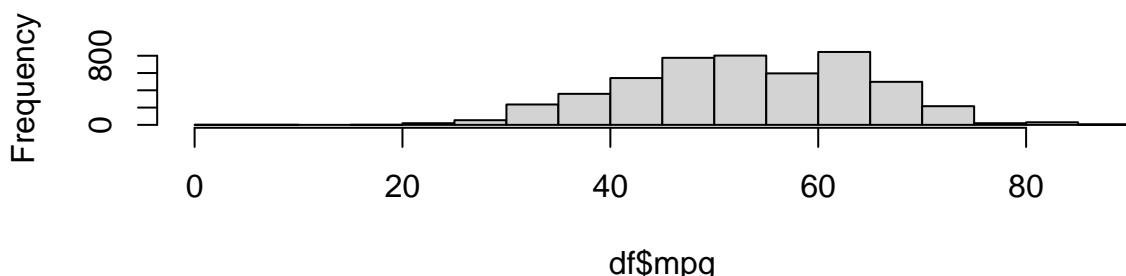
```
summary(df$mpg)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.1	45.6	53.3	53.1	61.4	88.3

```
par(mfrow = c(2, 1))
boxplot(df$mpg, horizontal = TRUE)
hist(df$mpg)
```



Histogram of df\$mpg



```
quants <- quantile(df$mpg, seq(0,1,0.25), na.rm=TRUE)
df$aux <- factor(cut(df$mpg, breaks=quants[1:5], include.lowest = TRUE))
df$f.mpg<-factor(cut(df$mpg,breaks=quants[1:5],include.lowest = TRUE ))
levels(df$f.mpg)<-paste("f.mpg-",c("muy bajo","bajo","medio","alto"),sep="")
table(df$f.mpg)
```

f.mpg-muy bajo	f.mpg-bajo	f.mpg-medio	f.mpg-alto
1320	1327	1194	1159

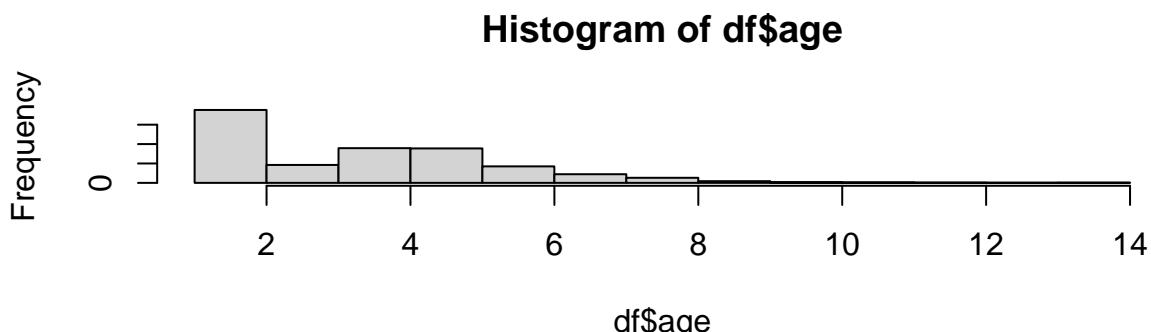
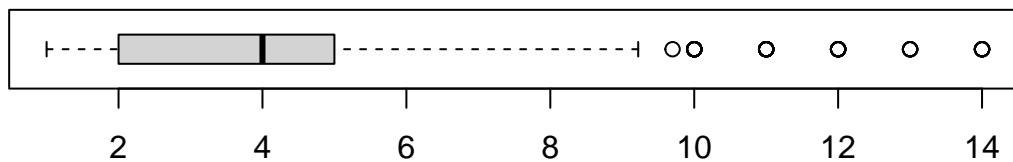
4.3.1 Age

La variable age, similarmente a la variable mileage, tiene una distribución que se acumula en valores bajos. De modo que vamos a aplicar una discretización seg\xfcren los primeros cuantiles generando tres intervalos.

```
summary(df$age)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.000	2.000	4.000	3.798	5.000	14.000

```
par(mfrow = c(2, 1))
boxplot(df$age, horizontal = TRUE)
hist(df$age)
```



```
quants <- quantile(df$age, seq(0, 1, 0.25), na.rm=TRUE)
```

```
df$aux <- factor(cut(df$age, breaks=c(quants[1:3], max(df$age)), include.lowest = TRUE))
summary(df$aux)
```

```
[1,2]  (2,4]  (4,14]
1880    1357    1763
```

```
df$f.age<-factor(cut(df$age, breaks=c(quants[1:3], max(df$age)), include.lowest = TRUE ))
levels(df$f.age)<-paste("f.age-",c(levels(df$f.age)[1:2], "(+4)", sep=""))
table(df$f.age)
```

```
f.age-[1,2] f.age-(2,4] f.age-(+4)
1880        1357        1763
```

4.3.2 Price

Por último, vamos a discretizar nuestro target numérico price. En este caso, lo vamos a dividir en 8 categorías ya que es uno de los requisitos que se marcan para entregas posteriores.

```
quants <- quantile(df$price, seq(0,1,0.125), na.rm=TRUE)

df$aux <- factor(cut(df$price, breaks=c(quants[1:8], max(df$price)), include.lowest = TRUE))
summary(df$aux)
```

```
[899,1.1e+04]  (1.1e+04,1.4e+04]  (1.4e+04,1.7e+04]  (1.7e+04,1.95e+04]
630              641              604              625
(1.95e+04,2.2e+04]  (2.2e+04,2.6e+04]  (2.6e+04,3.15e+04] (3.15e+04,1.09e+05]
632              619              624              625
```

```
df$f.price<-factor(cut(df$price,breaks=c(quants[1:8],max(df$price)),include.lowest = TRUE ))
levels(df$f.price)<-paste("f.price-",c(levels(df$f.price)),sep="")
table(df$f.price)
```

```
f.price-[899,1.1e+04]  f.price-(1.1e+04,1.4e+04]
630                      641
f.price-(1.4e+04,1.7e+04]  f.price-(1.7e+04,1.95e+04]
604                      625
f.price-(1.95e+04,2.2e+04]  f.price-(2.2e+04,2.6e+04]
632                      619
f.price-(2.6e+04,3.15e+04] f.price-(3.15e+04,1.09e+05]
624                      625
```

4.4 Generación del target categórico Audi

Vamos a proceder a generar nuestro target categórico.

Como indica la documentación de la práctica, este va a consistir en una variable que nos indique si el fabricante de un vehículo es Audi.

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

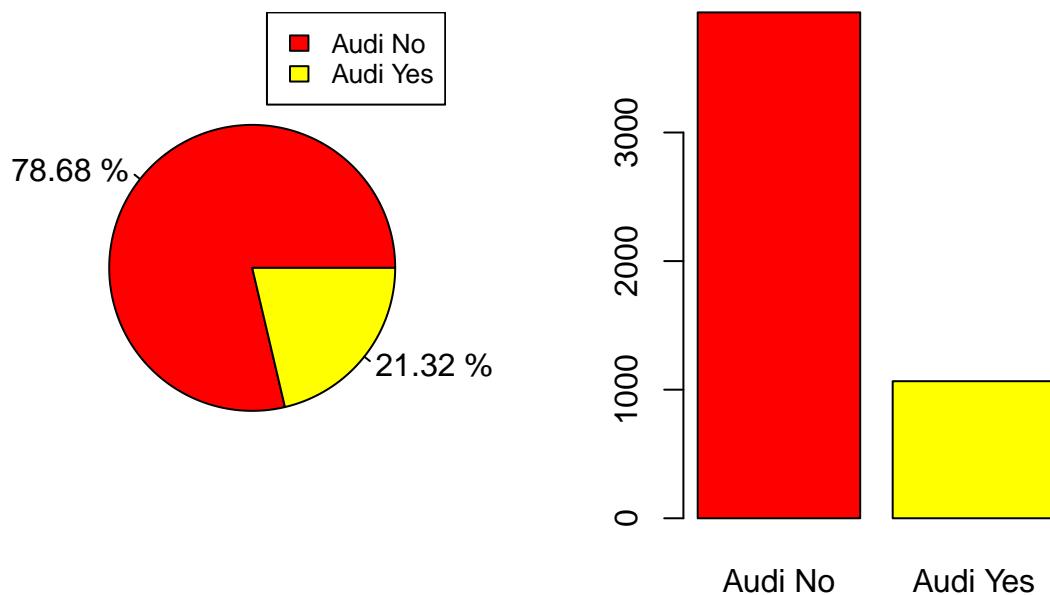
Audi	No	Audi	Yes
3934	1066		

A continuación algunos plots que nos ayudan a explicar esta variable.

```
par(mfrow = c(1, 2))
# Pie
piepercent<-round(100*(table(df$Audi)/nrow(df)),dig=2); piepercent
```

Audi	No	Audi	Yes
78.68	21.32		

```
pie(table(df$Audi),col=heat.colors(2),labels=paste(piepercent,"%"))
legend("topright", levels(df$Audi), cex = 0.8, fill = heat.colors(2))
# Bar Chart
barplot(table(df$Audi),col=c("red","yellow"))
```



5 Identificación los outliers multivariantes

En este apartado vamos a proceder con la identificación de los outliers multivariante. En primer lugar, vamos a cargar la librería y echaremos un primer vistazo a los posibles outliers.

```
library(mvoutlier)
```

```
Loading required package: sgeostat
```

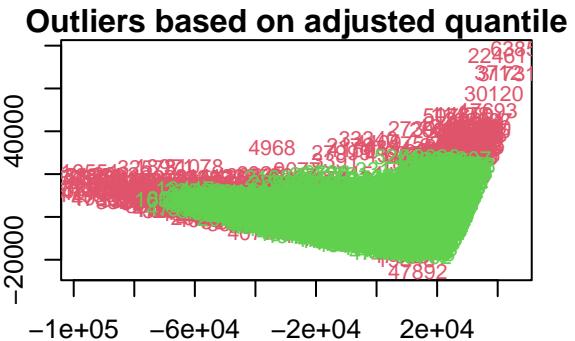
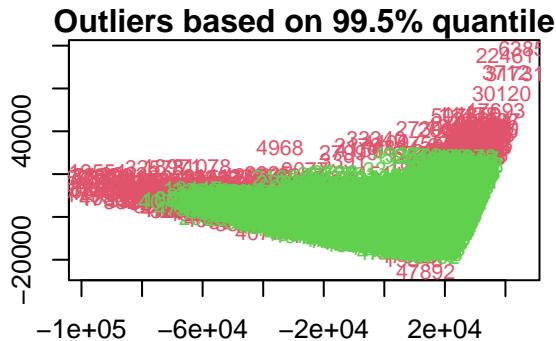
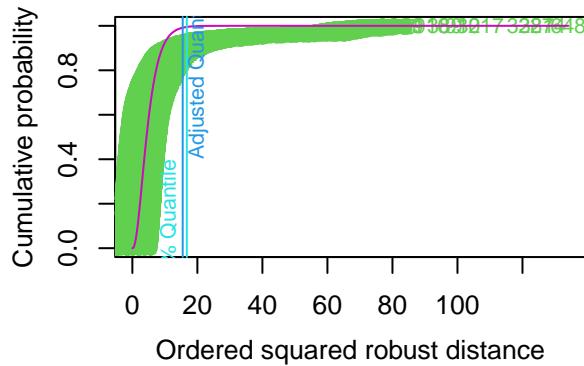
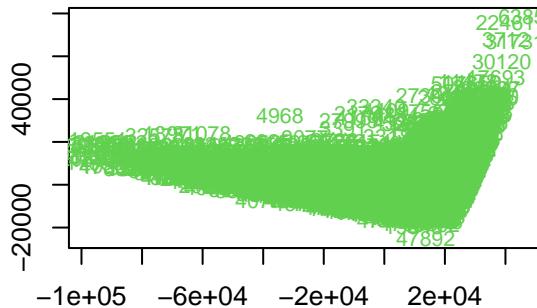
```
ll<-which(is.na(df$price)) #vacío  
summary(df[,c(vars_res[1],vars_num)])
```

```
   price      mileage       tax      mpg  
Min. : 899  Min. : 1  Min. :125.0  Min. : 1.1  
1st Qu.: 13995  1st Qu.: 5807  1st Qu.:145.0  1st Qu.:45.6  
Median : 19498  Median : 17706  Median :145.0  Median :53.3  
Mean   : 21207  Mean   : 23318  Mean   :146.9  Mean   :53.1  
3rd Qu.: 25980  3rd Qu.: 34095  3rd Qu.:147.5  3rd Qu.:61.4  
Max.   :109495  Max.   :119000  Max.   :205.0  Max.   :88.3  
  
   age  
Min. : 1.000  
1st Qu.: 2.000  
Median : 4.000  
Mean   : 3.798  
3rd Qu.: 5.000  
Max.   :14.000
```

La ejecución de la función aq.plot (Adjusted Quantile) genera 4 gráficos. * En el de arriba a la izquierda podemos ver los datos originales. * En el de arriba a la derecha podemos ver la aproximación de estos datos a una distribucion de chi-cuadrado. * En el de abajo a la izquierda podemos ver los outliers determinados por el cuantil especificado de la chi-cuadrado (99.5%). * En el de abajo a la derecha podemos ver los outliers determinados por el Adjusted Quantile (99.5%).

```
mout<-aq.plot(df[,c(vars_res[1],vars_num)],delta=qchisq(0.995,5),quan=0.995)
```

```
Projection to the first and second robust principal components.  
Proportion of total variation (explained variance): 0.9993919
```



A continuación, vamos a usar la función `Moutlier` para mostrar más información sobre los posibles outliers segun la distancia de Mahalanobis. Cabe destacar que el output de esta función nos devuelve tanto la distancia de Mahalanobis clásica como la robusta.

Esta función se debe aplicar a conjuntos de datos que siguen una distribución normal, de modo que, según el estudio que hemos realizado previamente, solo lo aplicaremos a las variables de price, mileage, mpg y distance.

Disclaimer: Para las variables que menciono anteriormente, el programa funciona bien. Si añadimos tax, falla. Pero si nos fijamos en los histogramas/barplots de las variables que mencionamos, las únicas en las que podríamos suponer una normalidad en los datos serían price y mpg, ya que age o mileage tienden a una acumulación en valores bajos siguiendo distribuciones que podrían ser parecidas a chi-cuadrado, log normal o exponencial.

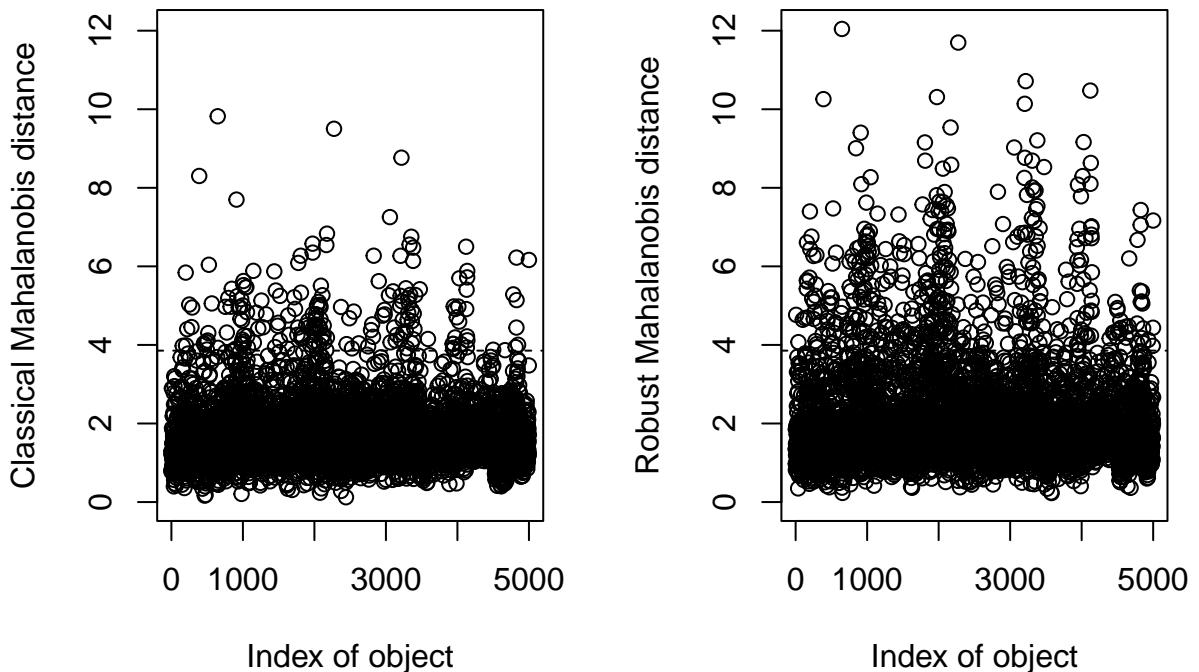
Seguimos con el análisis de Outliers Multivariantes.

En este caso, aplicaremos la función `Moutlier` a las variables que hemos mencionado anteriormente, mostrando las distancias de Mahalanobis clásica y robusta. Además, esta función nos genera un cutoff a partir del cual podemos detectar los outliers.

```
library(chemometrics)
```

Loading required package: rpart

```
mout<-Moutlier(df[,c("price","mileage","mpg","age")],quantile = 0.995, plot = TRUE)
```



```
11<-which(mout$md > mout$cutoff)
```

Vamos a echar un vistazo a las propiedades de los individuos considerados como Multivariant Outliers.

```
summary(df[11,c("price","mileage","mpg","age")])
```

	price	mileage	mpg	age
Min.	: 899	Min. : 1	Min. : 1.10	Min. : 1.000
1st Qu.	: 6920	1st Qu.: 6267	1st Qu.:30.10	1st Qu.: 2.000
Median	: 12972	Median : 74426	Median :42.80	Median : 5.000
Mean	: 28971	Mean : 57231	Mean :44.69	Mean : 5.972
3rd Qu.	: 59942	3rd Qu.: 92922	3rd Qu.:57.05	3rd Qu.: 8.583
Max.	:109495	Max. :119000	Max. :85.60	Max. :14.000

Finalmente, crearemos una variable auxiliar que nos marcará, para cada individuo, si es Multivariant Outlier o no.

```
df$mout <- 0
df$mout[ 11 ]<-1
df$mout <- factor( df$mout, labels=c( "NoMOut","YesMOut"))
table(df$mout)
```

NoMOut	YesMOut
4810	190

6 Profiling

Por último, vamos a realizar el profiling de nuestro dataframe según nuestras variables target: Audi como variable categórica y price como variable numérica.

6.1 Target numérico (Price)

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

Min. 1st Qu. Median Mean 3rd Qu. Max.
899 13995 19498 21207 25980 109495

vars_num <- c("mileage", "tax", "mpg", "age")
vars_cat <- c("model", "year", "transmission", "fuelType", "engineSize", "f.miles", "f.tax", "f.mpg", "f.age", "Audi")
```

Vamos a proceder al profiling del target numérico a partir de la función condes del paquete FactoMineR

```
res.condes<-condes(df[,c("price", vars_num, vars_cat, "manufacturer")], 1)

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

	correlation	p.value
mileage	-0.5291971	0
mpg	-0.5799013	0
age	-0.5814226	0

Como podemos apreciar en la anterior salida, las variables numéricas que más correlación tienen con nuestra variable target (price) son mileage, mpg y age. En este caso, cabe destacar que se correlacionan de manera negativa. Esto indica que son inversamente proporcionales, es decir, que a más mileage/mpg/age, menor price (y viceversa).

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

	R2	p.value
model	0.498113659	0.000000e+00
year	0.368876026	0.000000e+00
engineSize	0.417317006	0.000000e+00
f.miles	0.302896034	0.000000e+00
f.mpg	0.286791156	0.000000e+00
f.age	0.331018734	0.000000e+00
transmission	0.222053178	3.454238e-273
f.tax	0.155741942	4.820306e-183
manufacturer	0.085248015	3.575883e-96
fuelType	0.005311573	1.663919e-06
Audi	0.004578046	1.678942e-06

En el caso de las variables categóricas (o factores), podemos apreciar que existe una clara relación entre los factores model, year, engineSize. También con algunos de los factores que hemos creado derivados de las variables numéricas. Vamos a analizarlo más a fondo:

Ahora nos fijaremos en las primeras líneas de la salida de res.condes\$category:

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

	Estimate	p.value
f.age=f.age-[1,2]	7516.899	0.000000e+00
f.mpg=f.mpg-muy bajo	9207.286	0.000000e+00
f.miles=f.miles-[0.001,5.81]	7983.258	6.380064e-232
year=2019	16354.381	6.510724e-224
f.tax=f.tax-(144,145]	5858.741	9.173133e-174
engineSize=3	8313.245	2.099854e-173

En esta salida podemos apreciar como para coches nuevos, y con un consumo o kilometraje bajos, el precio estimado es más alto. Como casos notables también podemos mencionar algunos casos de coches premium, para los que el valor estimado es más alto:

```
model=Mercedes- G Class      45588.6851  5.645865e-07
model=BMW- X7                42925.9709  9.727955e-37
```

Y si nos fijamos en las últimas:

```
tail(res.condes$category)
```

	Estimate	p.value
f.tax=f.tax-[0,144]	-4062.936	4.491610e-87
manufacturer=VW	-4880.202	2.247380e-94
f.mpg=f.mpg-alto	-5547.111	4.457754e-106
f.miles=f.miles-(34.1,119]	-7304.049	3.927777e-191
transmission=f.Trans-Manual	-6839.754	4.814587e-273
f.age=f.age-(+4)	-6623.392	2.639305e-287

Podemos apreciar como factores como el cambio de marchas manual, un kilometraje alto o un consumo alto tienden a abaratar el vehículo.

Algunos casos especiales son modelos de VW, que ven su precio realmente reducido:

```
model=VW- CC      -17098.5649  7.630716e-03
model=VW- Beetle -20721.5371  1.122196e-04
```

6.2 Target factor (AUDI)

Vamos a proceder a ejecutar la función catdes con para identificar las asociaciones hacia el target categórico que hemos generado. Para esto, y ya que simplemente considero que no es indicativo, no usaré las variable modelo ni manufacturer, ya que no aportan ningún tipo de infomación al análisis.

```
res.catdes<-catdes(df[,c("Audi",vars_num,vars_cat[2:9])],1)
```

Procedemos a ver las variables que parecen estar más correlacionadas con nuestra variable target:

```
res.catdes$quanti.var # Global association to numeric variables
```

	Eta2	P-value
mpg	0.014005511	4.632358e-17
tax	0.003617358	2.084464e-05
age	0.001202533	1.419867e-02
mileage	0.001099604	1.903503e-02

Si lo analizamos un poco más las relaciones con variables numéricas, podemos ver que los vehículos Audi tienen consumos más bajos y kilometrages, antigüedad e impuestos más altos que la muestra que estudiamos. Si analizamos los vehículos que no son Audi, veremos lo contrario.

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

\$'Audi No'	v.test	Mean in category	Overall mean	sd in category	Overall sd
mpg	8.367410	53.785122	53.097238	11.081567	11.166173
mileage	-2.344551	22943.522121	23318.245205	21593.705113	21708.552196
age	-2.451828	3.762023	3.798202	1.984473	2.004245
tax	-4.252431	146.478947	146.865790	11.688011	12.355981
	p.value				
mpg	5.889854e-17				
mileage	1.904999e-02				

```
age      1.421326e-02
tax     2.114622e-05
```

```
$'Audi Yes'
      v.test Mean in category Overall mean sd in category Overall sd
tax      4.252431    148.29341   146.865790    14.468957   12.355981
age      2.451828     3.93172    3.798202    2.070118    2.004245
mileage  2.344551  24701.13508  23318.245205  22072.237729 21708.552196
mpg     -8.367410    50.55865    53.097238    11.110319   11.166173
      p.value
tax     2.114622e-05
age     1.421326e-02
mileage 1.904999e-02
mpg     5.889854e-17
```

Si estudiamos las variables categóricas, podemos ver que los factores generados con las discretizaciones que hemos realizado anteriormente se relacionan de manera estrecha con nuestro target, además del engineSize, el fuelType o la transmission.

```
res.catdes$test.chi2 # Global association to factors
```

	p.value	df
engineSize	3.613671e-90	24
f.mpg	1.584246e-16	3
fuelType	1.129965e-07	2
transmission	1.413295e-04	2
f.tax	6.712343e-04	3
f.miles	1.622641e-03	3
f.age	3.813175e-02	2

Por último, vamos a entrar más en detalle en la relación de las variables categóricas con nuestro target.

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

```
$'Audi No'
      Cla/Mod      Mod/Cla Global      p.value
engineSize=2.1      100.00000 10.47280122   8.24 9.327968e-46
engineSize=1.2      100.00000 3.73665480   2.94 2.714851e-16
f.mpg=f.mpg-alto    84.38309 24.86019319  23.18 2.960521e-08
engineSize=1.3      100.00000 1.60142349   1.26 2.471998e-07
engineSize=1.5      85.74007 12.07422471  11.08 7.887020e-06
fuelType=f.Fuel-Diesel 80.83478 59.07473310  57.50 1.626774e-05
f.mpg=f.mpg-medio    82.99832 25.19064565  23.88 2.161084e-05
f.miles=f.miles-(5.81,17.7] 82.25420 26.15658363  25.02 3.048507e-04
transmission=f.Trans-SemiAuto 81.32911 39.19674631  37.92 3.253466e-04
fuelType=f.Fuel-Hybrid    94.91525 1.42348754   1.18 6.320005e-04
engineSize=2.2      100.00000 0.50838841   0.40 8.180915e-03
year=2019          80.48624 31.97763091  31.26 3.475673e-02
engineSize=1       82.78689 7.70208439   7.32 4.298931e-02
f.mpg=f.mpg-bajo    76.71439 25.87697001  26.54 4.257922e-02
f.miles=f.miles-(34.1,119] 76.48000 24.30096594  25.00 2.943481e-02
f.age=f.age-(+4)    76.68746 34.36705643  35.26 1.150630e-02
engineSize=2.9      40.00000 0.10167768   0.20 1.006985e-02
engineSize=2.5      28.57143 0.05083884   0.14 6.759033e-03
engineSize=4.2      0.00000 0.00000000   0.08 2.056946e-03
engineSize=2       76.22150 41.63701068  42.98 2.392416e-04
engineSize=4       50.00000 0.45754957   0.72 1.564847e-04
f.tax=f.tax-(155,205] 71.58120 8.51550585   9.36 1.343933e-04
transmission=f.Trans-Manual 75.65753 34.36705643  35.74 1.111890e-04
engineSize=1.8      47.36842 0.45754957   0.76 2.457585e-05
fuelType=f.Fuel-Petrol 75.21781 39.50177936  41.32 6.003236e-07
f.mpg=f.mpg-muy bajo 71.74242 24.07219115  26.40 2.195198e-12
```

engineSize=1.4	45.16129	3.55871886	6.20	6.336915e-41
v.test				
engineSize=2.1	14.198736			
engineSize=1.2	8.185363			
f.mpg=f.mpg-alto	5.543756			
engineSize=1.3	5.159811			
engineSize=1.5	4.468228			
fuelType=f.Fuel-Diesel	4.310783			
f.mpg=f.mpg-medio	4.247563			
f.miles=f.miles-(5.81,17.7]	3.611143			
transmission=f.Trans-SemiAuto	3.594235			
fuelType=f.Fuel-Hybrid	3.417496			
engineSize=2.2	2.644511			
year=2019	2.111181			
engineSize=1	2.023814			
f.mpg=f.mpg-bajo	-2.027814			
f.miles=f.miles-(34.1,119]	-2.177614			
f.age=f.age-(+4)	-2.526934			
engineSize=2.9	-2.573421			
engineSize=2.5	-2.708489			
engineSize=4.2	-3.081884			
engineSize=2	-3.673510			
engineSize=4	-3.780546			
f.tax=f.tax-(155,205]	-3.818265			
transmission=f.Trans-Manual	-3.864781			
engineSize=1.8	-4.218660			
fuelType=f.Fuel-Petrol	-4.991113			
f.mpg=f.mpg-muy bajo	-7.021487			
engineSize=1.4	-13.396516			

\$‘Audi Yes’

	Cla/Mod	Mod/Cla	Global	p.value
engineSize=1.4	54.838710	15.9474672	6.20	6.336915e-41
f.mpg=f.mpg-muy bajo	28.257576	34.9906191	26.40	2.195198e-12
fuelType=f.Fuel-Petrol	24.782188	48.0300188	41.32	6.003236e-07
engineSize=1.8	52.631579	1.8761726	0.76	2.457585e-05
transmission=f.Trans-Manual	24.342473	40.8067542	35.74	1.111890e-04
f.tax=f.tax-(155,205]	28.418803	12.4765478	9.36	1.343933e-04
engineSize=4	50.000000	1.6885553	0.72	1.564847e-04
engineSize=2	23.778502	47.9362101	42.98	2.392416e-04
engineSize=4.2	100.000000	0.3752345	0.08	2.056946e-03
engineSize=2.5	71.428571	0.4690432	0.14	6.759033e-03
engineSize=2.9	60.000000	0.5628518	0.20	1.006985e-02
f.age=f.age-(+4)	23.312535	38.5553471	35.26	1.150630e-02
f.miles=f.miles-(34.1,119]	23.520000	27.5797373	25.00	2.943481e-02
f.mpg=f.mpg-bajo	23.285607	28.9868668	26.54	4.257922e-02
engineSize=1	17.213115	5.9099437	7.32	4.298931e-02
year=2019	19.513756	28.6116323	31.26	3.475673e-02
engineSize=2.2	0.000000	0.0000000	0.40	8.180915e-03
fuelType=f.Fuel-Hybrid	5.084746	0.2814259	1.18	6.320005e-04
transmission=f.Trans-SemiAuto	18.670886	33.2082552	37.92	3.253466e-04
f.miles=f.miles-(5.81,17.7]	17.745803	20.8255159	25.02	3.048507e-04
f.mpg=f.mpg-medio	17.001675	19.0431520	23.88	2.161084e-05
fuelType=f.Fuel-Diesel	19.165217	51.6885553	57.50	1.626774e-05
engineSize=1.5	14.259928	7.4108818	11.08	7.887020e-06
engineSize=1.3	0.000000	0.0000000	1.26	2.471998e-07
f.mpg=f.mpg-alto	15.616911	16.9793621	23.18	2.960521e-08
engineSize=1.2	0.000000	0.0000000	2.94	2.714851e-16
engineSize=2.1	0.000000	0.0000000	8.24	9.327968e-46
v.test				
engineSize=1.4	13.396516			
f.mpg=f.mpg-muy bajo	7.021487			
fuelType=f.Fuel-Petrol	4.991113			
engineSize=1.8	4.218660			

transmission=f.Trans-Manual	3.864781
f.tax=f.tax-(155,205]	3.818265
engineSize=4	3.780546
engineSize=2	3.673510
engineSize=4.2	3.081884
engineSize=2.5	2.708489
engineSize=2.9	2.573421
f.age=f.age-(+4)	2.526934
f.miles=f.miles-(34.1,119]	2.177614
f.mpg=f.mpg-bajo	2.027814
engineSize=1	-2.023814
year=2019	-2.111181
engineSize=2.2	-2.644511
fuelType=f.Fuel-Hybrid	-3.417496
transmission=f.Trans-SemiAuto	-3.594235
f.miles=f.miles-(5.81,17.7]	-3.611143
f.mpg=f.mpg-medio	-4.247563
fuelType=f.Fuel-Diesel	-4.310783
engineSize=1.5	-4.468228
engineSize=1.3	-5.159811
f.mpg=f.mpg-alto	-5.543756
engineSize=1.2	-8.185363
engineSize=2.1	-14.198736

Vamos a interpretar un poco la salida anterior. Si cogemos este ejemplo,

```
'Audi No'
          Cla/Mod      Mod/Cla Global      p.value      v.test
engineSize=2.1       100.00000 10.47280122    8.24 9.327968e-46  14.198736
```

podemos entender como el 100% de los vehículos con engineSize=2.1, no están fabricados por Audi. También podemos extraer que el 10.47% de los vehículos no fabricados por Audi tienen engineSize=2.1.

Otro ejemplo:

```
'Audi Yes'
          Cla/Mod      Mod/Cla Global      p.value      v.test
transmission=f.Trans-Manual  24.342473 40.8067542  35.74 1.111890e-04  3.864781
```

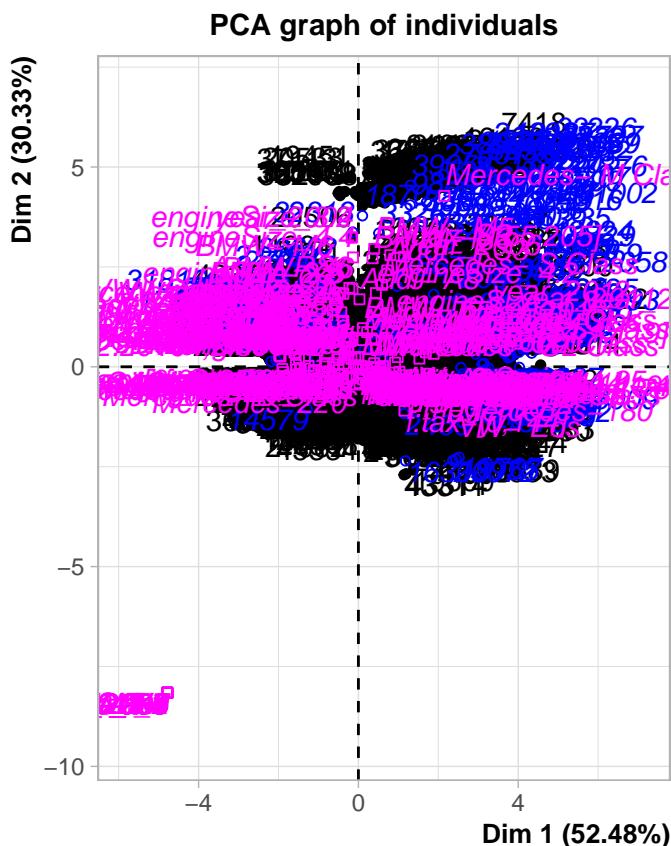
En este caso podemos ver como el 24.34% de los vehículos con transmisión manual son Audi. Además, el 40.80% de los vehículos fabricados por Audi, tienen transmisión manual.

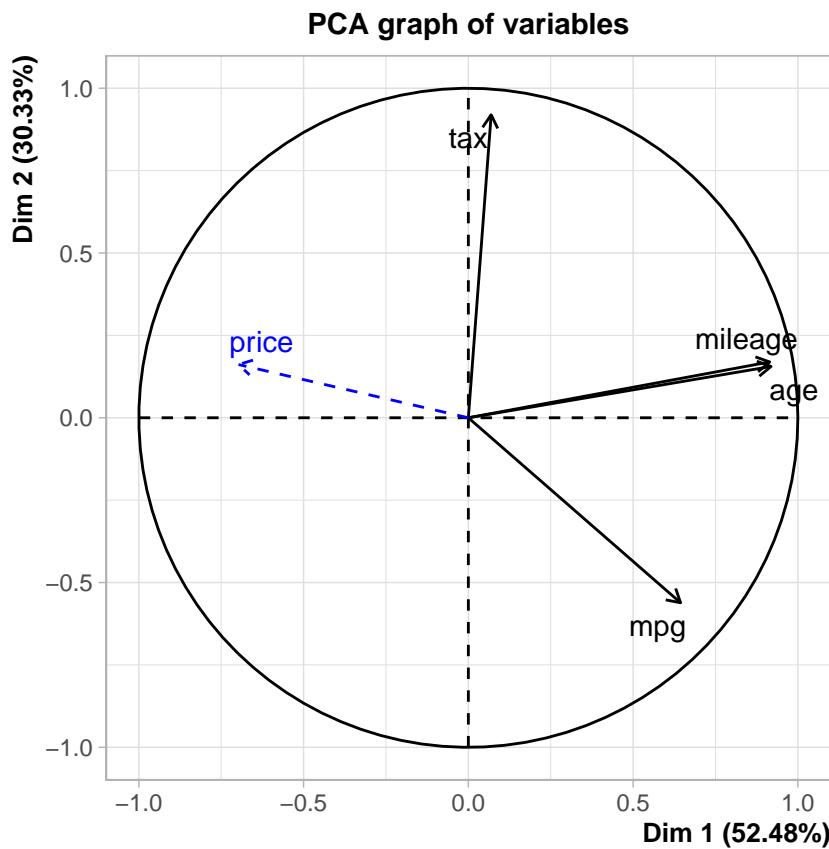
7 Análisis de Componentes Principales

En primer lugar, vamos a echar un vistazo a nuestras variables. Las vamos a mostrar en un orden concreto que nos va ayudar más tarde a referirnos a ellas a partir de sus índices.

Realizamos el análisis PCA de nuestro dataset, pasando como variables supplementarias cualitativas todos nuestros factores, y como variable supplementaria cuantitativa nuestro target numérico price. Además, añadimos los individuos que hemos categorizado como multivariant outliers como individuos supplementarios, para que no introduzcan ruido a la hora de calcular el PCA.

```
ll <- which( df$mout == "YesMOut")
res.pca<-PCA(df[,c(vars_res, vars_cat, vars_num)],quali.sup=c(2:13),quanti.sup= c(1), ind.sup = ll )
```





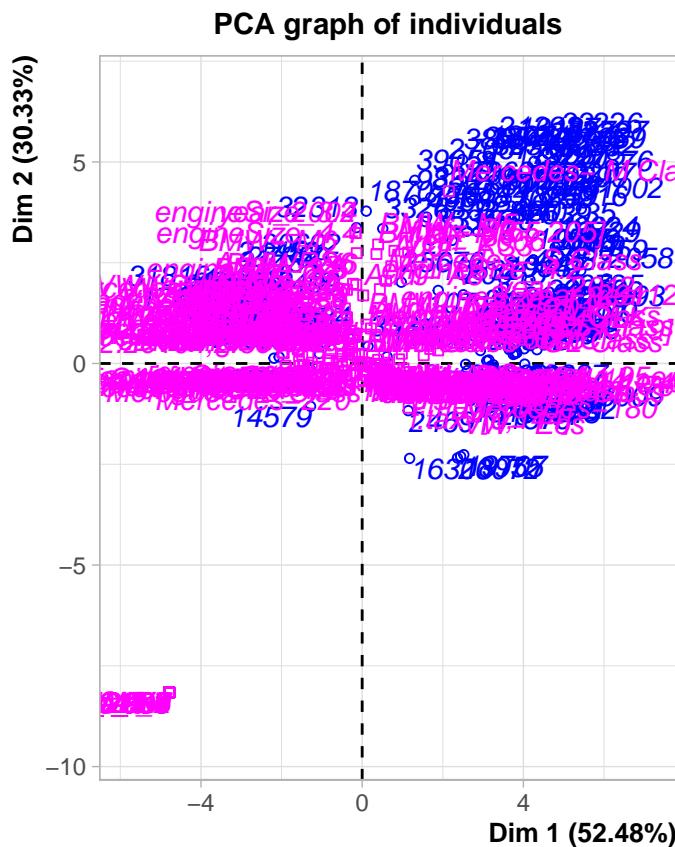
Como podemos apreciar en los gráficos previos, las dos primeras componentes principales aglutinan un 52,64% y un 30,09% de la variabilidad respectivamente.

Podemos apreciar también como la primera componente principal parece aglutinar la variabilidad de las variables mileage y age, que se proyectan en la misma dirección, solapándose en el diagrama, mientras que la segunda componente principal aglutina sobre todo la variabilidad de la variable tax. Así mismo, parece que las dos componentes reflejan la variable mpg por igual, sin embargo en el caso de la Dim2 esta relación es negativa.

Además de esto, podemos ver la proyección de nuestro target numérico price en el plano formado por las dos primeras componentes principales. Se puede apreciar que price está más relacionada con la primera componente que con la segunda, pero de manera negativa.

Por último, podemos ver la proyección de los individuos en el plano formado por las dos componentes principales. Cabe destacar un pequeño grupo de individuos en la esquina inferior izquierda del plot.

```
plot.PCA(res.pca, choix=c("ind"), invisible=c("ind"))
```



```
res.pca$eig
```

	eigenvalue	percentage of variance	cumulative percentage of variance
comp 1	2.0993588	52.483970	52.48397
comp 2	1.2133943	30.334858	82.81883
comp 3	0.4978429	12.446073	95.26490
comp 4	0.1894040	4.735099	100.00000

Se puede ver que se han creado 4 componentes diferentes, cuyos Eigenvalues normalizados son 2,106, 1,203, 0,510 y 0,181 respectivamente. Si seguimos el criterio de Kaiser, como estos eigenvalues ya se han normalizado, deberíamos quedarnos con aquellos componentes con eigenvalues superiores a 1, de modo que nos quedaríamos con las dos primeras componentes.

En la siguiente salida, podemos ver que, como ya habíamos mencionado previamente, la variable tax tiene más correlación con el segundo componente que en el primero, mientras que podemos ver que pasa lo contrario con las variables mileage o age. También podemos ver que la variable price se relaciona más con la primera componente aunque lo hace de manera negativa.

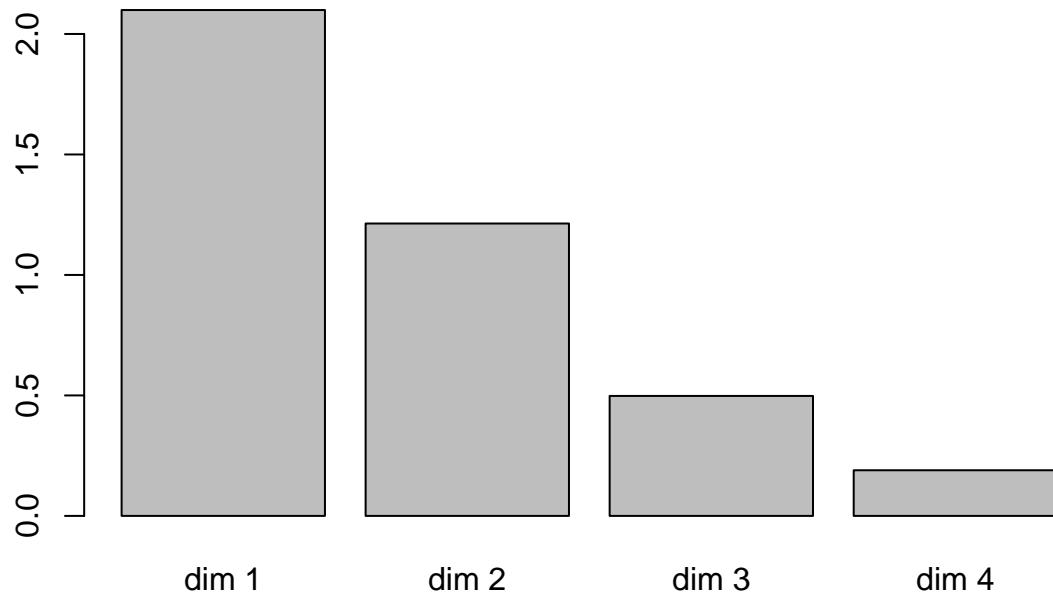
```
res.pca$var$cor
```

	Dim.1	Dim.2	Dim.3	Dim.4
mileage	0.9143891	0.1694102	-0.2036793	3.061168e-01
tax	0.0692207	0.9194200	0.3871374	-7.978036e-05
mpg	0.6440027	-0.5613679	0.5196964	6.494828e-03
age	0.9185425	0.1556517	-0.1907824	-3.092802e-01

En el siguiente plot podemos ver los eigenvalues de las cuatro componentes que se han generado.

```
barplot(res.pca$eig[,1], main="Eigenvalues", names.arg=paste("dim", 1:nrow(res.pca$eig)))
```

Eigenvalues



7.1 Análisis según los individuos

A continuación, vamos a proceder a analizar los individuos.

En la salida a continuación, podemos apreciar: En las dos primeras columnas, las coordenadas que reciben los individuos para las dos primeras componentes principales. En las siguientes dos columnas, los valores correspondientes al cos² para las dos dimensiones correspondientes a cada individuo. En las dos últimas columnas, la contribución que tienen los individuos en cada componente principal.

```
head(round(cbind(res.pca$ind$coord[,1:2],res.pca$ind$cos2[,1:2],res.pca$ind$contrib[,1:2]),2))
```

	Dim.1	Dim.2	Dim.1	Dim.2	Dim.1	Dim.2
1	-0.01	0.12	0.00	0.06	0.00	0.00
6	1.00	-0.10	0.98	0.01	0.01	0.00
9	1.86	0.21	0.98	0.01	0.03	0.00
23	0.31	-0.35	0.32	0.40	0.00	0.00
25	0.45	4.15	0.01	0.82	0.00	0.30
38	0.97	-1.57	0.22	0.56	0.01	0.04

En la siguiente salida, podemos apreciar, de manera ordenada para la primera dimensión, los individuos que más contribución tienen hacia la primera dimensión y así como sus contribuciones para el resto de dimensiones.

```
inds <- res.pca$ind$coord  
inds <- as.data.frame(inds)  
rang.dim1<-inds[order(inds$Dim.1, decreasing = TRUE),]  
head(rang.dim1)
```

	Dim.1	Dim.2	Dim.3	Dim.4
9622	5.626670	1.19250963	-0.61229299	-0.04132951
10514	4.561279	0.04518798	-0.09835203	0.86137245
19638	4.560640	1.31633699	-0.66744109	0.57115462
10662	4.522936	0.96035244	-2.08628288	0.24765606
7901	4.515517	0.61590850	-0.33152451	-0.49510460
8052	4.394697	1.22998080	-0.62336570	0.38918147

Vamos a proceder a hacer lo mismo para la segunda dimensión.

```
rang.dim2<-inds[order(inds$Dim.2, decreasing = TRUE),]
head(rang.dim2)
```

	Dim.1	Dim.2	Dim.3	Dim.4
7418	3.580894	5.586232	0.1140913	0.24880380
19126	2.610944	5.200150	0.7854647	-0.94867095
31585	2.727762	5.167990	0.8247494	0.65204991
45549	3.004591	5.061358	0.9626625	0.06639833
31637	1.200250	5.027415	1.1713438	-0.11149442
34460	1.339198	5.026728	1.1603328	0.01650992

Como se puede apreciar en las salidas anteriores, podemos ver como parece haber elementos con más contribución en la segunda dimensión que en la primera.

A continuación, podemos ver todas las variables de los 10 primeros individuos que más aportan a la primera componente principal:

```
df[which(row.names(df) %in% row.names(res.pca$ind$coord[row.names(rang.dim1)[1:10],])),]
```

	model	year	price	transmission	mileage	fuelType	
7901	Audi- A3	2011	6500	f.Trans-Manual	74000	f.Fuel-Diesel	
8052	Audi- A4	2012	7990	f.Trans-Manual	88000	f.Fuel-Diesel	
9622	Audi- A3	2010	4295	f.Trans-Manual	97000	f.Fuel-Diesel	
9730	Audi- A5	2011	8395	f.Trans-Manual	85000	f.Fuel-Diesel	
10514	Audi- A1	2013	6999	f.Trans-Manual	90000	f.Fuel-Diesel	
10662	Audi- A4	2011	6995	f.Trans-Manual	95000	f.Fuel-Diesel	
19638	BMW- 1 Series	2012	6500	f.Trans-Manual	93000	f.Fuel-Diesel	
19988	BMW- 5 Series	2011	11995	f.Trans-Automatic	85896	f.Fuel-Diesel	
33500	Mercedes- E Class	2011	7490	f.Trans-Manual	89000	f.Fuel-Diesel	
47421	VW- Up	2012	4495	f.Trans-Manual	85000	f.Fuel-Petrol	
	tax	mpg	engineSize	manufacturer	age	outs	f.miles
7901	153.0634	68.9	1.6	Audi	10	2	f.miles-(34.1,119]
8052	157.2018	62.8	2	Audi	9	2	f.miles-(34.1,119]
9622	157.6048	68.9	1.6	Audi	11	3	f.miles-(34.1,119]
9730	200.0000	48.7	2	Audi	10	3	f.miles-(34.1,119]
10514	148.9249	74.3	1.6	Audi	8	2	f.miles-(34.1,119]
10662	145.0000	53.3	2	Audi	10	2	f.miles-(34.1,119]
19638	157.8682	62.8	2	BMW	9	2	f.miles-(34.1,119]
19988	165.0000	50.4	3	BMW	10	2	f.miles-(34.1,119]
33500	160.0000	52.3	2.1	Mercedes	10	2	f.miles-(34.1,119]
47421	158.4613	60.1	1	VW	9	2	f.miles-(34.1,119]
	f.tax	f.mpg	f.age			f.price	Audi
7901	f.tax-(145,155]	f.mpg-alto	f.age-(+4)			f.price-[899,1.1e+04]	Audi Yes
8052	f.tax-(155,205]	f.mpg-alto	f.age-(+4)			f.price-[899,1.1e+04]	Audi Yes
9622	f.tax-(155,205]	f.mpg-alto	f.age-(+4)			f.price-[899,1.1e+04]	Audi Yes
9730	f.tax-(155,205]	f.mpg-bajo	f.age-(+4)			f.price-[899,1.1e+04]	Audi Yes
10514	f.tax-(145,155]	f.mpg-alto	f.age-(+4)			f.price-[899,1.1e+04]	Audi Yes
10662	f.tax-(144,145]	f.mpg-bajo	f.age-(+4)			f.price-[899,1.1e+04]	Audi Yes
19638	f.tax-(155,205]	f.mpg-alto	f.age-(+4)			f.price-[899,1.1e+04]	Audi No
19988	f.tax-(155,205]	f.mpg-bajo	f.age-(+4)	f.price-(1.1e+04,1.4e+04]		Audi No	
33500	f.tax-(155,205]	f.mpg-bajo	f.age-(+4)	f.price-[899,1.1e+04]		Audi No	
47421	f.tax-(155,205]	f.mpg-medio	f.age-(+4)	f.price-[899,1.1e+04]		Audi No	
	mout						
7901	NoMOut						
8052	NoMOut						
9622	NoMOut						
9730	NoMOut						
10514	NoMOut						
10662	NoMOut						
19638	NoMOut						
19988	NoMOut						

```
33500 NoMOut  
47421 NoMOut
```

Como se puede apreciar, la mayoría tienen valores altos para las variables age y mileage, que son las que más se relacionan con la primera componente. Contrariamente, por norma general, estos vehículos también tienen precios menores, cosa que era esperable si tenemos en cuenta que la proyección de la variable price en el plano formado por las dos componentes principales tiene sentido negativo.

7.2 Análisis según las variables

7.2.1 Variables numéricas

Desde el punto de vista de las variables, en la salida a continuación, podemos ver los valores correspondientes al cos2 para cada una de las componentes principales en las dos primeras columnas, así como las contribuciones que tienen hacia estas en las dos últimas. Cabe destacar las contribuciones de las variables age y mileage para la primera dimensión.

```
round(cbind(res.pca$var$cos2[,1:2],res.pca$var$contrib[,1:2]),2)
```

	Dim.1	Dim.2	Dim.1	Dim.2
mileage	0.84	0.03	39.83	2.37
tax	0.00	0.85	0.23	69.67
mpg	0.41	0.32	19.76	25.97
age	0.84	0.02	40.19	2.00

A continuación podemos ver las correlaciones de todas nuestras variables numéricas con la primera componente principal. Cabe destacar las fuertes correlaciones con age y mileage, como habíamos mencionado previamente.

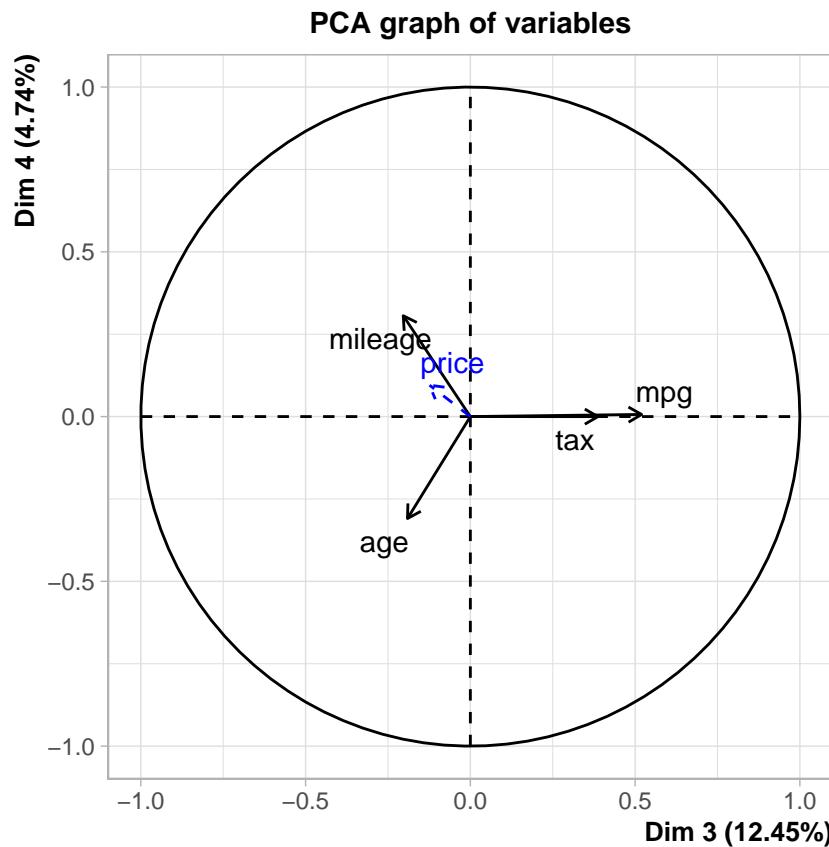
```
res.des<-dimdesc(res.pca)
```

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

	correlation	p.value
age	0.9185425	0.000000e+00
mileage	0.9143891	0.000000e+00
mpg	0.6440027	0.000000e+00
tax	0.0692207	1.545501e-06
price	-0.6974886	0.000000e+00

Por último, y aunque hemos afirmado que, según el criterio de Kaiser, solo eran necesarias 2 componentes principales, vamos a analizar graficamente las componentes 3 y 4. Podemos apreciar como las variables tax y mpg tienen gran importancia en la tercera componente, mientras que para la cuarta tienen más importancia mileage y age.

```
plot.PCA(res.pca, choix=c("var"), axes=c(3,4))
```

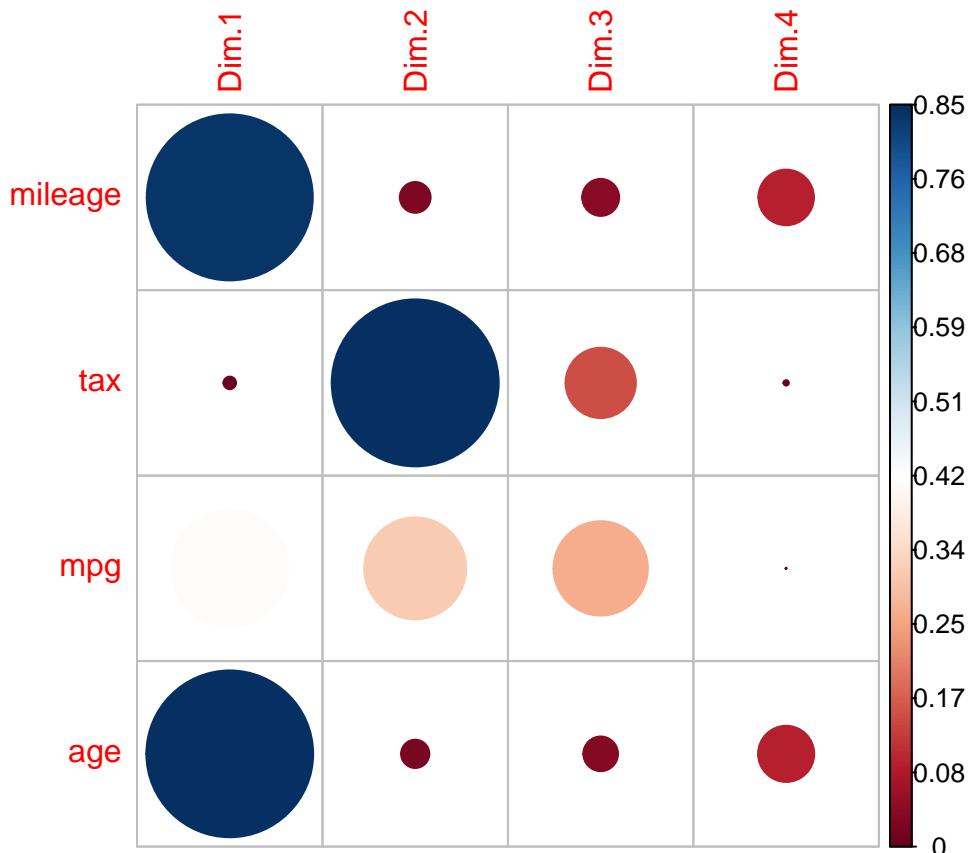


En el siguiente gráfico, podemos ver gráficamente y de manera resumida la importancia que tienen cada una de las variables numéricas en las diferentes componentes principales.

```
library("corrplot")
```

```
corrplot 0.90 loaded
```

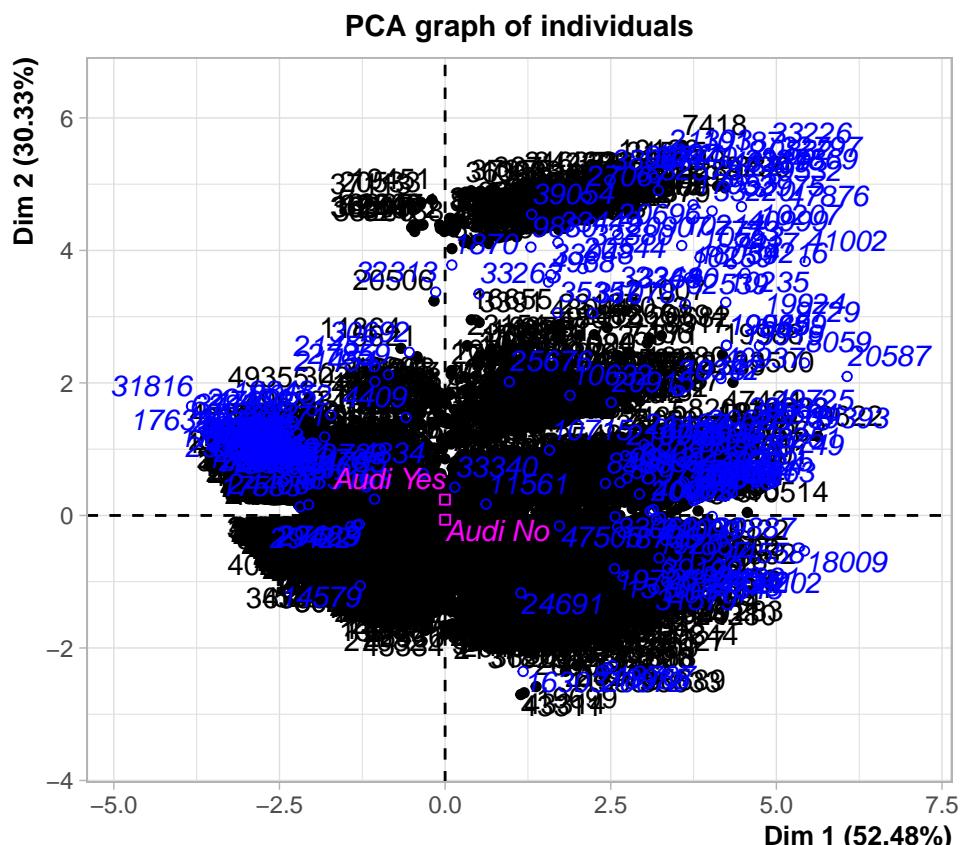
```
corrplot(res.pca$var$cos2, is.corr=FALSE)
```

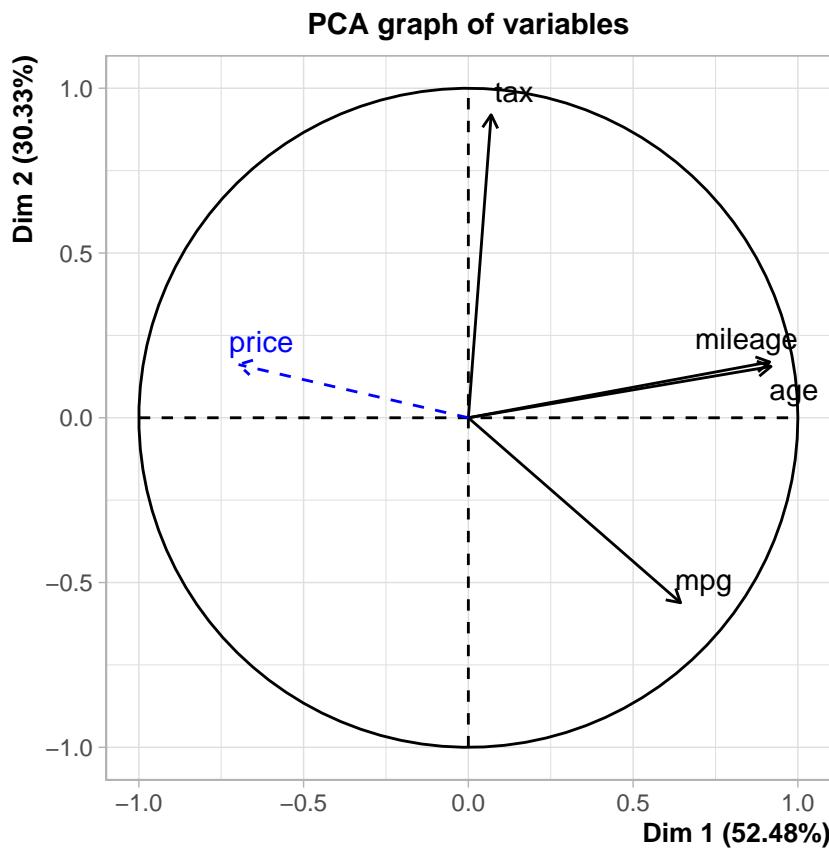


7.2.2 Targets

En el siguiente plot, podemos apreciar como nuestro target categórico AUDI, tiene más relación con la segunda dimensión que con la primera.

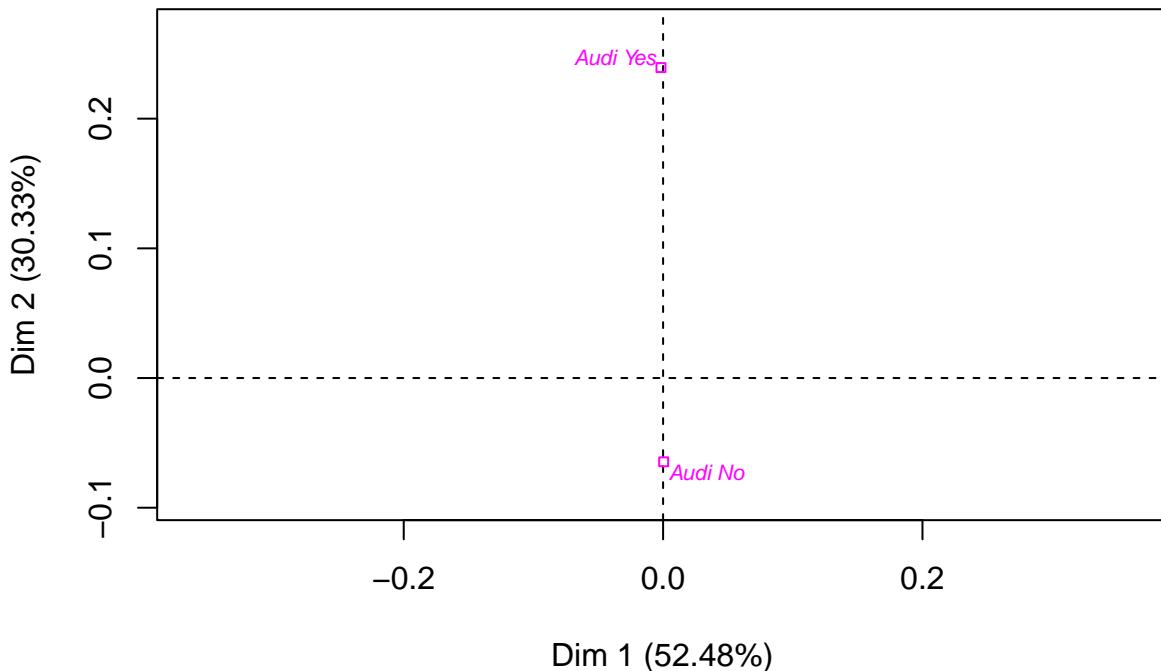
```
11 <- which( df$mout == "YesMOut")
res.pca<-PCA(df[,c(vars_res, vars_num)],quali.sup=c(2),quanti.sup= c(1), ind.sup = 11, ncp=2)
```





```
plot(res.pca, choix="ind", invisible=c("ind", "ind.sup"), cex=0.7, graph.type = "classic")
```

PCA graph of individuals



Podemos ver como la componente más representativa de este target es la segunda, ya que las coordenadas que aparecen vienen dadas por la correlación.

```
res.pca$quali.sup$coord
```

	Dim.1	Dim.2
"Audi Yes"	0.05	0.18
"Audi No"	-0.05	-0.08

```
Audi No  0.0004390175 -0.06458903
Audi Yes -0.0016271999  0.23939654
```

7.2.3 Factores

A continuación podemos ver los coeficientes R-squared que aparecen para nuestros factores: Cabe destacar la variabilidad de la componente que viene explicada por la variabilidad de factores como year o f.miles. Si lo analizamos, nos daremos cuenta de que estos factores se crearon en la entrega anterior a partir de las variables mileage y age, de modo que es coherente que los estos factores derivados también tengan una alta explicabilidad de la varianza de la primera componente.

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

	R2	p.value
year	0.85314874	0.000000e+00
f.price	0.54270410	0.000000e+00
f.miles	0.77911185	0.000000e+00
f.mpg	0.38649278	0.000000e+00
f.tax	0.35082285	0.000000e+00
f.age	0.75558367	0.000000e+00
engineSize	0.10996048	3.871403e-107
fuelType	0.09454849	2.115918e-104
model	0.12001479	1.179578e-80
transmission	0.05923874	1.810498e-64

Para la segunda componente, podemos ver como los factores más relevantes son f.tax y f.mpg.

```
res.des$Dim.2$quali
```

	R2	p.value
f.mpg	0.28941823	0.000000e+00
f.tax	0.72028695	0.000000e+00
engineSize	0.17425329	1.510644e-183
model	0.21173430	1.646812e-183
year	0.06701187	5.182795e-64
fuelType	0.03378855	1.321252e-36
f.price	0.03598778	1.302880e-34
transmission	0.02814738	1.576104e-30
f.miles	0.02719812	1.534633e-28
f.age	0.02011902	6.096614e-22
manufacturer	0.01745050	3.137067e-18
Audi	0.01274309	4.090000e-15

En resumen, hemos podido ver que tras realizar el PCA, las dos primeras componentes son capaces que aglutinar un 82.73% de la variabilidad de nuestro target. Las dos variables que más peso tienen para la primera componente son mileage y age, mientras que para la segunda, la variable más relevante es tax.

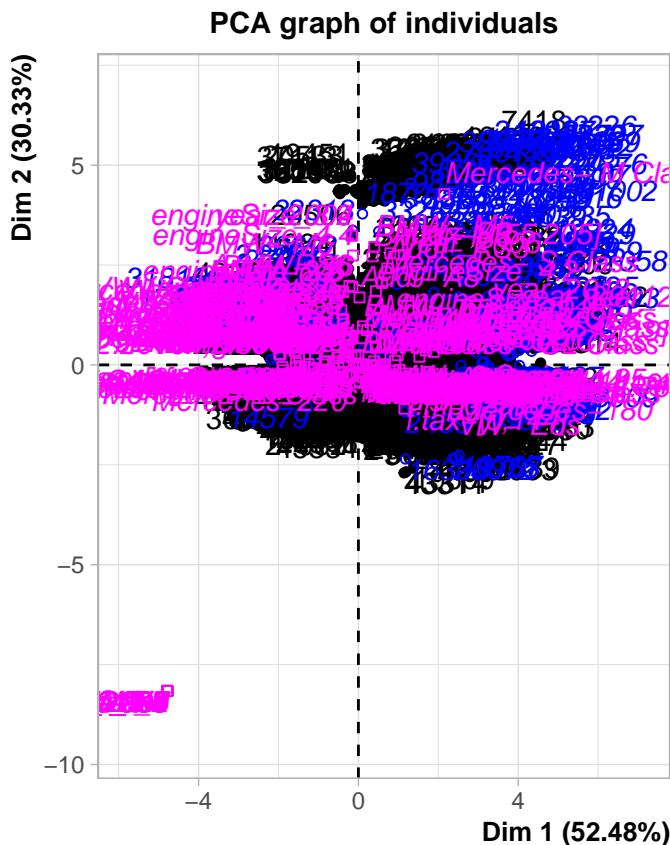
También podemos ver como al añadir los factores como suplementarios, los factores derivados de las variables numéricas adquieren una alta correlación con aquellas componentes donde sus variables numéricas asociadas tienen un peso mayor.

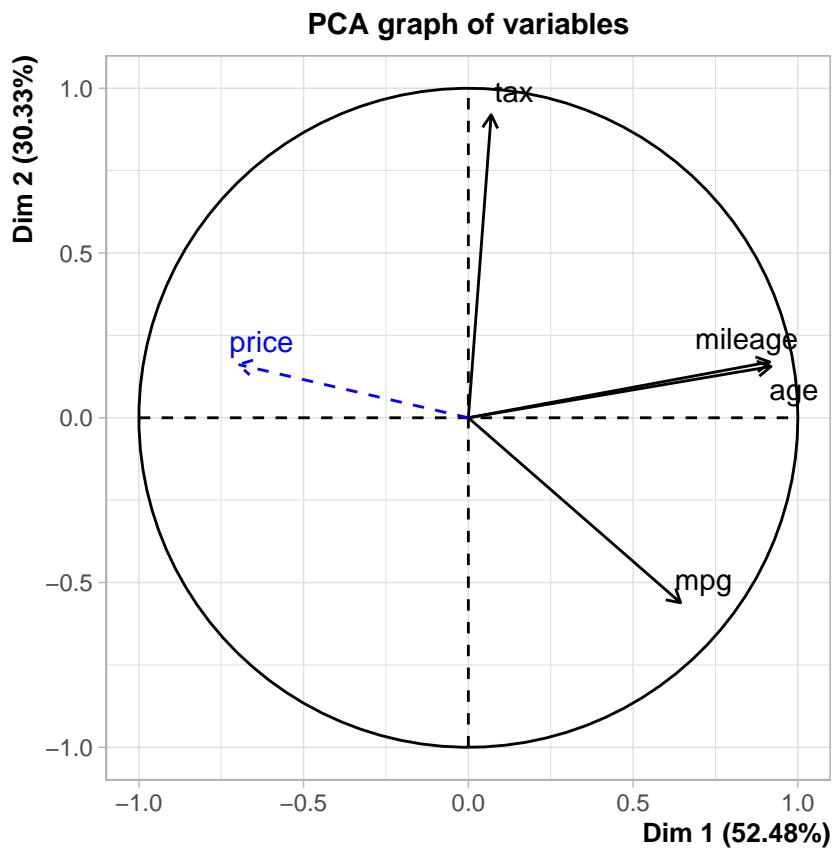
8 Hierarchical Clustering

A continuación, y para ser prácticos, vamos a proceder a realizar el proceso de clustering jerárquico, a partir del cual vamos a determinar (o al menos aproximar) el número óptimo de clusters para ejecutar el clustering con K-means.

En primer lugar, vamos a volver a ejecutar el PCA con las variables categóricas como suplementarias y quedándonos solo con las dos primeras componentes principales, como se ha determinado anteriormente con el criterio de Kaiser para el ACP.

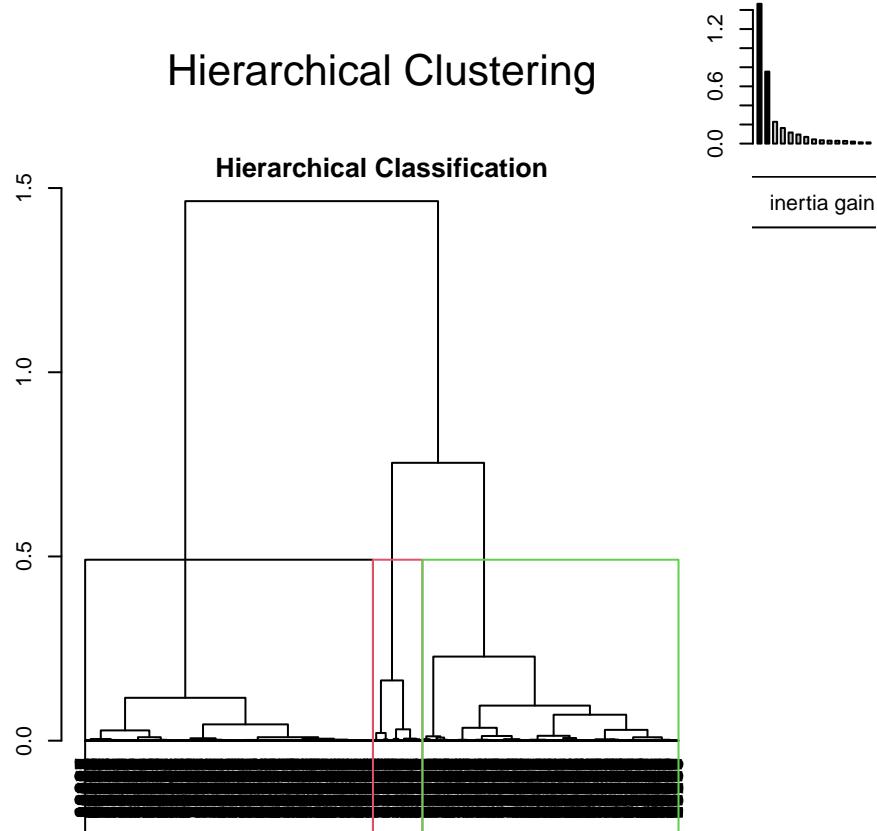
```
res.pca<-PCA(df[,c(vars_res, vars_cat, vars_num)],quali.sup=c(2:13),quanti.sup= c(1), ind.sup = 11, ncp
```



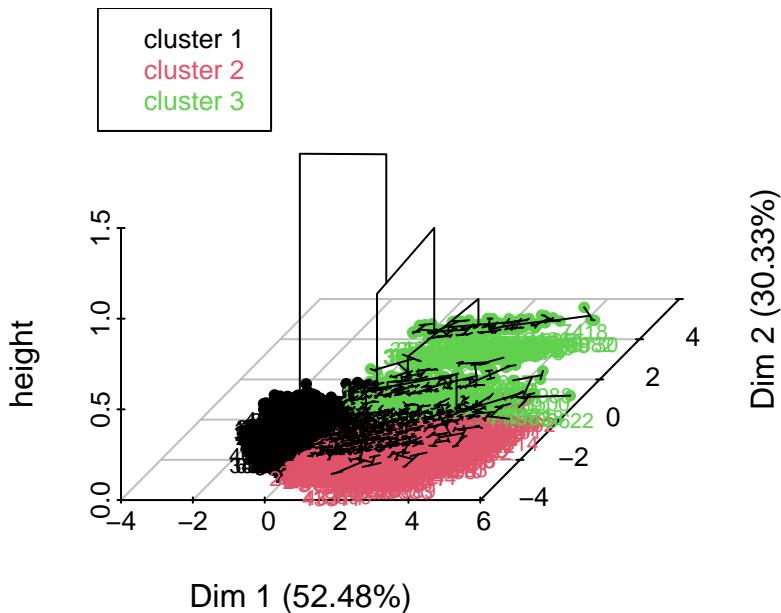


A continuación ejecutaremos el HCPC a partir del ACP anterior. Con el parámetro `nb.clust=-1` indicamos al sistema que tome el número óptimo de clusters según la partición con la que el decreto relativo de inercia es más alto. (Según la documentación - $(i(\text{clusters } n+1)/i(\text{cluster } n))$). Podemos ver que para este caso, se ha seleccionado como óptima una partición de 3 clusters.

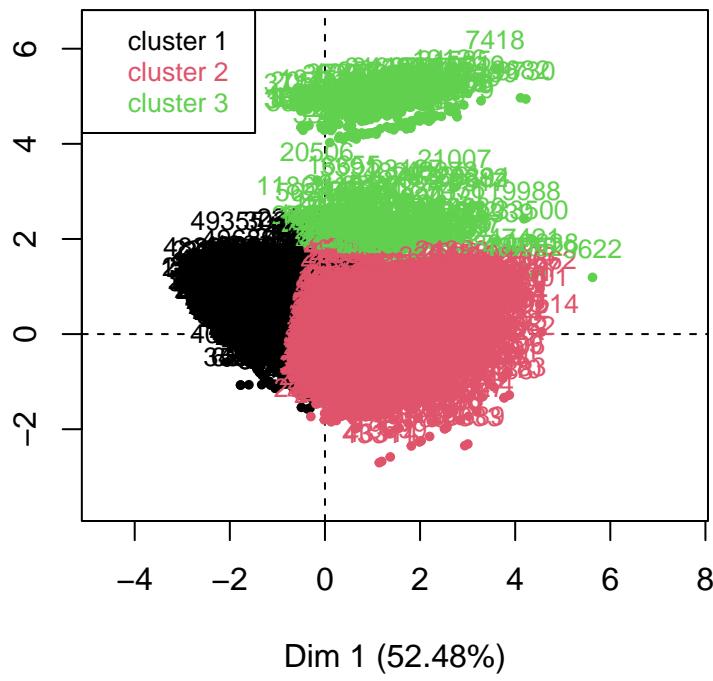
```
res.hcpc<-HCPC(res.pca, order=TRUE, nb.clust=-1)
```



Hierarchical clustering on the factor map



Factor map



Si aplicamos el criterio de Kaiser, podemos ver que dos componentes son suficientes.

```
length(which(res.hcpc$call$t$res$eig[,1] > mean(res.hcpc$call$t$res$eig[,1])))
```

[1] 2

Según los dos criterios aplicados (el que aplica el sistema con el parámetro `nb.clust=-1` y el de Kaiser) obtenemos que el número óptimo de componentes es 2 o 3. Procederemos con 3 componentes ya que así aglomeramos mayor variabilidad.

Ejecutando el siguiente comando, podemos ver como se relacionan las dos componentes principales a partir de las cuales hemos generado la clusterización, con los diferentes clusters que se han generado. Se puede apreciar como para el cluster 1, la coordenada de la primera componente principal es significativamente más baja que para el conjunto del datafrfame. En el caso del cluster 2, ambas componentes principales tienen un valor mayor. Por último, para el cluster 3, la primera componente tiene un valor mayor mientras que la segunda tiene un valor menor.

```
res.hpc$desc.axes
```

```
Link between the cluster variable and the quantitative variables
=====
      Eta2 P-value
Dim.1 0.7135508      0
Dim.2 0.6205426      0

Description of each cluster by quantitative variables
=====
$'1'
      v.test Mean in category Overall mean sd in category Overall sd p.value
Dim.1 -58.57292      -1.223298 1.456834e-14      0.5932563   1.448916      0

$'2'
      v.test Mean in category Overall mean sd in category Overall sd
Dim.1 50.45816       1.2144791 1.456834e-14      0.9006260   1.448916
Dim.2 -28.09612      -0.5141178 -5.036388e-15      0.6837658   1.101542
      p.value
Dim.1 0.000000e+00
Dim.2 1.092573e-173

$'3'
      v.test Mean in category Overall mean sd in category Overall sd
Dim.2 52.19387       3.020491 -5.036388e-15      1.401495    1.101542
Dim.1 16.87655       1.284648 1.456834e-14      1.046601    1.448916
      p.value
Dim.2 0.000000e+00
Dim.1 6.694136e-64
```

8.1 Análisis según las variables

A continuación vamos a ver como se relacionan las variables originales del dataframe con los clusters que se han generado.

En primer lugar, podemos ver el número de individuos que se han asignado a cada cluster.

```
summary(res.hpc$data.clust$clust)
```

1	2	3
2406	2067	337

8.1.1 Factores

A partir del test de chi2, se puede determinar que factores diferencian los clusters que se han generado.

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

	p.value	df
year	0.000000e+00	24
f.price	0.000000e+00	14
f.miles	0.000000e+00	6
f.mpg	0.000000e+00	6
f.tax	0.000000e+00	6

```

f.age      0.000000e+00  4
engineSize 7.037819e-239 36
model      4.695962e-176 168
transmission 8.065894e-72  4
fuelType    1.955102e-56  4
manufacturer 6.084521e-10 6
Audi       2.202681e-05  2

```

Si profundizamos un poco mas en esto, podemos ver como caracterizan los valores de las variables cualitativas los distintos clusters que se han generado.

<He omitido la salida porque ocupa mucho, pero abajo podemos ver algunas de las conclusiones>

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

Podemos destacar, por ejemplo la acumulación de coches nuevos y con poco kilometraje en el primer cluster(f.age=f.age-[1,2] -> Mod/Cla 76.0016694)

8.1.2 Variables numéricas

Con el test eta-squared, podemos determinar qué variables numéricas han sido influyentes a la hora de generar la clusterización.

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

	Eta2	P-value
price	0.3883781	0
mileage	0.5418861	0
tax	0.5847477	0
mpg	0.4712473	0
age	0.6611570	0

Podemos ver que las variables más representativas son age, mileage y tax, que son las mismas que aparecen como más determinantes a la hora de realizar el PCA.

Por último, podemos analizar como estas variables numéricas caracterizan los distintos clusters.

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

	v.test	Mean in category	Overall mean	sd in category	Overall sd
price	42.408212	26439.214464	20900.044491	8710.9679373	9061.554246
tax	-3.821674	145.860063	146.518027	2.6971094	11.944192
mpg	-39.459997	47.325298	53.429394	8.7110675	10.731794
mileage	-50.448418	7925.595836	21978.643932	7059.7796296	19325.509377
age	-55.696059	2.223816	3.712352	0.8097817	1.854142
	p.value				
price	0.00000000000				
tax	0.0001325488				
mpg	0.00000000000				
mileage	0.00000000000				
age	0.00000000000				
\$'1'					
	v.test	Mean in category	Overall mean	sd in category	Overall sd
price	42.408212	26439.214464	20900.044491	8710.9679373	9061.554246
tax	-3.821674	145.860063	146.518027	2.6971094	11.944192
mpg	-39.459997	47.325298	53.429394	8.7110675	10.731794
mileage	-50.448418	7925.595836	21978.643932	7059.7796296	19325.509377
age	-55.696059	2.223816	3.712352	0.8097817	1.854142
	p.value				
price	0.00000000000				
tax	0.0001325488				
mpg	0.00000000000				
mileage	0.00000000000				
age	0.00000000000				
\$'2'					
	v.test	Mean in category	Overall mean	sd in category	Overall sd
mpg	47.47148	61.892318	53.429394	6.931857	10.731794
age	44.00506	5.067731	3.712352	1.278477	1.854142
mileage	39.94247	34801.394298	21978.643932	16836.880029	19325.509377
tax	-22.90447	141.973461	146.518027	8.074326	11.944192
price	-40.95483	14735.185776	20900.044491	4640.286631	9061.554246
	p.value				
mpg	0.000000e+00				

```

age      0.000000e+00
mileage 0.000000e+00
tax      4.192817e-116
price    0.000000e+00

$'3'
      v.test Mean in category Overall mean sd in category  Overall sd
tax      51.907153     179.089779   146.518027   19.842975   11.944192
age      23.756408      6.026445    3.712352    1.385459    1.854142
mileage 21.356086    43661.162371  21978.643932  18640.176220 19325.509377
price   -3.643174    19165.682493  20900.044491  6556.740965  9061.554246
mpg     -14.770362    45.101780    53.429394    5.718682    10.731794
      p.value
tax      0.000000e+00
age      9.433547e-125
mileage 3.423205e-101
price   2.692964e-04
mpg     2.274878e-49

```

En el cluster 1, podemos apreciar como aparecen coches con precios más altos y menores kilometrajes y años, de modo que este cluster está formado por coches nuevos y con poco uso, que tienen asociado un precio más elevado.

En el cluster 2, podemos ver como aparecen coches más viejos y con más kilometraje, con un precio más bajo.

Por último, podríamos calificar el cluster 3 como el cluster ‘ECO’, ya que aparecen coches con un consumo muy bajo y que han pagado menos impuestos, puede que debido a subvenciones.

8.2 Análisis según los individuos

Vamos a analizar los parámetros de cada cluster que se ha derivado a partir del clustering jerárquico.

```
res.hcpc$desc.ind$para
```

```

Cluster: 1
  30871      34302      39747      46361      29137
0.004243074 0.028026589 0.040901743 0.041542905 0.043433428
-----
Cluster: 2
  20977      48747      24277      27898      19474
0.04811671 0.05306710 0.05418321 0.05541993 0.05853118
-----
Cluster: 3
  48041      23127      9145       48052      10617
0.3627738 0.4477281 0.5171342 0.5477070 0.5524531

```

Si analizamos los parámetros del primer cluster, podemos ver como aparecen coches nuevos de gasolina. Todos tienen kilometrajes e impuestos similares.

```
summary(df[c("30871", "24055", "28264", "48543", "34122"),])
```

	model	year	price	transmission	
Mercedes- C Class	3	2019 :5	Min. :20149	f.Trans-Manual :0	
Mercedes- B Class	1	2001 :0	1st Qu.:21890	f.Trans-SemiAuto :3	
VW- Arteon	:1	2002 :0	Median :25299	f.Trans-Automatic:2	
Audi- A1	:0	2003 :0	Mean :24357		
Audi- A3	:0	2004 :0	3rd Qu.:26056		
Audi- A4	:0	2005 :0	Max. :28391		
(Other)	:0	(Other):0			
	mileage	fuelType	tax	mpg	engineSize
Min.	:10061	f.Fuel-Diesel:0	Min. :145	Min. :45.60	1.5 :3
1st Qu.	:10096	f.Fuel-Petrol:5	1st Qu.:145	1st Qu.:46.30	1.3 :1
Median	:10682	f.Fuel-Hybrid:0	Median :145	Median :46.30	2 :1

```

Mean      :10883          Mean     :145   Mean    :46.32   1       :0
3rd Qu.:11785          3rd Qu.:145   3rd Qu.:46.30   1.2     :0
Max.     :11789          Max.    :145   Max.    :47.10   1.4     :0
                                         (Other):0

  manufacturer      age       outs           f.miles
Audi      :0      Min.    :2      Min.    :0      f.miles-[0.001,5.81]:0
BMW       :0      1st Qu.:2     1st Qu.:0      f.miles-(5.81,17.7] :5
Mercedes:4      Median :2     Median :0      f.miles-(17.7,34.1] :0
VW        :1      Mean    :2      Mean    :0      f.miles-(34.1,119] :0
                  3rd Qu.:2     3rd Qu.:0
                  Max.    :2      Max.    :0

f.tax            f.mpg         f.age
f.tax-[0,144] :0  f.mpg-muy bajo:1  f.age-[1,2]:5
f.tax-(144,145]:5  f.mpg-bajo     :4  f.age-(2,4]:0
f.tax-(145,155]:0  f.mpg-medio   :0  f.age-(+4) :0
f.tax-(155,205]:0  f.mpg-alto    :0

```

```

f.price          Audi      mout
f.price-(1.95e+04,2.2e+04]:2  Audi No :5  NoMOut :5
f.price-(2.6e+04,3.15e+04]:2  Audi Yes:0  YesMOut:0
f.price-(2.2e+04,2.6e+04] :1
f.price-[899,1.1e+04]       :0
f.price-(1.1e+04,1.4e+04] :0
f.price-(1.4e+04,1.7e+04] :0
(Other)                   :0

```

Por último, echaremos un vistazo a los individuos más típicos de los clusters:

```
res.hcpc$desc.ind$dist
```

```
Cluster: 1
 48339     17208     24771     23883     17600
4.215805  4.110782  4.081395  4.016452  3.943313
```

```
Cluster: 2
 17283     44210     2438      38689     35633
5.022955  5.017212  5.016667  4.827626  4.788841
```

```
Cluster: 3
 7418      9730      9882      19126     31585
6.543255  6.232543  6.200846  5.882428  5.880168
```

Si analizamos el primer cluster, podemos ver que todos son del 2020 y en su mayoría semi-automáticos. Además, tienen kilometrajes y consumos muy bajos. Todos están en el rango más alto de precio.

```
summary(df[c("31816","17634","17208","48339","24771")])
```

	model	year	price	transmission
BMW- X3	:1	2020	:5	Min. :33900 f.Trans-Manual :0
BMW- X4	:1	2001	:0	1st Qu.:43995 f.Trans-SemiAuto :4
Mercedes- A Class	:1	2002	:0	Median :49980 f.Trans-Automatic:1
Mercedes- GLC Class	:1	2003	:0	Mean :47473
VW- Touareg	:1	2004	:0	3rd Qu.:52991
Audi- A1	:0	2005	:0	Max. :56499
(Other)	:0	(Other):0		
mileage	fuelType	tax	mpg	engineSize
Min. :345	f.Fuel-Diesel:0	Min. :135	Min. : 1.10	3 :3
1st Qu.:2000	f.Fuel-Petrol:3	1st Qu.:140	1st Qu.: 5.50	1.3 :1
Median :3999	f.Fuel-Hybrid:2	Median :145	Median :25.50	2 :1

```

Mean      :3141          Mean     :143    Mean    :17.08   1       :0
3rd Qu.:4360          3rd Qu.:145   3rd Qu.:26.40   1.2     :0
Max.     :5000          Max.    :150    Max.    :26.90   1.4     :0
                                         (Other):0

  manufacturer      age       outs           f.miles
Audi      :0      Min.    :1      Min.   :1.0    f.miles-[0.001,5.81]:5
BMW       :2      1st Qu.:1     1st Qu.:1.0   f.miles-(5.81,17.7] :0
Mercedes:2     Median :1     Median :1.0    f.miles-(17.7,34.1] :0
VW        :1      Mean    :1      Mean   :1.2    f.miles-(34.1,119]  :0
                           3rd Qu.:1     3rd Qu.:1.0
                           Max.    :1      Max.   :2.0

f.tax            f.mpg         f.age
f.tax-[0,144] :2  f.mpg-muy bajo:5  f.age-[1,2]:5
f.tax-(144,145]:2 f.mpg-bajo     :0  f.age-(2,4]:0
f.tax-(145,155]:1 f.mpg-medio   :0  f.age-(+4) :0
f.tax-(155,205]:0 f.mpg-alto   :0

```

```

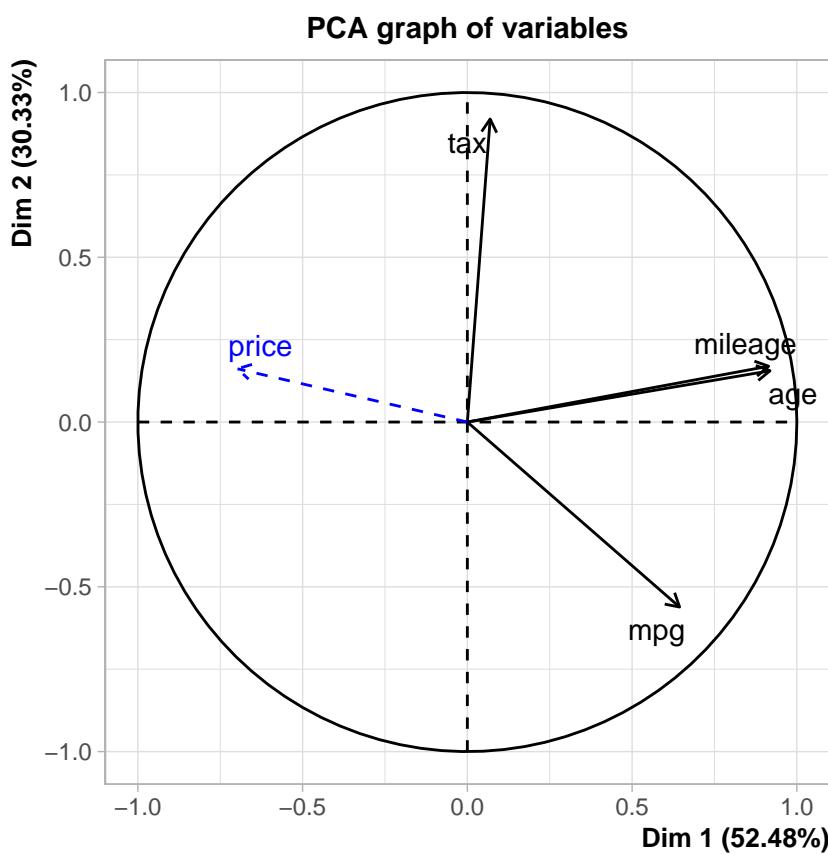
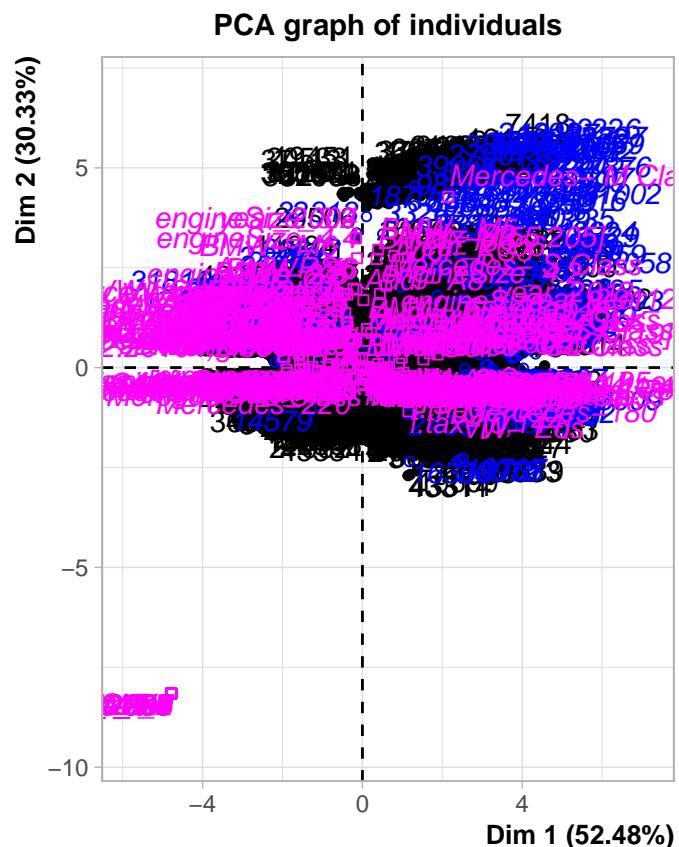
f.price          Audi      mout
f.price-(3.15e+04,1.09e+05]:5  Audi No :5  NoMOut :3
f.price-[899,1.1e+04]      :0  Audi Yes:0  YesMOut:2
f.price-(1.1e+04,1.4e+04]    :0
f.price-(1.4e+04,1.7e+04]    :0
f.price-(1.7e+04,1.95e+04]   :0
f.price-(1.95e+04,2.2e+04]   :0
(Other)                  :0

```

Podemos ver amplias diferencias entre los individuos típicos de los clusters y los parámetros.

9 K-means Clustering desde ACP

```
res.pca<-PCA(df[,c(vars_res, vars_cat, vars_num)],quali.sup=c(2:13),quanti.sup= c(1), ind.sup = 11, ncp=
```



```
ppcc <- res.pca$ind$coord
kc <- kmeans (dist(ppcc),3)
df[-11, "claKMPCA"] <- kc$cluster
```

Podemos ver como dentro de nuestra variable kc, se guardan datos como el cluster al que se asigna cada elemento, las distancias en el interior de los clusters o las distancias entre clusters.

```
summary(kc)
```

	Length	Class	Mode
cluster	4810	-none-	numeric
centers	14430	-none-	numeric
totss	1	-none-	numeric
withinss	3	-none-	numeric
tot.withinss	1	-none-	numeric
betweenss	1	-none-	numeric
size	3	-none-	numeric
iter	1	-none-	numeric
ifault	1	-none-	numeric

Estos valores nos ayudan a determinar la calidad de la cluserización, ya que idealmente, queremos clusters con elementos muy juntos entre si y mucha diferenciación entre clusters. Para var la calidad de la clusterización, vamos a realizar el cociente de las distancias entre clusters entre la suma de todas las distancias.

```
kc$betweenss/kc$totss
```

```
[1] 0.7034349
```

Podemos ver que la suma de las distancias entre los clusters suma un 70% del total.

Por otro lado, si comprobamos la suma de las distancias dentro de los clusters, podemos ver como estas tan solo suman el 30% del total.

```
kc$tot.withinss/kc$totss
```

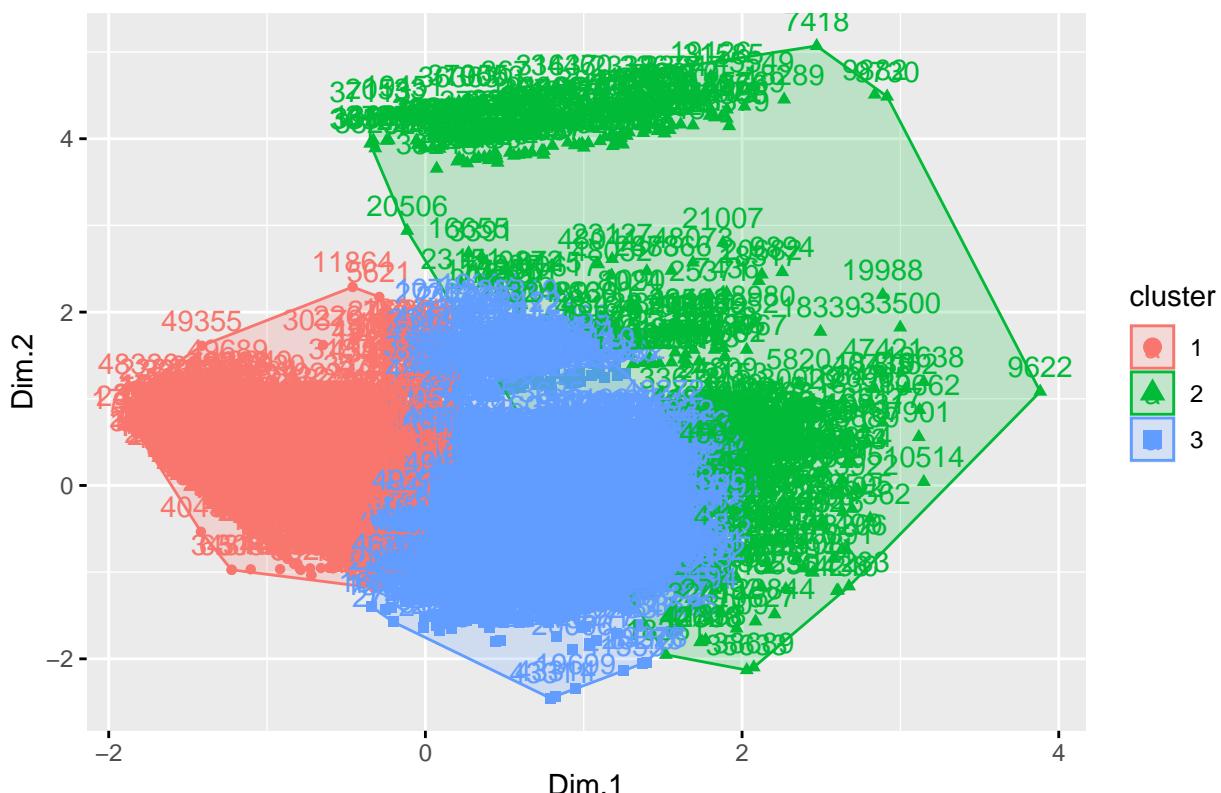
```
[1] 0.2965651
```

De este modo, podemos determinar que la calidad de la clusterización es relativamente buena ya que la distancia entre clusters es grande (los clusters están diferenciados entre si), pero las distancias dentro de los clusters son pequeñas (los clusters están formados por elementos muy parecidos).

A continuación, vamos a mostrar un plot donde se muestran claramente los clusters de diferentes colores.

```
fviz_cluster(kc, data=ppcc)
```

Cluster plot



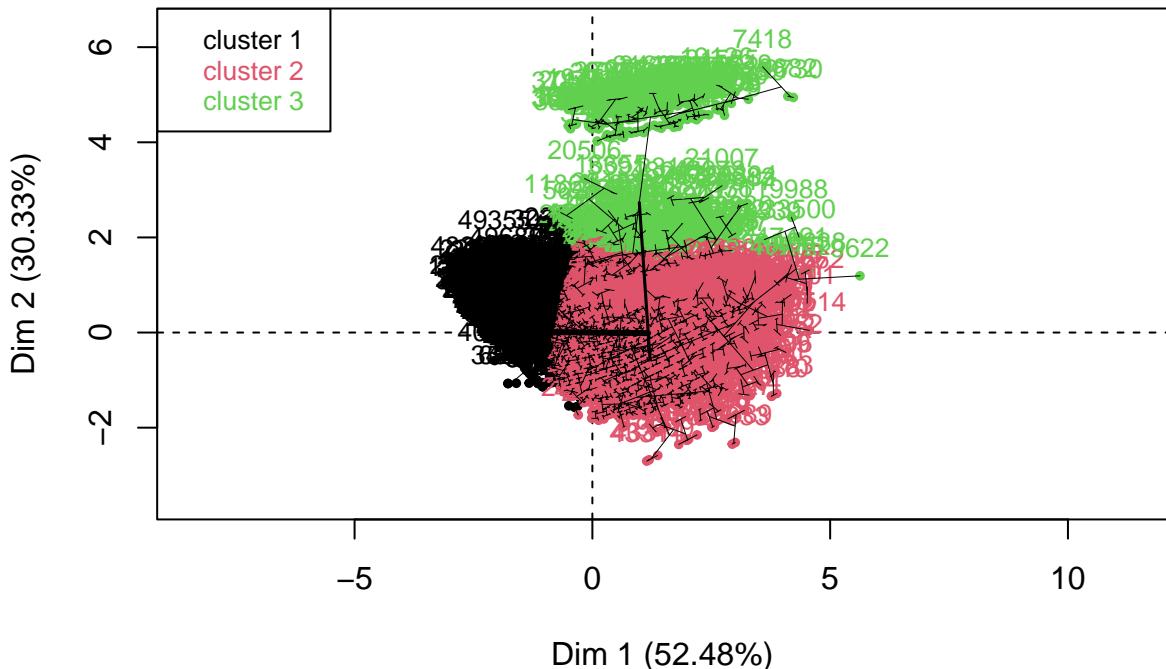
En

este gráfico podemos ver como no existe una definición tan clara como esperábamos en la clusterización.

Por último, vamos a volver a mostrar el gráfico resultante del clustering jerárquico, para poder comparar mejor los resultados:

```
plot.HCPC(res.hcpc, choice="map")
```

Factor map



Podemos ver como los dos procesos de clusterización han llevado a resultados realmente diferentes. A primera vista, se puede ver como la clusterización jerárquica da un resultado mucho más comprensible en el plano de las dos componentes que se han usado para realizar la clusterización.

Por otro lado, en los dos procesos se ha definido el primer cluster de manera muy similar.

Vamos a analizar los clusters que se han creado a partir de K-Means.

```
df$claKMPCA <- factor(df$claKMPCA)
km.catdes<-catdes(df[,c("claKMPCA",vars_num,vars_cat, "price")],1)
km.catdes$quanti
```

```
$'1'
      v.test Mean in category Overall mean sd in category    Overall sd
price   34.329921    26738.684712 21206.729400    8699.9825065 10553.646562
tax     -5.163384     145.891664   146.865790     2.6147948   12.355981
mpg    -36.772788     46.827719   53.097238     8.4742982   11.166173
mileage -47.629470    7530.870325 23318.245205    6740.6064489 21708.552196
age     -52.880138     2.179948   3.798202     0.7799129   2.004245
          p.value
price   2.808252e-258
tax     2.425247e-07
mpg    5.027592e-296
mileage 0.000000e+00
age     0.000000e+00

$'2'
      v.test Mean in category Overall mean sd in category    Overall sd
tax     36.759013    167.996776 146.865790    26.040424   12.355981
mileage 33.360025    57010.954203 23318.245205   19223.537976 21708.552196
age     31.165791     6.704283   3.798202     1.450761   2.004245
mpg     2.158693     54.218673  53.097238    12.115629   11.166173
price   -13.320892   14666.174941 21206.729400   6895.921934 10553.646562
          p.value
tax     8.345927e-296
mileage 5.212628e-244
age     3.099439e-213
mpg     3.087400e-02
price   1.750034e-40

$'3'
      v.test Mean in category Overall mean sd in category    Overall sd
mpg     40.08561    60.604275  53.097238    7.541902   11.166173
age     29.98293     4.806064   3.798202     1.107078   2.004245
mileage 20.82747    30901.282053 23318.245205   13319.253151 21708.552196
tax     -19.41806    142.841782 146.865790     8.759779   12.355981
price   -31.21660   15681.327719 21206.729400   4822.603388 10553.646562
          p.value
mpg     0.000000e+00
age     1.638369e-197
mileage 2.440094e-96
tax     5.430444e-84
price   6.344015e-214

$'NA'
      v.test Mean in category Overall mean sd in category    Overall sd
mileage 21.95240    57231.308988 23318.245205   41848.097570 21708.552196
age     15.23806     5.971578   3.798202     3.710618   2.004245
price   10.33778    28970.700000 21206.729400   28100.779565 10553.646562
tax     10.01250     155.669668 146.865790    18.039506   12.355981
mpg    -10.58219    44.688441  53.097238    17.087612   11.166173
          p.value
mileage 8.214913e-107
age     1.976402e-52
price   4.754373e-25
tax     1.343117e-23
mpg     3.604523e-26
```

Podemos ver como el primer cluster esta formado por coches con consumo bajo y con impuestos bajos, lo que sería equivalente al cluster 3 ‘ECO’ que hemos obtenido en la clusterización jerárquica.

Para el cluster 2, podemos ver como aparecen coches más baratos pero viejos y con más kilometraje.

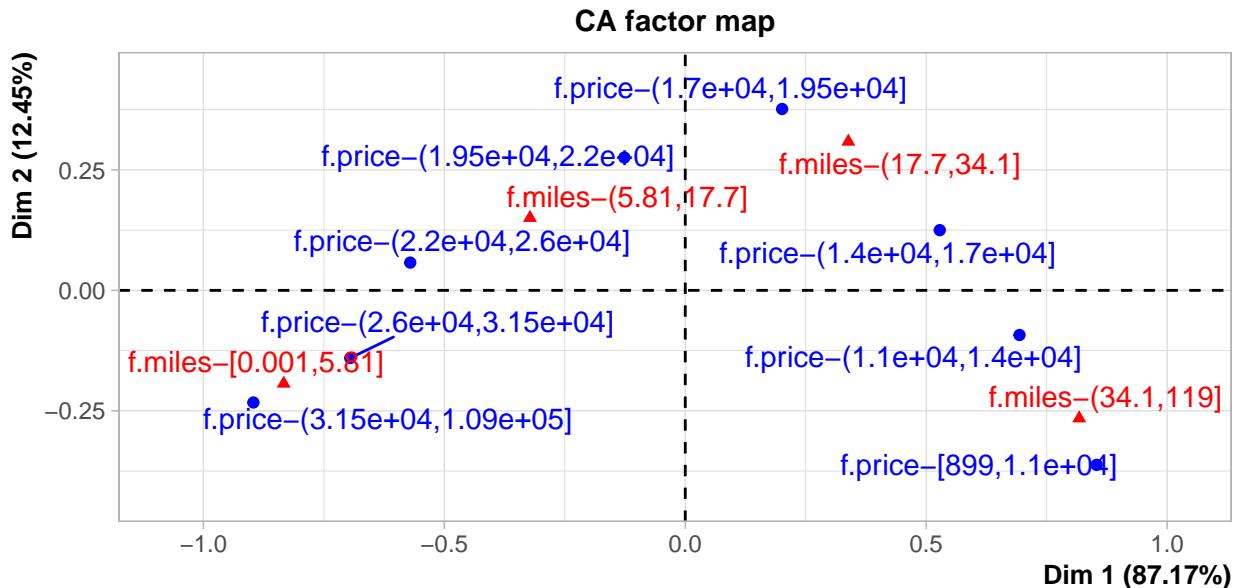
En el cluster 3, aparecen los coches más caros, nuevos y con menor kilometraje. Curioso que también tengan mayor consumo.

10 Correspondence Analysis

Vamos a proceder a realizar el análisis de correspondencias entre variables.

En primer lugar, vamos a analizar la correspondencia entre f.price y f.miles.

```
tt<-table(df[,c("f.price","f.miles")])  
res.ca<-CA(tt)
```



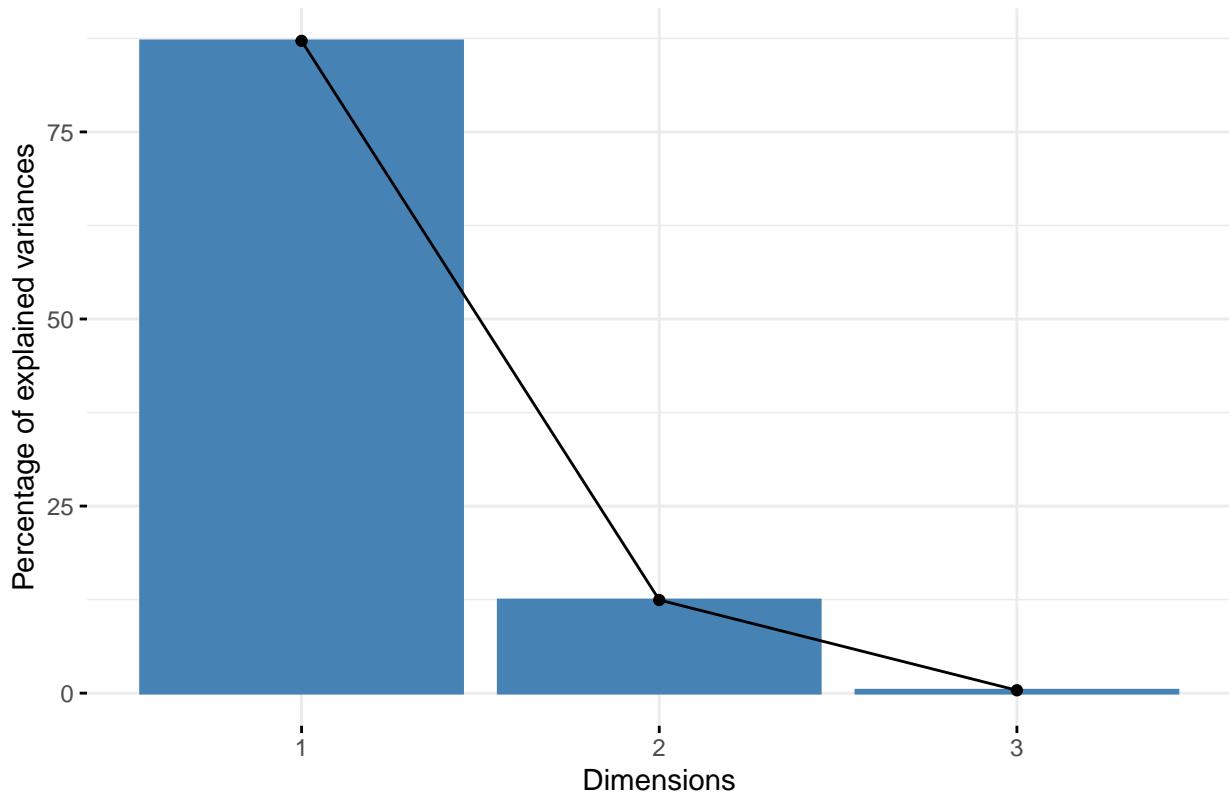
```
chisq.test(tt)
```

```
Pearson's Chi-squared test  
  
data: tt  
X-squared = 2267.6, df = 21, p-value < 2.2e-16
```

Podemos ver que la mayoría de la variabilidad se acumula en la primera componente (87%).

```
fviz_eig(res.ca)
```

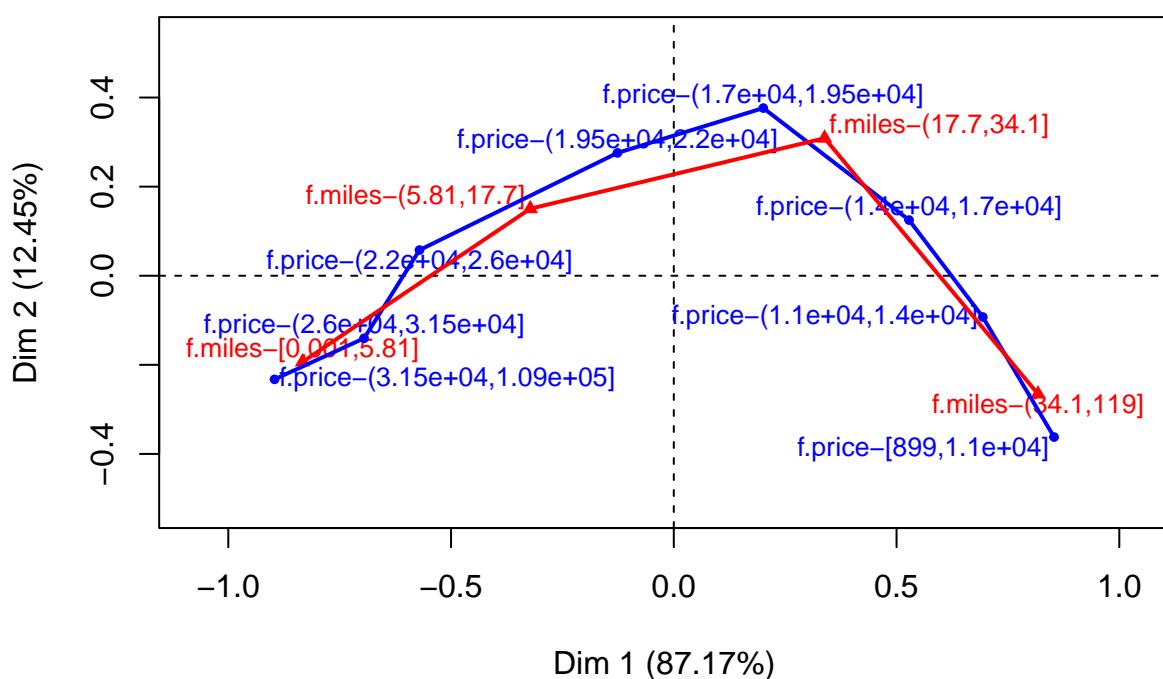
Scree plot



Si realizamos los plots, podemos ver la presencia del efecto Guttman que nos indica que las variables estan fuertemente relacionadas.

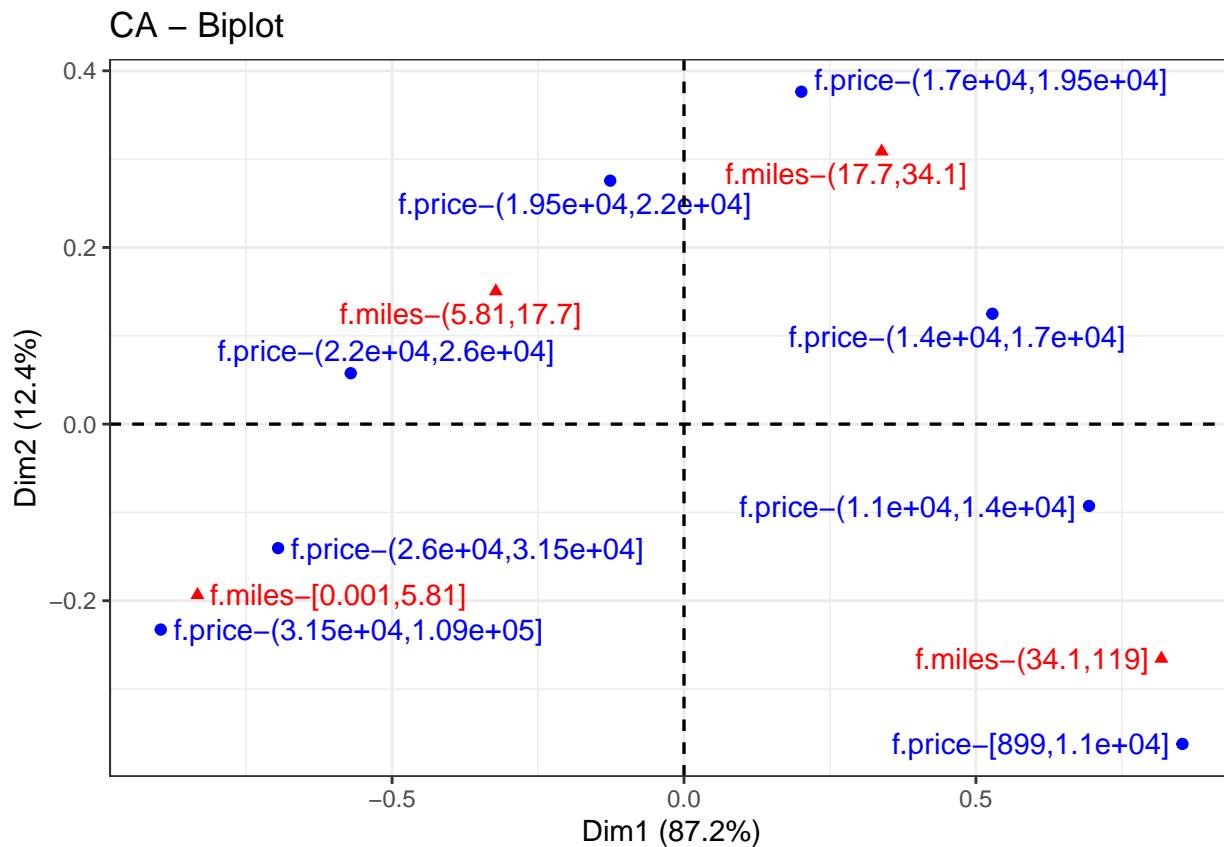
```
plot( res.ca, cex=0.8, graph.type = "classic" )
lines( res.ca$row$coord[,1], res.ca$row$coord[,2], col="blue", lwd = 2 )
lines( res.ca$col$coord[,1], res.ca$col$coord[,2], col="red", lwd = 2 )
```

CA factor map



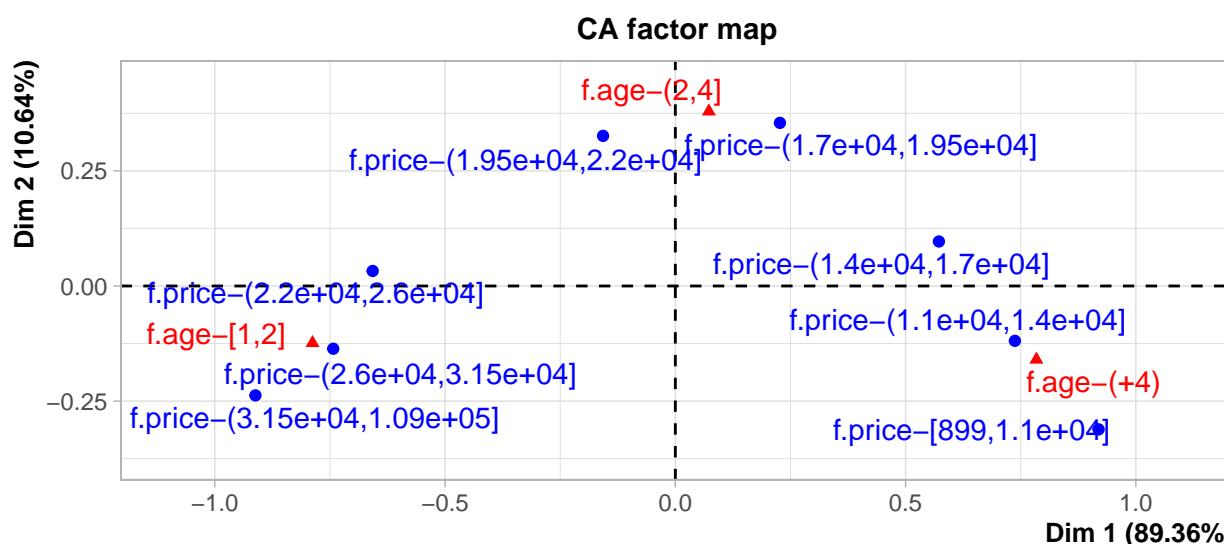
En el siguiente gráfico se puede ver como los coches con menor kilometraje tienen precios más altos y viceversa.

```
fviz_ca_biplot(res.ca, repel=TRUE)+theme_bw()
```



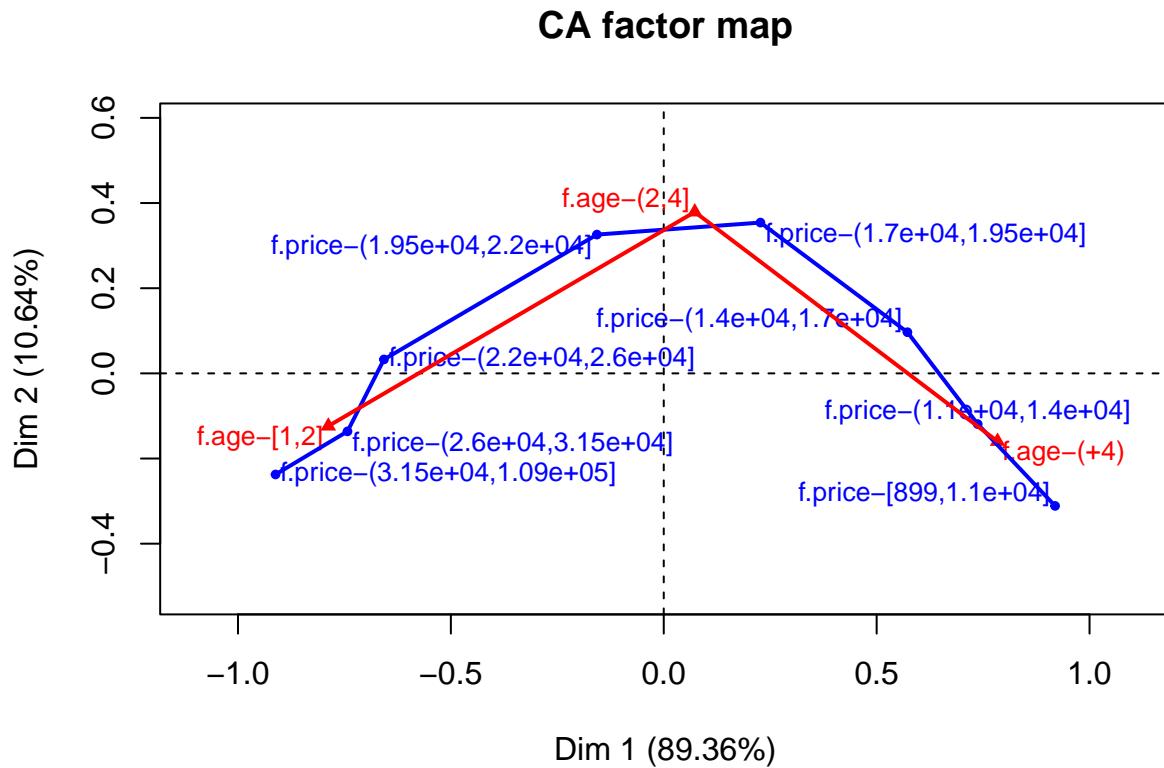
A continuación, vamos a realizar el mismo proceso para las variables f.price y f.age.

```
tt<-table(df[,c("f.price","f.age")])
res.ca<-CA(tt)
```



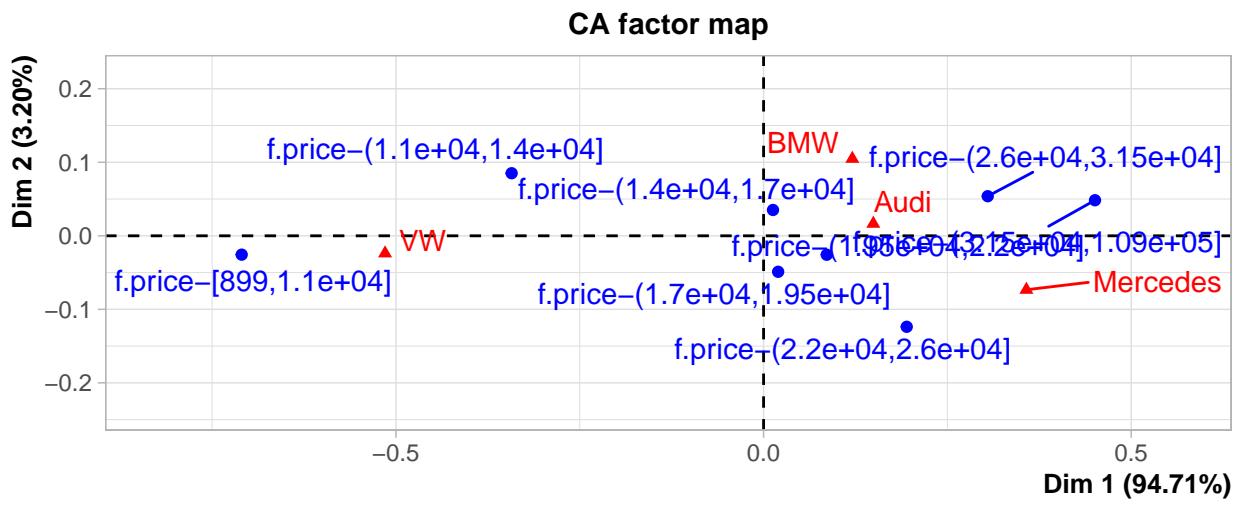
En este caso, también se puede ver la clara relación que hay entre las variables, donde los coches más nuevos son más caros y los viejos más baratos.

```
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 )
```



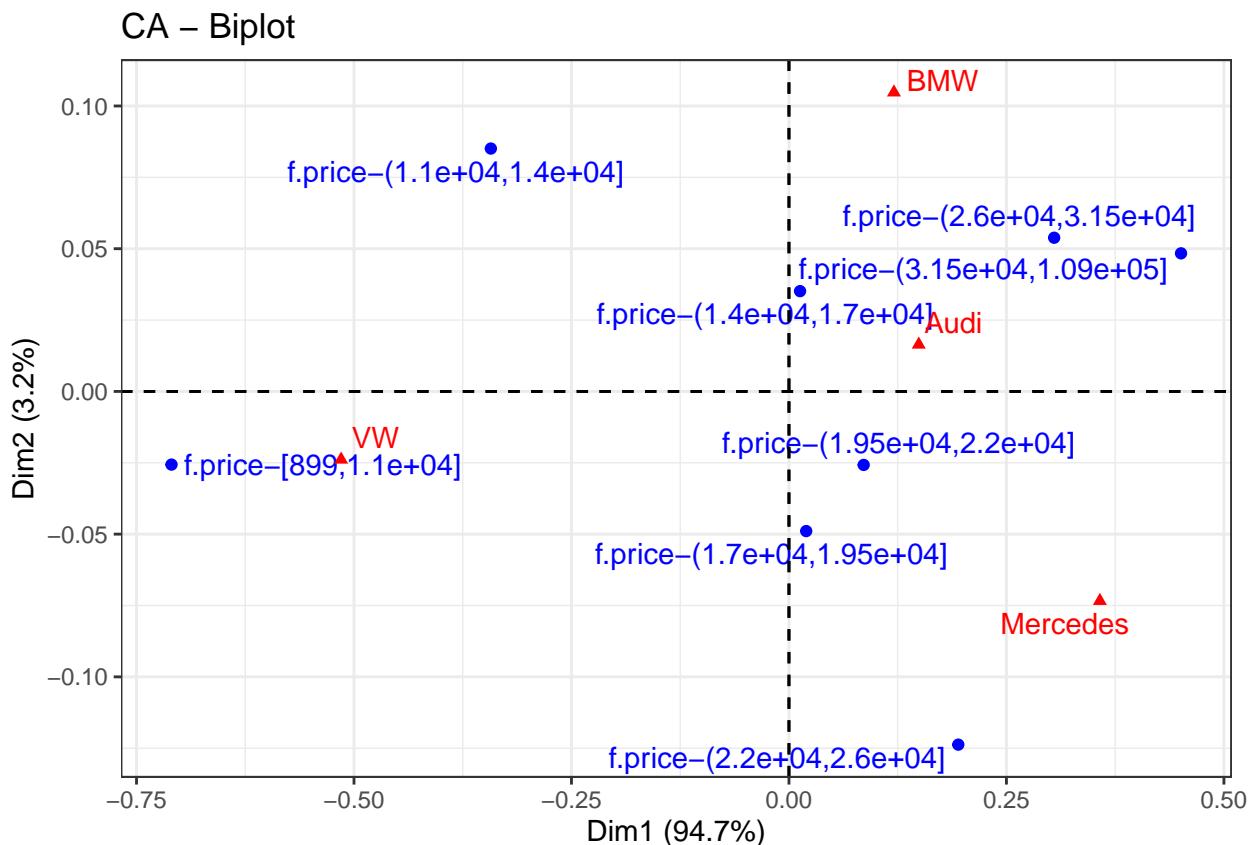
Por último, vamos a analizar la relación entre f.price y manufacturer. En este caso, se puede ver claramente los coches VW tienen precios más baratos que los BMW, Audi o Mercedes.

```
tt<-table(df[,c("f.price","manufacturer")])
res.ca<-CA(tt)
```



En este caso, en el siguiente gráfico se puede ver claramente los coches VW tienen precios más baratos que los BMW, Audi o Mercedes.

```
fviz_ca_biplot(res.ca, repel=TRUE)+theme_bw()
```



Es por este motivo, que factores como pueden ser manufacturer o engineSize pueden no resultar determinantes a la hora de explicar nuestro target numérico price.

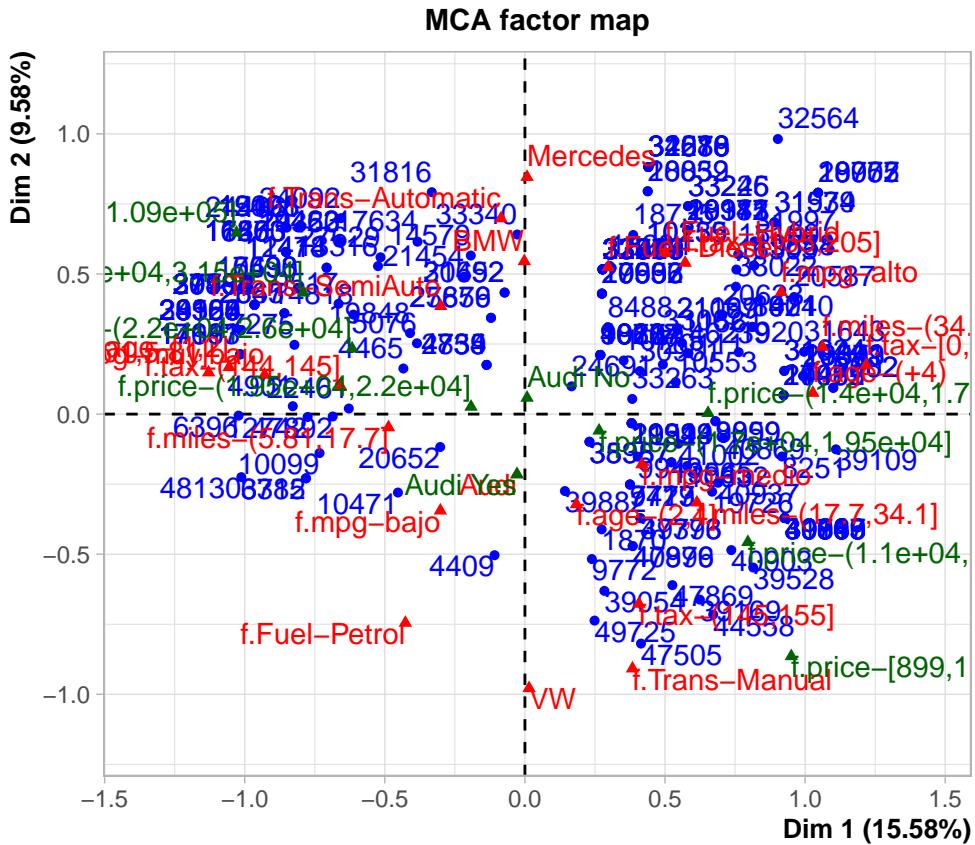
11 Multiple Correspondence Analysis

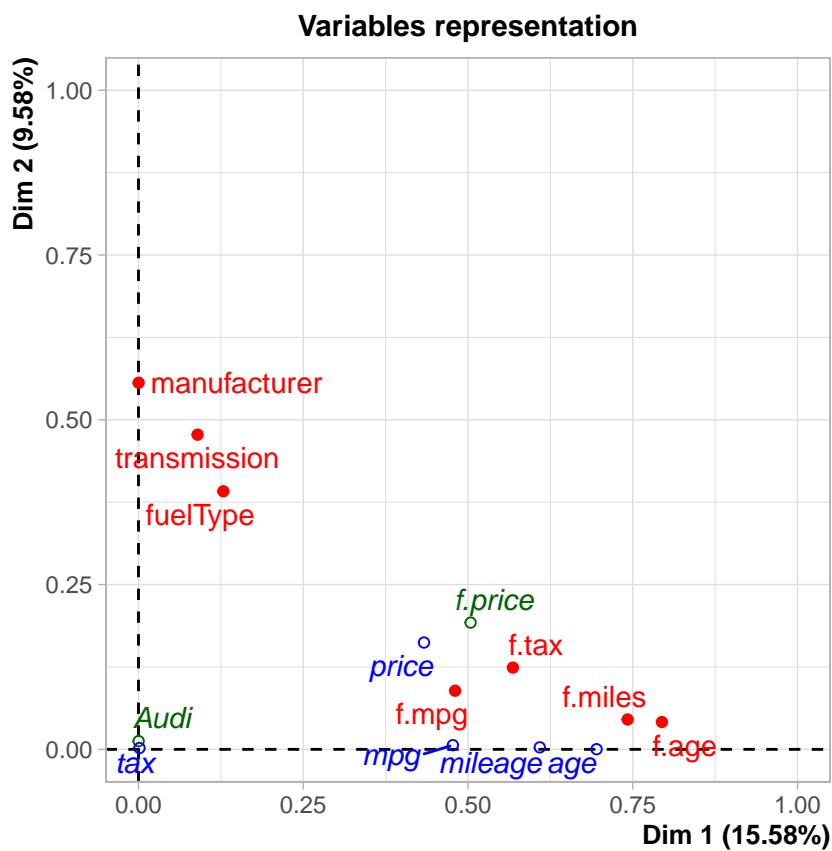
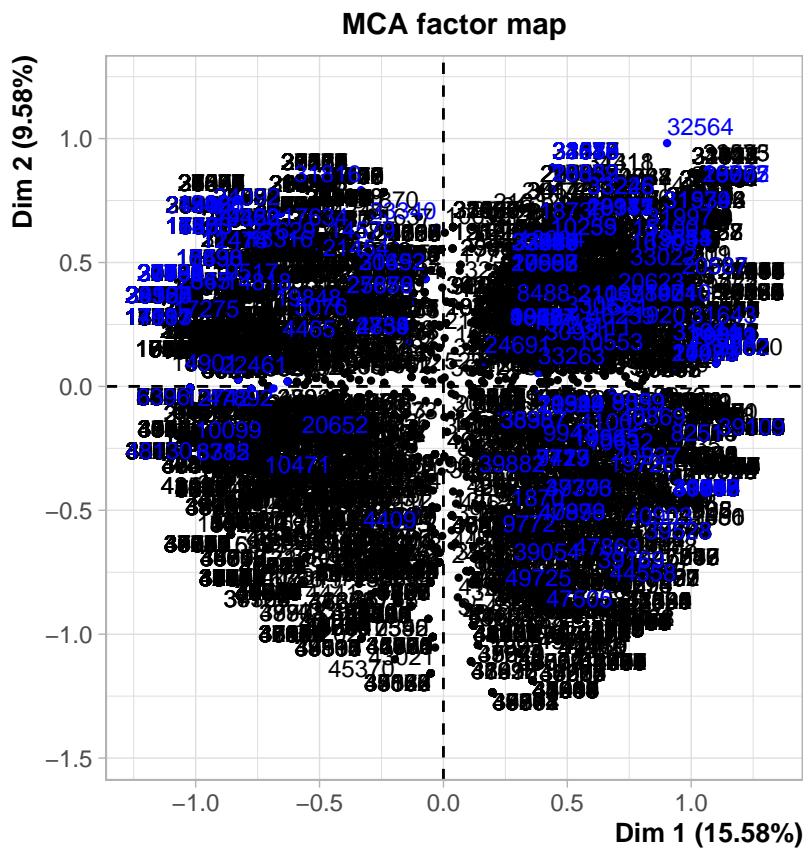
A continuación vamos a proceder a realizar el MCA. Para ello, en primer lugar vamos a seleccionar los individuos que hemos considerado como multivariant outliers para añadirlos como suplementarios. También se ha considerado la eliminación de la variable engineSize ya que si se incluye genera errores. También se han suprimido los factores model y year. Model se ha quitado ya que tiene demasiados valores y acaba no siendo explicativo. Year se ha eliminado porque guarda demasiada relación con f.age. Se añaden como variables suplementarias los factores f.price y Audi, además de la variable price, ya que son nuestros targets.

```
llvout<-which(df$mout=="YesMOut");length(llvout)
```

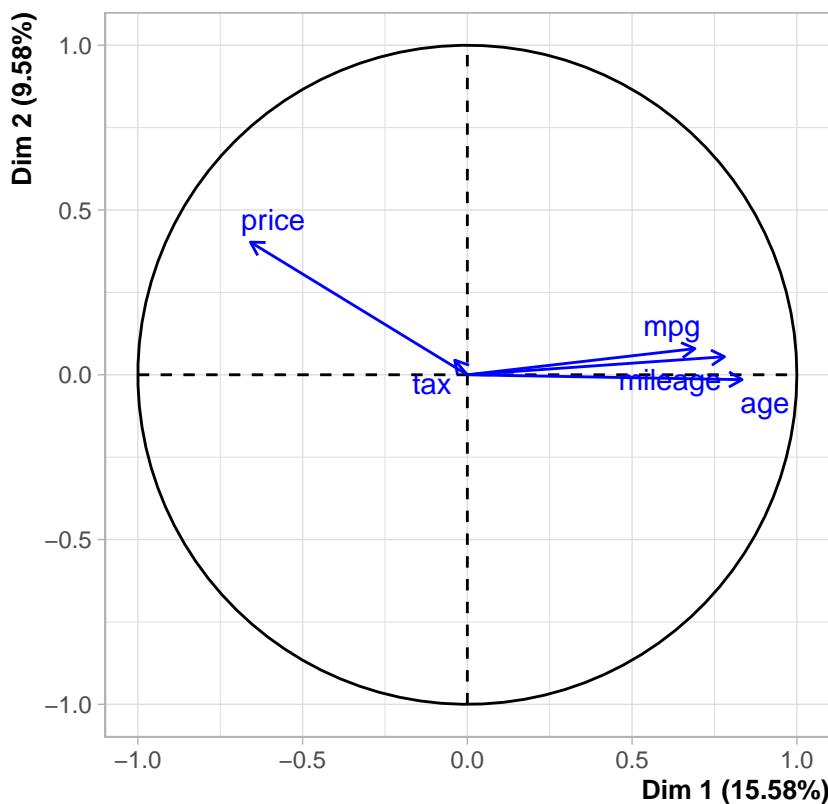
```
[1] 190
```

```
res.mca<-MCA(df[,c("f.price","Audi",vars_cat[c(3:4,6,8:11)],"price", vars_num) ], quali.sup=c(1,2),quant=
```





Supplementary quantitative variables



Vamos a aplicar el criterio de Kaiser para determinar el número de dimensiones relevantes para continuar el análisis MCA. En este caso, como los eigenvalues no están normalizados, nos quedaremos con todas las dimensiones que tengan un eigenvalue mayor que la media de todos los eigenvalues.

```
length(which(res.mca$eig[,1] > mean(res.mca$eig[,1])))
```

```
[1] 7
```

A continuación podemos ver los eigenvalues de las dimensiones seleccionadas, los porcentajes de varianza que acumulan y las varianza que hay acumulada hasta esa dimensión. Podemos ver que para el caso de la dimensión 7, que es la que nos ha indicado el criterio de Kaiser, se ha aglomerado cerca de un 60% de la variabilidad.

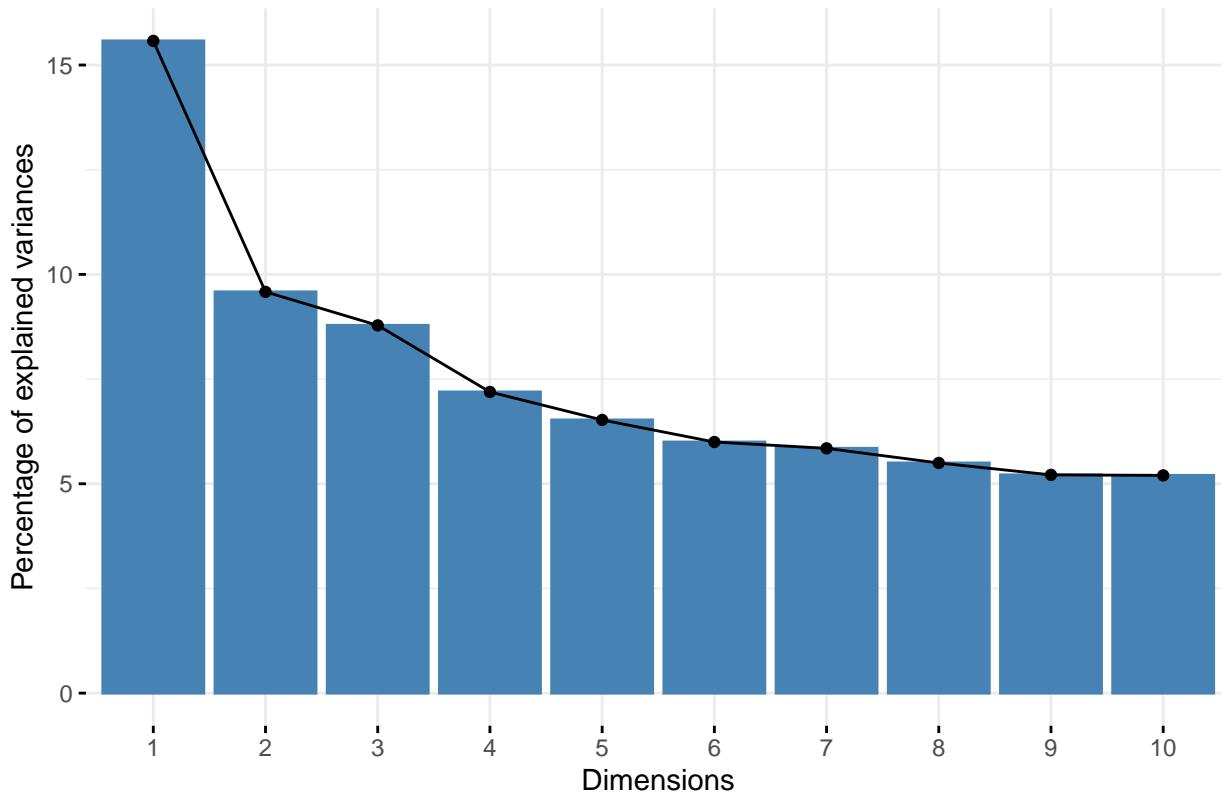
```
res.mca$eig[1:7,]
```

	eigenvalue	percentage of variance	cumulative percentage of variance
dim 1	0.4005255	15.575993	15.57599
dim 2	0.2464240	9.583154	25.15915
dim 3	0.2258525	8.783152	33.94230
dim 4	0.1849530	7.192617	41.13492
dim 5	0.1677634	6.524133	47.65905
dim 6	0.1542294	5.997809	53.65686
dim 7	0.1503358	5.846392	59.50325

En el siguiente gráfico podemos ver de una manera más visual la variabilidad acumulada para cada dimensión:

```
fviz_eig(res.mca)
```

Scree plot



11.1 Análisis segun las variables

11.1.1 Factores

Si analizamos el estadístico cos2, podemos ver la correlación que existe entre los factores y las componentes que se han creado. Podemos ver que para la Dim 1, el valor más relevante es f.age-[1,2] o f.gae-[+4] y para la segunda dimensión estaría f.Trans-Manual.

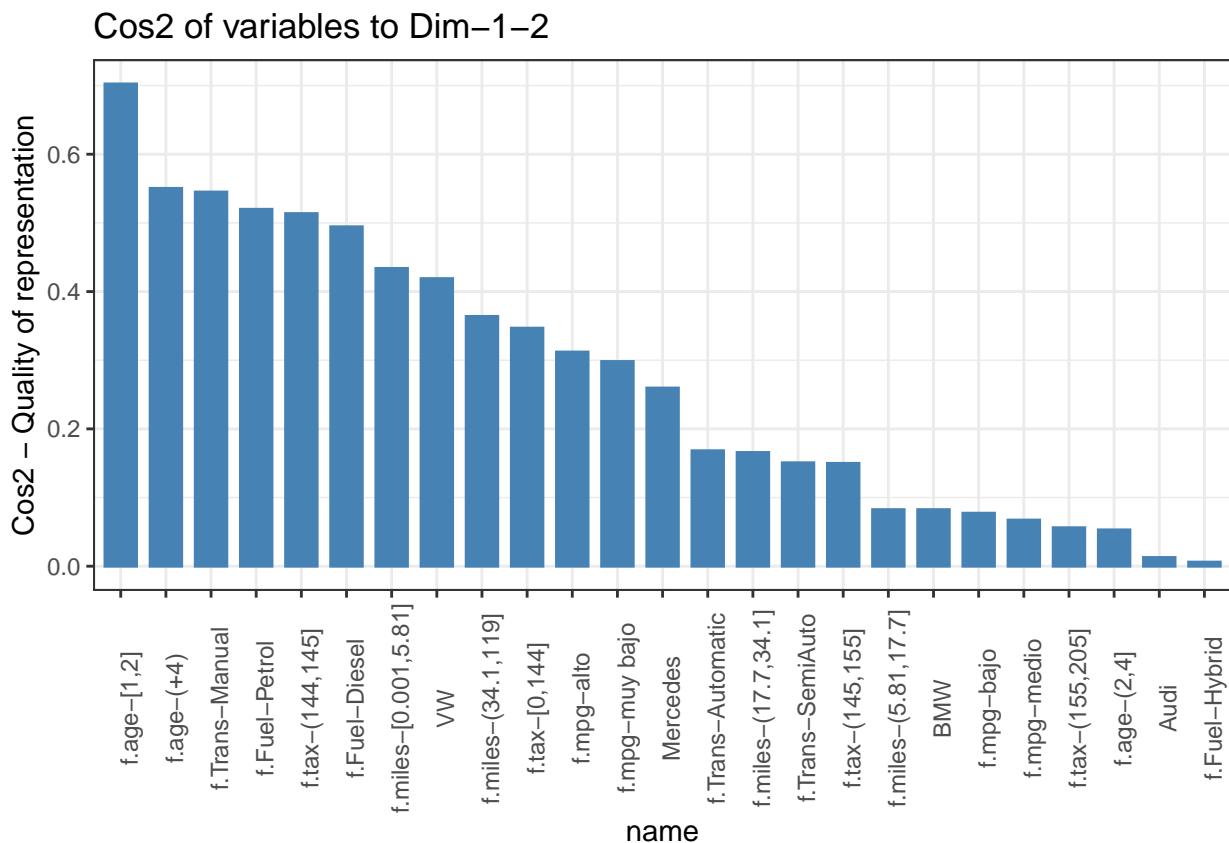
```
res.mca$var$cos2
```

	Dim 1	Dim 2	Dim 3	Dim 4
f.Trans-Manual	8.261124e-02	0.4624106064	0.0002621037	0.1020145464
f.Trans-SemiAuto	5.695219e-02	0.0937358980	0.0199128764	0.0442618988
f.Trans-Automatic	2.477682e-03	0.1657299241	0.0196348484	0.0136130127
f.Fuel-Diesel	1.201912e-01	0.3741907847	0.0268911121	0.0507058445
f.Fuel-Petrol	1.283272e-01	0.3915367523	0.0306853316	0.0577016097
f.Fuel-Hybrid	2.580690e-03	0.0034230154	0.0027164737	0.0048931975
Audi	2.166616e-04	0.0124966762	0.0673263421	0.0495163511
BMW	1.509009e-06	0.0824552664	0.0273593439	0.0104436890
Mercedes	2.102729e-05	0.2595383944	0.1295793160	0.0174250598
VW	9.568609e-05	0.4188159327	0.0010978790	0.0543093671
f.miles-[0.001,5.81]	4.263376e-01	0.0073760520	0.0303611859	0.1900065377
f.miles-(5.81,17.7]	8.168782e-02	0.0007749767	0.1097464322	0.0270074584
f.miles-(17.7,34.1]	1.310314e-01	0.0346545771	0.0763923390	0.2176306688
f.miles-(34.1,119]	3.461070e-01	0.0176855898	0.2009534582	0.0420491270
f.mpg-muy bajo	2.915478e-01	0.0066357814	0.1060344474	0.0534827270
f.mpg-bajo	3.353720e-02	0.0436873144	0.0410525687	0.0079853417
f.mpg-medio	5.671235e-02	0.0104919444	0.0776513572	0.0008043396
f.mpg-alto	2.544439e-01	0.0575120268	0.0693830286	0.0914884197
f.tax-[0,144]	3.397496e-01	0.0069776696	0.0407745199	0.0297628771
f.tax-(144,145]	5.020387e-01	0.0115896017	0.0777783146	0.0023839829
f.tax-(145,155]	3.971364e-02	0.1100897232	0.0112999773	0.0178834573
f.tax-(155,205]	2.965216e-02	0.0264069113	0.4049711247	0.2733933784
f.age-[1,2]	6.851524e-01	0.0172383517	0.0059742240	0.1498315612

f.age-(2,4]	1.301561e-02	0.0399291681	0.3157426471	0.2424994343
f.age-(+4)	5.473299e-01	0.0029535020	0.2040717746	0.0047454797
	Dim 5			
f.Trans-Manual	7.704956e-02			
f.Trans-SemiAuto	4.306830e-02			
f.Trans-Automatic	5.440223e-03			
f.Fuel-Diesel	4.069134e-02			
f.Fuel-Petrol	4.901571e-03			
f.Fuel-Hybrid	4.216071e-01			
Audi	2.688441e-02			
BMW	7.308836e-03			
Mercedes	3.237392e-04			
VW	2.692662e-03			
f.miles-[0.001,5.81]	5.667813e-04			
f.miles-(5.81,17.7]	3.490506e-03			
f.miles-(17.7,34.1]	1.911962e-04			
f.miles-(34.1,119]	2.578156e-03			
f.mpg-muy bajo	2.820517e-02			
f.mpg-bajo	1.356807e-01			
f.mpg-medio	3.104087e-01			
f.mpg-alto	1.248795e-01			
f.tax-[0,144]	4.676081e-03			
f.tax-(144,145]	5.505679e-02			
f.tax-(145,155]	1.678324e-01			
f.tax-(155,205]	4.353105e-03			
f.age-[1,2]	9.057793e-05			
f.age-(2,4]	2.548606e-03			
f.age-(+4)	3.299650e-03			

En el siguiente gráfico se pueden apreciar los valores de cos2 de un modo más visual:

```
fviz_cos2(res.mca, choice = "var", axes = 1:2)+theme_bw()+theme(axis.text.x = element_text(angle=90))
```



Por último, si echamos un vistazo a eta2, podemos ver la relevancia que tienen los factores, no solo los valores que estos toman.

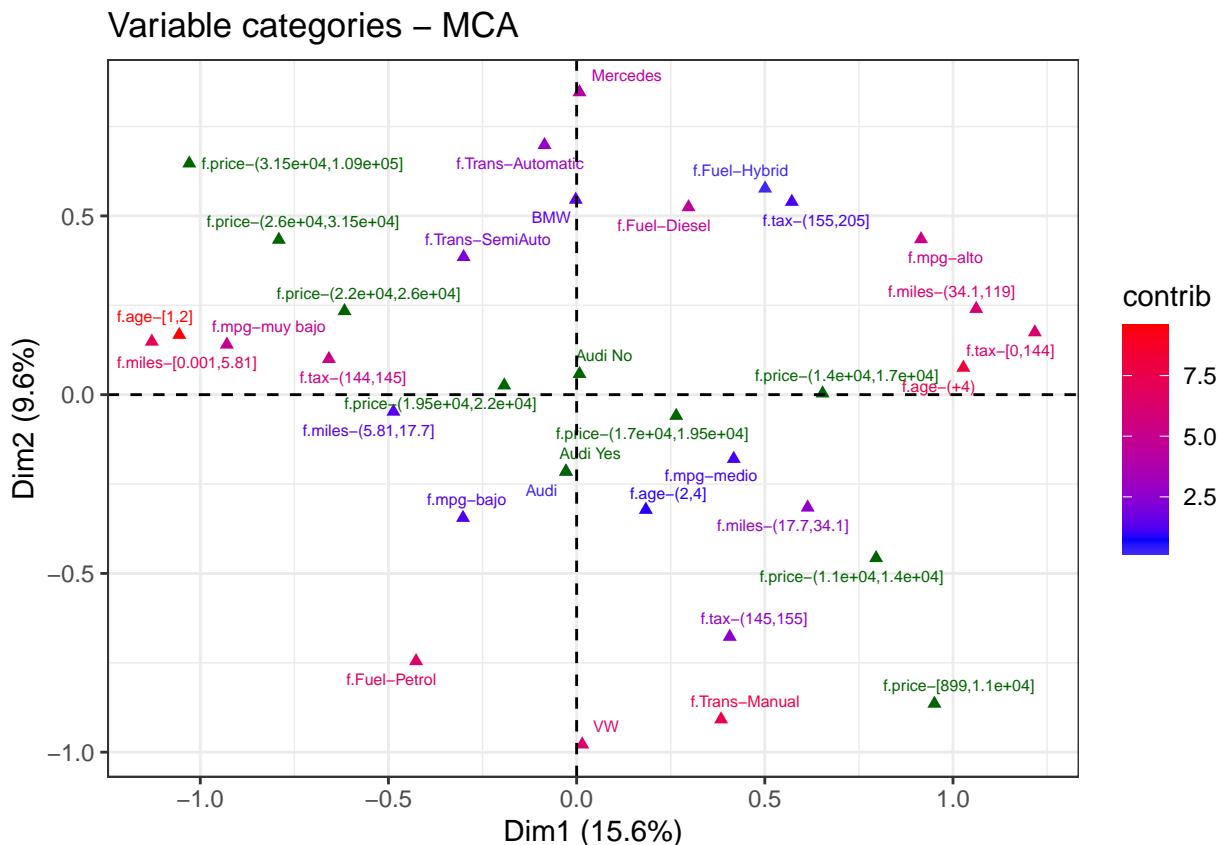
```
res.mca$var$eta2
```

	Dim 1	Dim 2	Dim 3	Dim 4
transmission	0.0896876857	0.47743313	0.02702704	0.10265356
fuelType	0.1287416604	0.39156403	0.03207911	0.06016866
manufacturer	0.0002538023	0.55621467	0.17030269	0.09773981
f.miles	0.7422126065	0.04534731	0.31478954	0.35610268
f.mpg	0.4804106769	0.08891975	0.22107931	0.11659461
f.tax	0.5681491694	0.12403696	0.44960417	0.29039675
f.age	0.7942231433	0.04145192	0.36608541	0.27101506
	sum.res.mca\$var\$contrib/100			
transmission		0.079825885		
fuelType		0.437427821		
manufacturer		0.029005336		
f.miles		0.005134763		
f.mpg		0.450235563		
f.tax		0.168645131		
f.age		0.004069529		

Podemos ver que para la primera dimensión, los factores más relevantes son f.age y f.miles, mientras que para la segunda serían manufacturer y transmission.

En el siguiente gráfico, podemos ver la contribución que tienen las variables para las dos primeras dimensiones generadas por el MCA, y que recordemos, acumulan cerca del 25% de la variabilidad.

```
fviz_mca_var(res.mca, col.var="contrib", repel=TRUE, labelsize = 2) +
  scale_color_gradient2(low="green", mid="blue",
  high="red", midpoint=0.75)+theme_bw()
```



11.1.2 Variables numéricas

Vamos a echar un vistazo a las variables numéricas. En la siguiente salida podemos observar la correlación que existe entre las variables numéricas originales y las componentes que se han obtenido con el MCA.

```
res.mca$quanti
```

```
$coord
      Dim 1      Dim 2      Dim 3      Dim 4      Dim 5
price -0.65819012  0.40252281 -0.01679887  0.11375357  0.08822518
mileage  0.78030181  0.05479618  0.29979135  0.04691816  0.03051676
tax     -0.03820536  0.04436195  0.51931286  0.39880873 -0.01136986
mpg      0.69063947  0.07941426 -0.40390622 -0.32762453 -0.14828726
age      0.83374388 -0.01487821  0.26999425  0.14039688  0.02986808
```

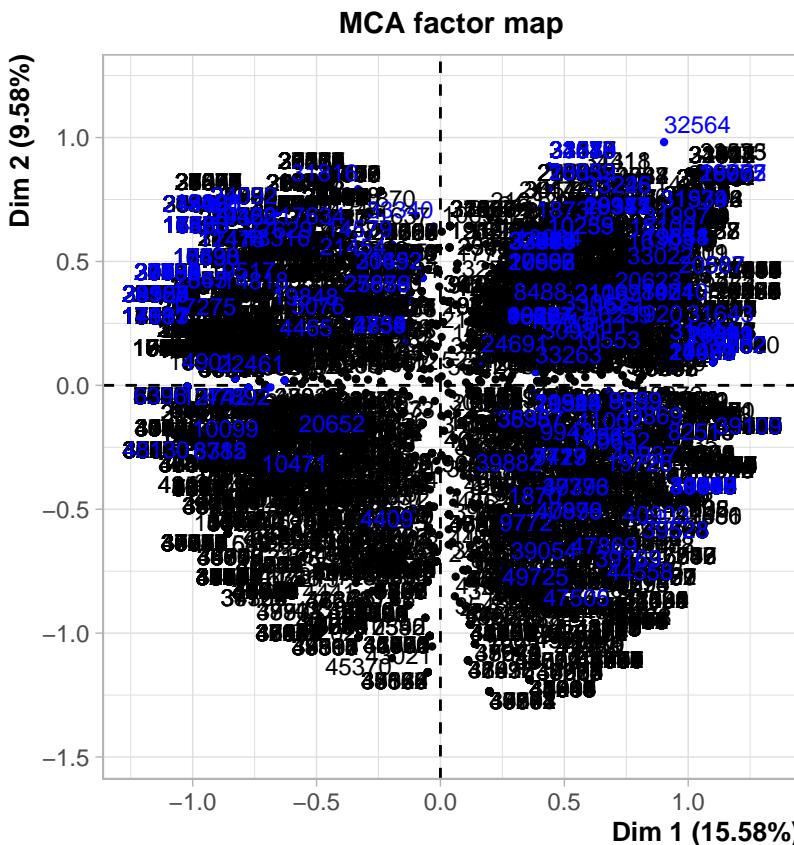
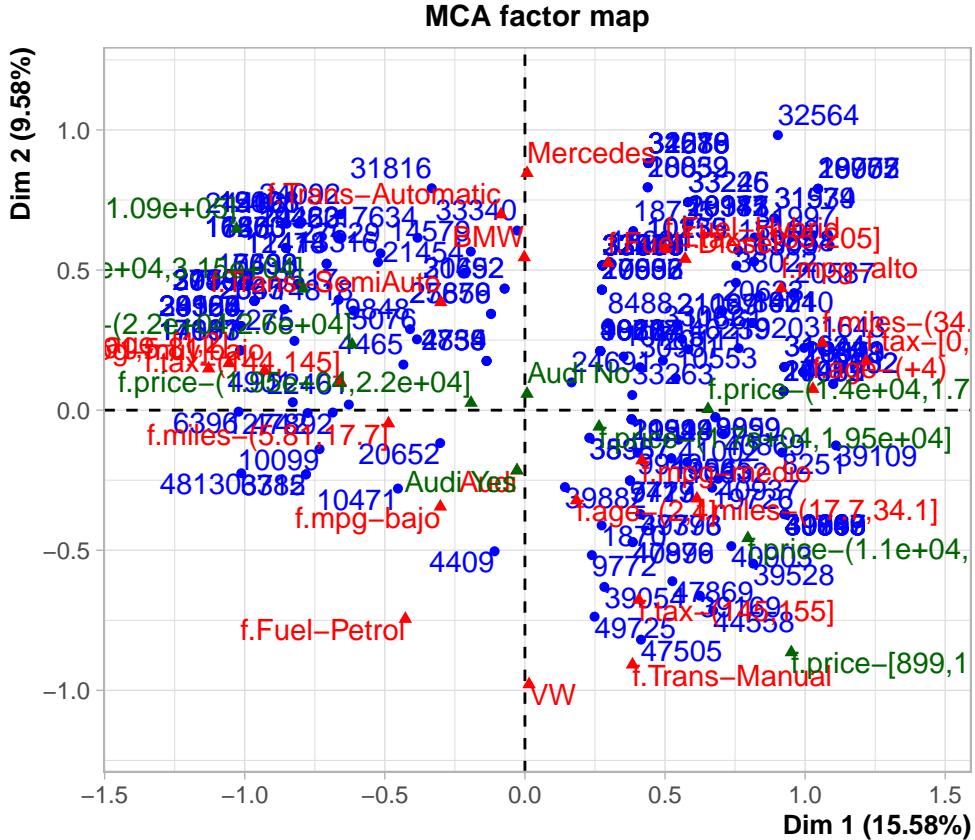
Podemos destacar la fuerte correlación de las variables age y mileage con la primera componente, y de price con la segunda.

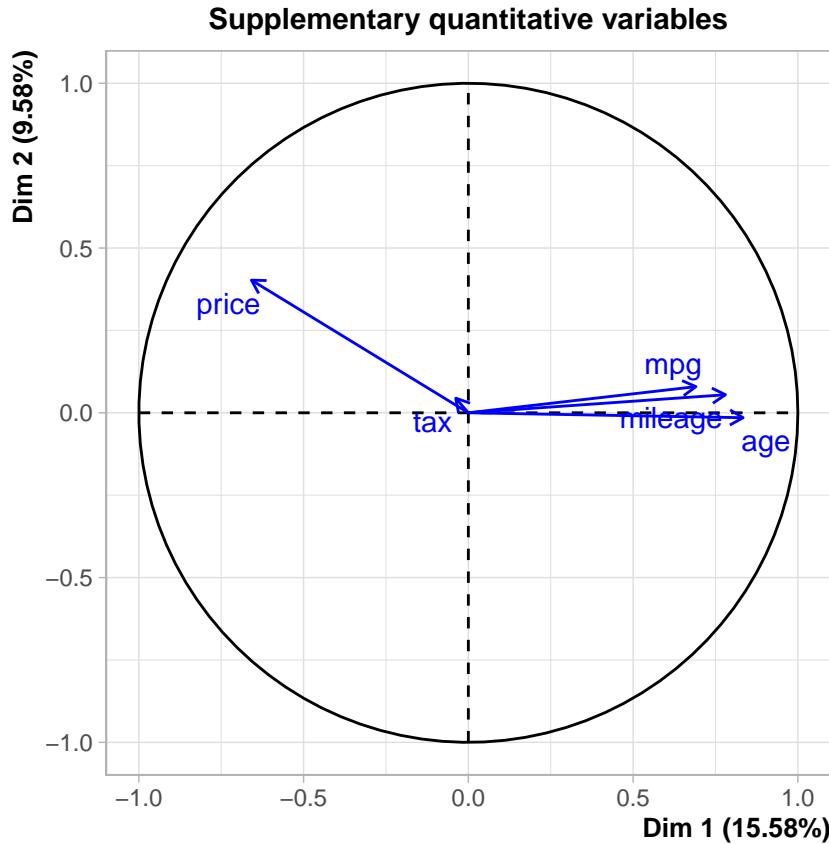
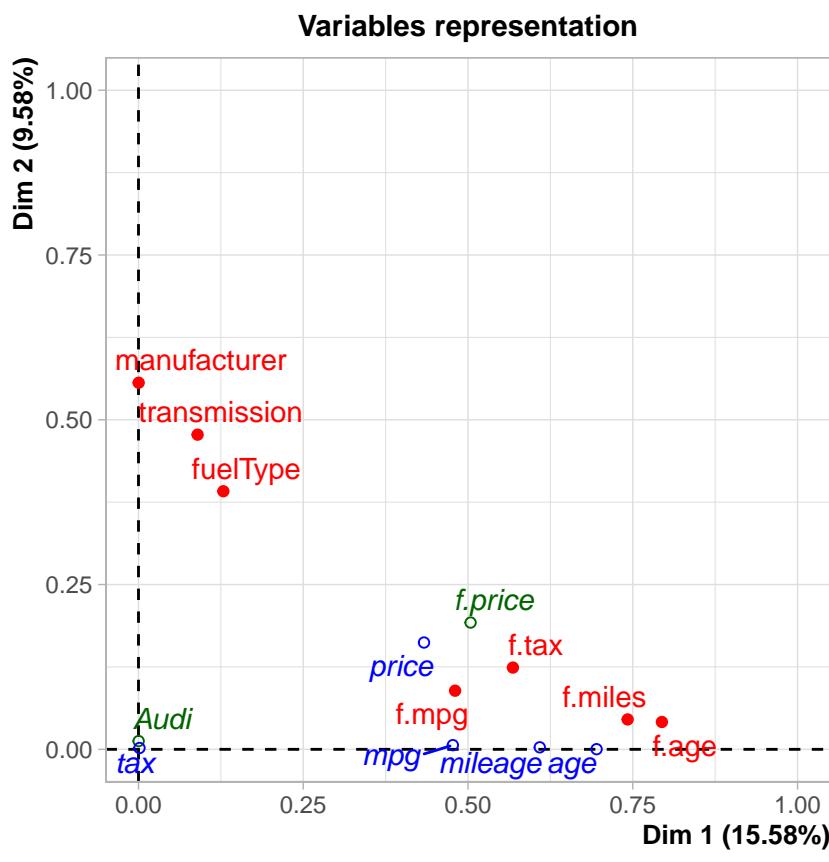
Como curiosidad, podemos mencionar que el hecho de que para la primera dimensión age y mileage aparezcan tan correlacionadas puede ser debido al hecho de sus factores derivados son los que más relevancia adquieran en esta primera dimensión.

12 Clustering Jerárquico desde MCA

Vamos a proceder a volver a realizar el clustering jerárquico pero esta vez lo vamos a lanzar desde el MCA en lugar del PCA. Vamos a añadir el número de componentes que hemos determinado durante el análisis MCA a partir del criterio de Kaiser.

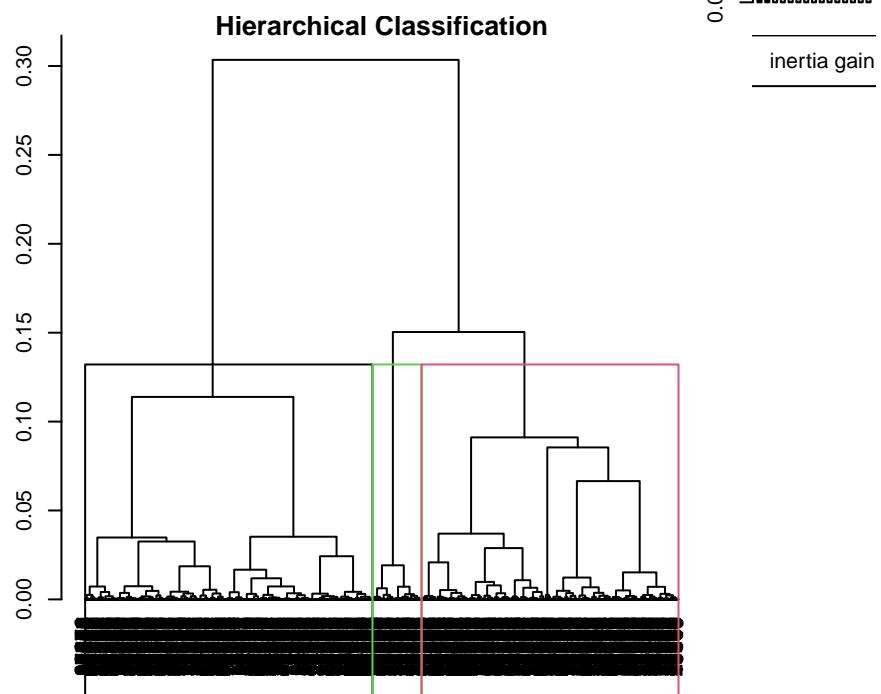
```
res.mca<-MCA(df[,c("f.price","Audi",vars_cat[c(3:4,6,8:11)],"price",vars_num) ],quali.sup=c(1,2),quanti...
```



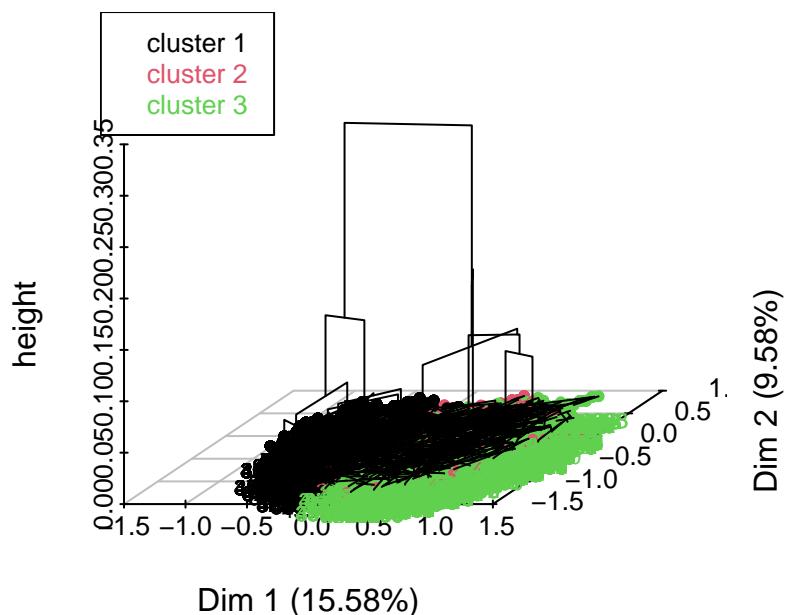


```
res.hcmc<-HCPC(res.mca, nb.clust=-1, order=TRUE)
```

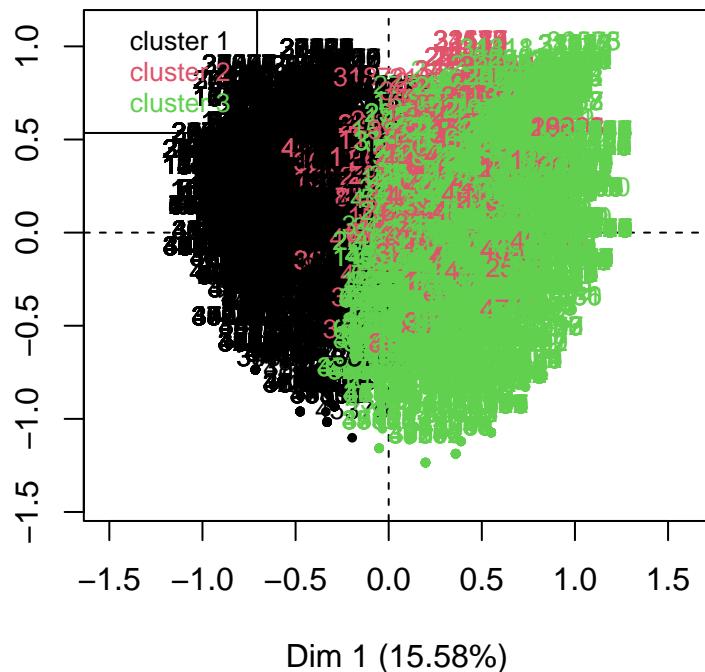
Hierarchical Clustering



Hierarchical clustering on the factor map



Factor map



Como podemos ver en la salida, en este caso, a partir del parámetro `nb.clust=-1` que como hemos mencionado anteriormente, automatizaba la selección del número óptimo de clusters, se han generado 4 clusters.

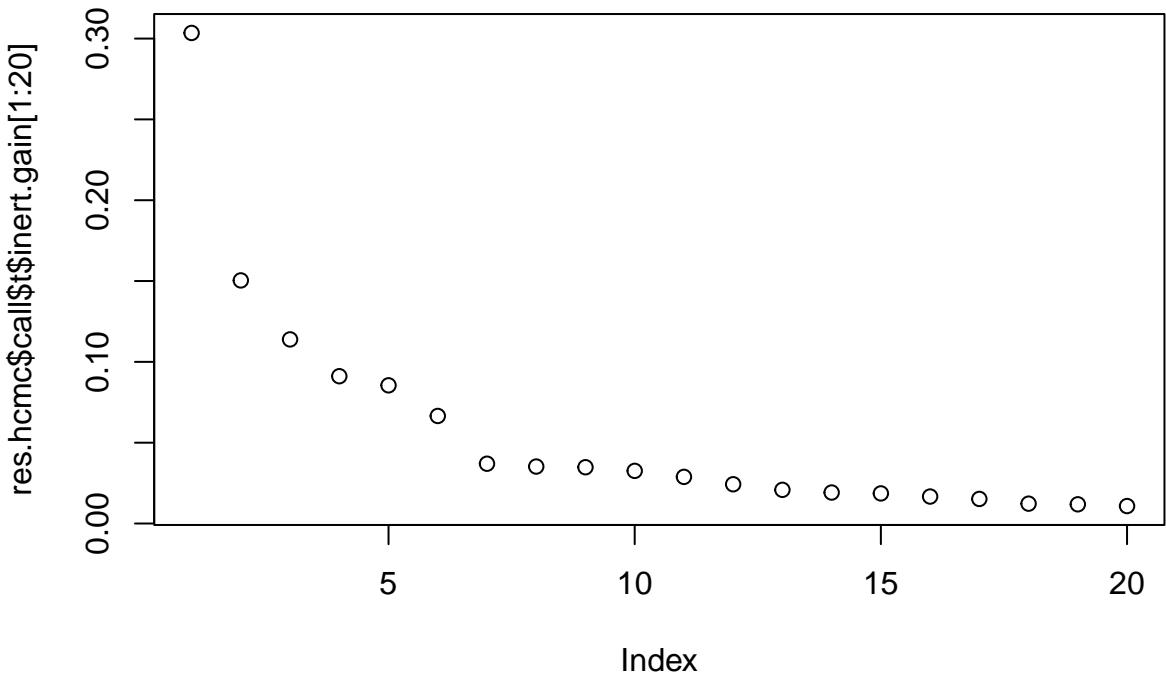
Sin embargo, si tenemos en cuenta el criterio de Kaiser, podemos ver que, en este caso, este número de componentes no son suficientes. Segundo Kaiser, deberíamos tomar 7 componentes.

```
length(which(res.hcmc$call$t$res$eig[, 1] > mean(res.hcmc$call$t$res$eig[, 1])))
```

```
[1] 7
```

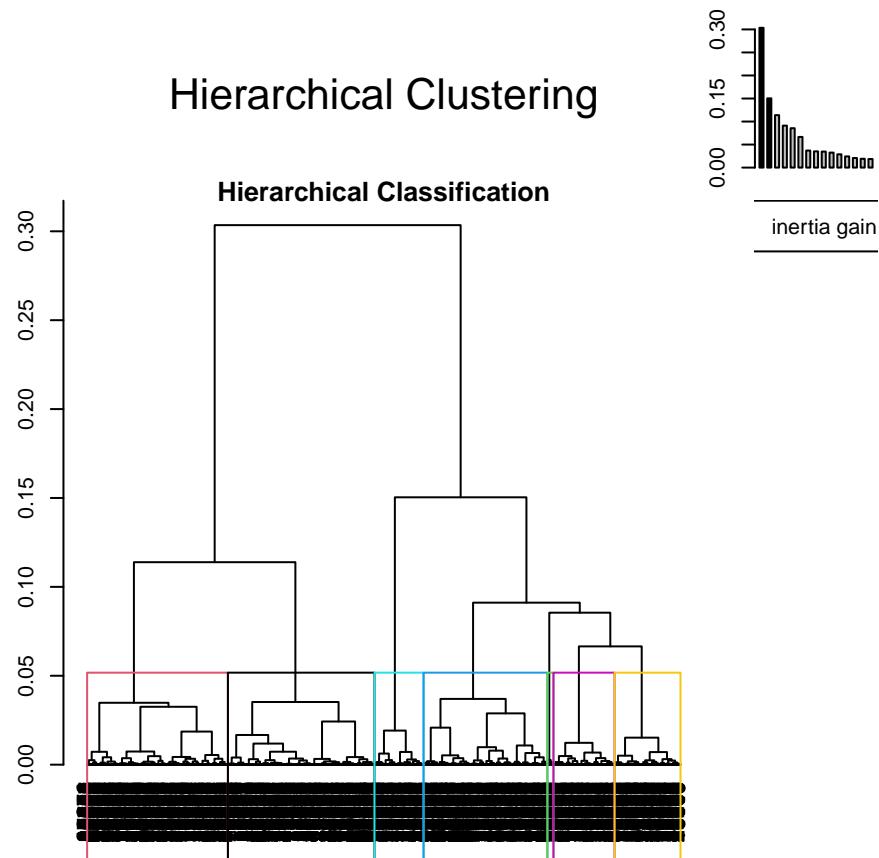
En el siguiente plot podemos ver la ganancia de inercia que se produce en nuestro modelo de clusterización jerárquica. Si aplicamos la regla de elbow, podemos determinar que el número óptimo de clusters sería alrededor de 6.

```
plot(res.hcmc$call$t$inert.gain[1:20])
```

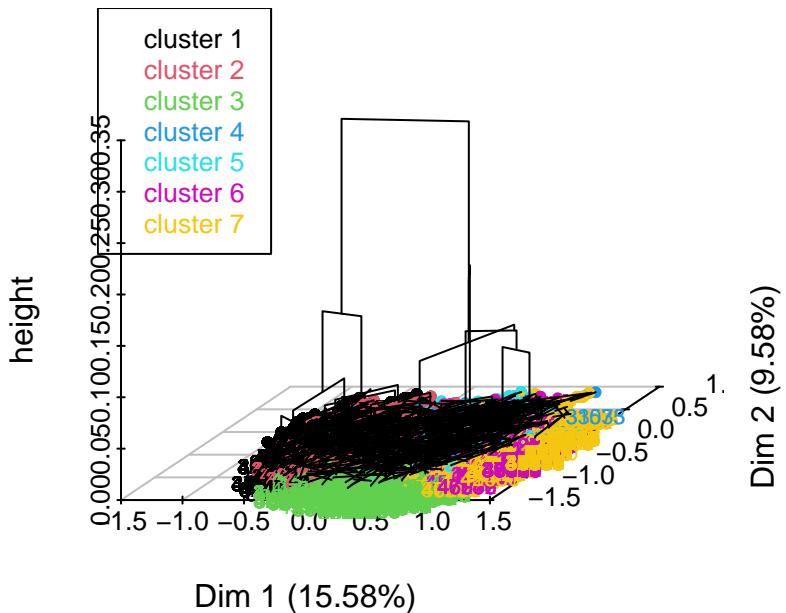


Por lo tanto, y resumiento, nb.clust nos da 4 componentes, elbow nos da 6 y Kaiser nos da 7. Nos quedaremos con el de Kaiser porque es el más específico.

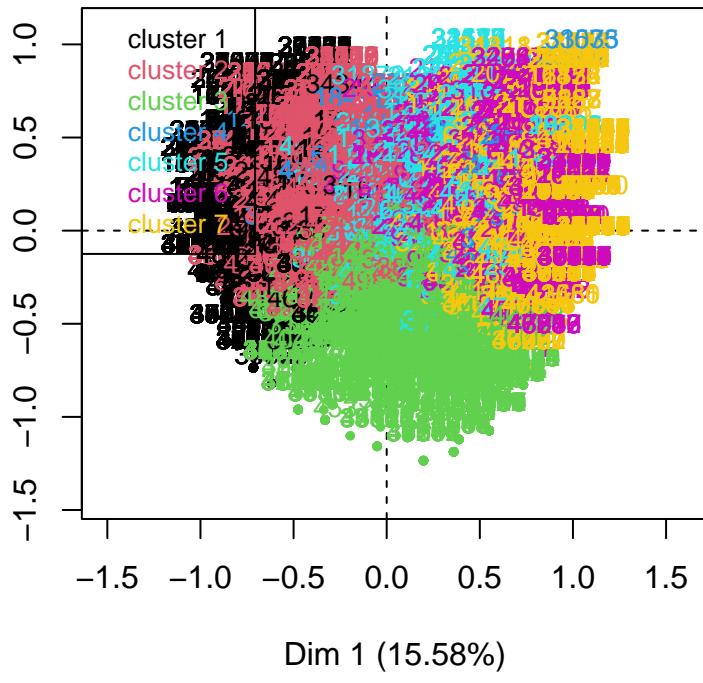
```
res.hcmc<-HCPC(res.mca, nb.clust=7, order=TRUE)
```



Hierarchical clustering on the factor map



Factor map



A continuación podemos ver inercia acumulada en las 7 primeras componentes.

```
(res.hcmc$call$t$within[1]-res.hcmc$call$t$within[1:7])/res.hcmc$call$t$within[1]
```

```
[1] 0.0000000 0.1983365 0.2966197 0.3710541 0.4306063 0.4864711 0.5299479
```

Crearemos una nueva variable en nuestro df para guardar en que cluster se han asignado los individuos.

```

df$claHCMC<-7
df[row.names(res.hcmc$data.clust),"claHCMC"]<-res.hcmc$data.clust$clust
df$claHCMC<-factor(df$claHCMC)
levels( df$claHCMC ) <- paste0( "f.claHCMC-",levels( df$claHCMC ) )
table(df$claHCMC)

```

```

f.claHCMC-1 f.claHCMC-2 f.claHCMC-3 f.claHCMC-4 f.claHCMC-5 f.claHCMC-6
 1235        890       863        49       399       681
f.claHCMC-7
 883

```

12.1 Análisis según las variables

12.1.1 Factores

En primer lugar, vamos a analizar el estadístico chi-squared para determinar que factores son las que más determinan las diferencias entre clusters:

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

	p.value	df
f.price	0.000000e+00	42
fuelType	0.000000e+00	12
f.miles	0.000000e+00	18
f.mpg	0.000000e+00	18
f.tax	0.000000e+00	18
f.age	0.000000e+00	12
transmission	1.201301e-266	12
manufacturer	4.805326e-246	18
Audi	2.307286e-17	6

Si profundizamos un poco más podemos ver caracterizar un poco los clusters según los valores de estos factores. <He omitido la salida porque ocupa mucho, pero podemos ver algunas de las conclusiones abajo>

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

Podemos ver, por ejemplo, que el cluster 1 esta compuesto por coches muy nuevos (f.age=f.age-[1,2] -> Mod/Cla 96.6631908) y con poco kilometraje (f.miles=f.miles-[0.001,5.81] -> Mod/Cla 69.9687174). Por el contrario, prácticamente no hay coches viejos (f.age=f.age-(+4) -> Mod/Cla 0.1042753) o con precios bajos (f.price=f.price-[899,1.1e+04] -> Mod/Cla 0.00000000).

Contrariamente, en el cluster 7, se acumulan los vehículos con más de 4 años, Diesel y con mpg elevados.

12.1.2 Variables numéricas

En los que se refiere a las variables cuantitativas:

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

	Eta2	P-value
price	0.4484417	0
mileage	0.6931149	0
tax	0.5605186	0
mpg	0.4377990	0
age	0.6974664	0

Podemos ver que la que mejor explica la separación entre clusters es age, mientras que price parece ser la menos explicativa.

En este apartado, cabe destacar que todas las variables tienen factores derivados que se han usado para generar el clustering. Sin embargo, hay que tener en cuenta que en el caso de nuestro target price, su discretización solo

se ha añadido como variable suplementaria, de modo que no ha tenido influencia en la clusterización, hecho que explica que aparezca como la menos representativa.

Si analizamos más en profundidad, podemos ver las distribuciones que toman cada una de las variables cualitativas dentro de los distintos clusters.

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

```
$'1'
      v.test Mean in category Overall mean sd in category    Overall sd
price   33.078691    28254.121457 20900.044491    8605.699644 9061.554246
tax     -3.579428     145.469095  146.518027     1.528215   11.944192
mpg    -28.011157     46.054080  53.429394     8.902400   10.731794
mileage -39.336128    3327.747368 21978.643932    3257.691577 19325.509377
age    -42.269949     1.789474   3.712352     0.436459   1.854142
      p.value
price   6.018978e-240
tax     3.443477e-04
mpg    1.188298e-172
mileage 0.000000e+00
age    0.000000e+00

$'2'
      v.test Mean in category Overall mean sd in category    Overall sd
price   18.264323    25908.766292 20900.044491    7962.6083774 9061.554246
tax     -2.175707     145.731565  146.518027     2.8908302   11.944192
mpg    -8.331449     50.723483  53.429394     10.4747207  10.731794
mileage -15.807977    12733.197753 21978.643932    7729.7882473 19325.509377
age    -18.702091     2.662921   3.712352     0.8785051   1.854142
      p.value
price   1.591754e-74
tax     2.957720e-02
mpg    7.985860e-17
mileage 2.741308e-56
age    4.760138e-78

$'3'
      v.test Mean in category Overall mean sd in category    Overall sd
age     6.722713     4.096756   3.712352     1.351009   1.854142
mpg    2.239478     54.170568  53.429394     6.453064   10.731794
tax    -3.370149     145.276642 146.518027     6.813252   11.944192
price  -22.329590    14660.049826 20900.044491    5504.784484 9061.554246
      p.value
age     1.783713e-11
mpg    2.512484e-02
tax     7.512765e-04
price  1.906689e-110

$'4'
      v.test Mean in category Overall mean sd in category    Overall sd
price  2.594924     24242.3878  20900.044     8003.578686 9061.55425
tax    -2.674306     141.9777   146.518     6.593425   11.94419
      p.value
price  0.009461175
tax    0.007488416

$'5'
      v.test Mean in category Overall mean sd in category    Overall sd
tax     50.241299    175.290179 146.518027     19.81048   11.944192
age    24.151822     5.859428   3.712352     1.41618   1.854142
mileage 21.101217    41530.732446 21978.643932    18737.31644 19325.509377
price  -2.903561     19638.541353 20900.044491    6917.12272 9061.554246
mpg    -15.598380     45.403258  53.429394     5.98443   10.731794
      p.value
```

```

tax      0.000000e+00
age     7.144614e-129
mileage 7.751453e-99
price   3.689451e-03
mpg    7.466468e-55

$'6'
      v.test Mean in category Overall mean sd in category Overall sd
mpg      26.593227      63.563020      53.429394      6.2785597      10.731794
age      12.532524      4.537445      3.712352      0.8849327      1.854142
mileage  6.344930 26332.552129 21978.643932 8218.8520713 19325.509377
price   -9.763454 17758.612335 20900.044491 4468.7431748 9061.554246
tax     -16.324624      139.594595      146.518027      7.4100994      11.944192
      p.value
mpg     8.130070e-156
age     4.955969e-36
mileage 2.225269e-10
price   1.615569e-22
tax     6.594602e-60

$'7'
      v.test Mean in category Overall mean sd in category Overall sd
mileage 42.35669      50749.277760 21978.643932 15358.688117 19325.509377
age      34.74918      5.976912      3.712352      1.223643      1.854142
mpg     27.12984      63.662700      53.429394      7.250020      10.731794
tax     -11.94813      141.502076      146.518027      8.915325      11.944192
price   -25.71663      12709.493506 20900.044491 3391.589999 9061.554246
      p.value
mileage 0.000000e+00
age     1.426177e-264
mpg     4.379502e-162
tax     6.639873e-33
price   7.618285e-146

```

Como se ha comentado con anterioridad según la descripción a partir de los factores, podemos ver que en el primer cluster price es más alto, mientras que mileage, age o mpg son signitivamente más bajos que en el resto del df. De modo que en este cluster aparecen los coches más nuevos y caros.

Para el caso del cluster 2, podemos ver como aparecen coches algo más caros que la media, pero sin llegar al nivel del primer cluster.

En el caso del cluster 3, podemos ver como price pasa a ser un factor no tan relevante en favor de age y mpg.

Asimismo, podemos ver como en el cluster 7, se acumulan vehiculos más viejos y con más kilometraje, pero también más baratos.

12.2 Análisis según las componentes del MCA

Vamos a proceder a analizar la clusterización a apartir de los ejes generados a partir del MCA.

En primer lugar, podemos ver la relevancia que han tenido las distintas componentes del MCA para generar la clusterización:

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

	Eta2	P-value
Dim.1	0.8175065	0.000000e+00
Dim.2	0.5001458	0.000000e+00
Dim.3	0.6708714	0.000000e+00
Dim.4	0.5519670	0.000000e+00
Dim.5	0.4406989	0.000000e+00
Dim.7	0.3124665	0.000000e+00
Dim.6	0.2608531	1.12528e-310

Podemos ver como la dimensión 1 ha tenido la mayor relevancia, mientras que la dimensión 6 ha sido la menos determinante.

Por último, podemos ver algunas estadísticas sobre como se distribuyen las coordenadas del MCA para cada individuo en los diferentes clusters:

```
res.hcmc$desc.axes$quanti
```

\$'1'

	v.test	Mean in category	Overall mean	sd in category	Overall	sd
Dim.6	20.066739	0.19334704	-3.068549e-17	0.3521307	0.3927205	
Dim.3	13.084511	0.15256224	-4.262551e-17	0.2187523	0.4752394	
Dim.7	12.518080	0.11908199	3.736760e-17	0.3116664	0.3877316	
Dim.2	6.831405	0.08320102	-2.913470e-17	0.4121689	0.4964111	
Dim.4	-26.731055	-0.28204861	-2.700542e-17	0.2559375	0.4300616	
Dim.1	-47.790199	-0.74204646	1.489684e-16	0.1983562	0.6328709	
	p.value					
Dim.6	1.441576e-89					
Dim.3	4.037517e-39					
Dim.7	5.945604e-36					
Dim.2	8.408698e-12					
Dim.4	2.050744e-157					
Dim.1	0.000000e+00					

\$'2'

	v.test	Mean in category	Overall mean	sd in category	Overall	sd
Dim.4	13.36443	0.17394093	-2.700542e-17	0.2997370	0.4300616	
Dim.2	12.28193	0.18451377	-2.913470e-17	0.3284702	0.4964111	
Dim.5	-2.68283	-0.03325542	-3.352597e-18	0.2872620	0.4095893	
Dim.1	-19.48194	-0.37313664	1.489684e-16	0.2940334	0.6328709	
Dim.7	-22.49625	-0.26397450	3.736760e-17	0.3099956	0.3877316	
Dim.3	-23.05604	-0.33160244	-4.262551e-17	0.2919285	0.4752394	
Dim.6	-25.30627	-0.30076846	-3.068549e-17	0.3200777	0.3927205	
	p.value					
Dim.4	9.758007e-41					
Dim.2	1.132615e-34					
Dim.5	7.300205e-03					
Dim.1	1.562530e-84					
Dim.7	4.516769e-112					
Dim.3	1.279475e-117					
Dim.6	2.725163e-141					

\$'3'

	v.test	Mean in category	Overall mean	sd in category	Overall	sd
Dim.4	10.321439	0.13689000	-2.700542e-17	0.3523504	0.4300616	
Dim.1	7.261789	0.14172926	1.489684e-16	0.3553174	0.6328709	
Dim.5	2.825623	0.03569140	-3.352597e-18	0.3723042	0.4095893	
Dim.7	2.192065	0.02621111	3.736760e-17	0.3515096	0.3877316	
Dim.6	-5.647159	-0.06839342	-3.068549e-17	0.3499403	0.3927205	
Dim.3	-8.914294	-0.13064721	-4.262551e-17	0.3300221	0.4752394	
Dim.2	-48.359175	-0.74032281	-2.913470e-17	0.2518262	0.4964111	
	p.value					
Dim.4	5.637213e-25					
Dim.1	3.820028e-13					
Dim.5	4.718884e-03					
Dim.7	2.837484e-02					
Dim.6	1.631211e-08					
Dim.3	4.909339e-19					
Dim.2	0.000000e+00					

\$'4'

	v.test	Mean in category	Overall mean	sd in category	Overall	sd
Dim.5	45.027865	2.6215230	-3.352597e-18	0.3356363	0.4095893	
Dim.7	21.895784	1.2067446	3.736760e-17	0.4417437	0.3877316	

Dim.2	4.057250	0.2862839	-2.913470e-17	0.2693861	0.4964111
Dim.1	3.522859	0.3169086	1.489684e-16	0.4732424	0.6328709
Dim.6	-2.004802	-0.1119125	-3.068549e-17	0.3227711	0.3927205
Dim.3	-3.614349	-0.2441553	-4.262551e-17	0.3704298	0.4752394
Dim.4	-4.850916	-0.2965365	-2.700542e-17	0.2904452	0.4300616
p.value					
Dim.5	0.000000e+00				
Dim.7	2.849591e-106				
Dim.2	4.965383e-05				
Dim.1	4.269180e-04				
Dim.6	4.498420e-02				
Dim.3	3.011027e-04				
Dim.4	1.228925e-06				
\$'5'					
	v.test	Mean in category	Overall mean	sd in category	Overall sd
Dim.3	44.267018	1.00866599	-4.262551e-17	0.2980871	0.4752394
Dim.4	35.553505	0.73310777	-2.700542e-17	0.3160583	0.4300616
Dim.1	11.978482	0.36347253	1.489684e-16	0.2244045	0.6328709
Dim.2	11.586730	0.27577638	-2.913470e-17	0.3253128	0.4964111
Dim.7	2.396412	0.04454996	3.736760e-17	0.3629378	0.3877316
Dim.5	-4.685592	-0.09201694	-3.352597e-18	0.2104945	0.4095893
Dim.6	-9.866681	-0.18578445	-3.068549e-17	0.4375580	0.3927205
p.value					
Dim.3	0.000000e+00				
Dim.4	7.334651e-277				
Dim.1	4.606804e-33				
Dim.2	4.811505e-31				
Dim.7	1.655648e-02				
Dim.5	2.791524e-06				
Dim.6	5.805361e-23				
\$'6'					
	v.test	Mean in category	Overall mean	sd in category	Overall sd
Dim.1	26.227928	0.5893870	1.489684e-16	0.2881696	0.6328709
Dim.6	19.067901	0.2658936	-3.068549e-17	0.3045646	0.3927205
Dim.7	16.339981	0.2249595	3.736760e-17	0.2901502	0.3877316
Dim.2	12.366406	0.2179749	-2.913470e-17	0.3381695	0.4964111
Dim.4	8.261521	0.1261573	-2.700542e-17	0.2799375	0.4300616
Dim.5	-9.498852	-0.1381469	-3.352597e-18	0.3163747	0.4095893
Dim.3	-27.603951	-0.4658061	-4.262551e-17	0.2770298	0.4752394
p.value					
Dim.1	1.276580e-151				
Dim.6	4.666603e-81				
Dim.7	5.126880e-60				
Dim.2	3.971996e-35				
Dim.4	1.438279e-16				
Dim.5	2.122171e-21				
Dim.3	9.975901e-168				
\$'7'					
	v.test	Mean in category	Overall mean	sd in category	Overall sd
Dim.1	36.605798	0.8142568	1.489684e-16	0.2062161	0.6328709
Dim.3	12.628344	0.2109382	-4.262551e-17	0.2220498	0.4752394
Dim.2	8.222847	0.1434697	-2.913470e-17	0.3952433	0.4964111
Dim.7	-17.455590	-0.2378823	3.736760e-17	0.3076561	0.3877316
Dim.4	-27.541962	-0.4163150	-2.700542e-17	0.2189583	0.4300616
p.value					
Dim.1	2.312774e-293				
Dim.3	1.473460e-36				
Dim.2	1.987289e-16				
Dim.7	3.121357e-68				
Dim.4	5.523831e-167				

12.3 Análisis según individuos

En las siguientes salidas podemos ver los parámetros del primer cluster.

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

Cluster: 1
40410 44910 45035 45038 45440
0.2560676 0.2560676 0.2560676 0.2560676 0.2560676

Cluster: 2
48378 32842 38576 45057 45921
0.3670303 0.3739795 0.4059449 0.4059449 0.4059449

Cluster: 3
36534 37522 41873 42151 42468
0.3719036 0.3719036 0.3719036 0.3719036 0.3719036

Cluster: 4
16294 16900 16987 17777 16491
0.4902792 0.4902792 0.4902792 0.5825205 0.5944956

Cluster: 5
30937 824 847 2099 2443
0.3384156 0.4138110 0.4138110 0.4138110 0.4138110

Cluster: 6
39164 36867 38915 37758 48919
0.3635256 0.3802465 0.3802465 0.4459803 0.4459803

Cluster: 7
49669 49680 38870 48823 39319
0.3870723 0.3870723 0.4041328 0.4041328 0.4059869

Curiosamente, entre los parámetros del primer cluster, aparecen vehículos VW semi-automáticos Diesel. Todos tienen entre 1 y 2 años y consumos y kilometrajes muy bajos.

```
summary(df[c("40410", "44910", "45035", "45038", "45440")])
```

	model	year	price	transmission	
VW-	Tiguan:4	2019 :5	Min. :25500	f.Trans-Manual :0	
VW-	Passat:1	2001 :0	1st Qu.:27998	f.Trans-SemiAuto :5	
Audi-	A1 :0	2002 :0	Median :28199	f.Trans-Automatic:0	
Audi-	A3 :0	2003 :0	Mean :28339		
Audi-	A4 :0	2004 :0	3rd Qu.:29999		
Audi-	A5 :0	2005 :0	Max. :29999		
(Other)	:0	(Other):0			
	mileage	fuelType	tax	mpg	engineSize
Min.	: 1	f.Fuel-Diesel:5	Min. :145	Min. :39.8	2 :5
1st Qu.	: 669	f.Fuel-Petrol:0	1st Qu.:145	1st Qu.:40.4	1 :0
Median	:3000	f.Fuel-Hybrid:0	Median :145	Median :44.1	1.2 :0
Mean	:2737		Mean :145	Mean :43.1	1.3 :0
3rd Qu.	:4289		3rd Qu.:145	3rd Qu.:45.6	1.4 :0
Max.	:5726		Max. :145	Max. :45.6	1.5 :0
				(Other):0	
	manufacturer	age	outs	f.miles	
Audi	:0	Min. :2	Min. :0	f.miles-[0.001,5.81]:5	
BMW	:0	1st Qu.:2	1st Qu.:0	f.miles-(5.81,17.7] :0	
Mercedes	:0	Median :2	Median :0	f.miles-(17.7,34.1] :0	
VW	:5	Mean :2	Mean :0	f.miles-(34.1,119] :0	
		3rd Qu.:2	3rd Qu.:0		
		Max. :2	Max. :0		

```

          f.tax           f.mpg           f.age
f.tax-[0,144] :0   f.mpg-muy bajo:5   f.age-[1,2]:5
f.tax-(144,145]:5 f.mpg-bajo     :0   f.age-(2,4):0
f.tax-(145,155]:0 f.mpg-medio    :0   f.age-(+4) :0
f.tax-(155,205]:0 f.mpg-alto     :0

          f.price        Audi      mout  claKMPCA
f.price-(2.6e+04,3.15e+04]:4   Audi No :5   NoMOut :5   1:5
f.price-(2.2e+04,2.6e+04] :1   Audi Yes:0  YesMOut:0  2:0
f.price-[899,1.1e+04]       :0
f.price-(1.1e+04,1.4e+04] :0
f.price-(1.4e+04,1.7e+04] :0
f.price-(1.7e+04,1.95e+04] :0
(Other)                  :0

          claHCMC
f.claHCMC-1:5
f.claHCMC-2:0
f.claHCMC-3:0
f.claHCMC-4:0
f.claHCMC-5:0
f.claHCMC-6:0
f.claHCMC-7:0

```

En la siguiente salida podemos ver los individuos característicos de cada cluster:

```

res.hcmc$desc.ind$dist

Cluster: 1
    7854     10322     10323     10482      853
1.576288 1.576288 1.576288 1.576288 1.567202
-----
Cluster: 2
    10889     11890     16143     16608     17218
1.545064 1.545064 1.545064 1.545064 1.545064
-----
Cluster: 3
    35209     37488     37509     38588     38865
1.614295 1.614295 1.614295 1.614295 1.614295
-----
Cluster: 4
    18820     19986     20160     16612     11341
3.432083 3.432083 3.432083 3.312572 3.308374
-----
Cluster: 5
    10003     10375     7910      8292      9145
2.021428 2.021428 1.968406 1.968406 1.968406
-----
Cluster: 6
    32575     33183     33568     33587     33811
1.54681 1.54681 1.54681 1.54681 1.54681
-----
Cluster: 7
    12394     15223     31066     31717     31785
1.587369 1.587369 1.516852 1.516852 1.516852

```

Si nos fijamos, en este caso, los infivi

```
summary(df[c("5388", "7854", "10322", "10323", "10482"),])
```

```

model      year     price           transmission
Audi- A1:1  2019   :4   Min.   :21500   f.Trans-Manual  :0
Audi- A3:1  2020   :1   1st Qu.:23500   f.Trans-SemiAuto :0
Audi- A5:1  2001   :0   Median  :30000   f.Trans-Automatic:5
Audi- Q3:1  2002   :0   Mean    :28804
Audi- TT:1   2003   :0   3rd Qu.:32000
Audi- A4:0  2004   :0   Max.    :37020
(Other)  :0   (Other):0

mileage      fuelType     tax       mpg      engineSize
Min.   : 2500   f.Fuel-Diesel:0   Min.   :145   Min.   :31.70  2      :3
1st Qu.: 2796   f.Fuel-Petrol:5   1st Qu.:150   1st Qu.:38.70  1.5    :2
Median  : 2893   f.Fuel-Hybrid:0   Median  :150   Median  :40.90  1      :0
Mean    : 6299                    Mean    :149   Mean    :39.66  1.2    :0
3rd Qu.: 3215                    3rd Qu.:150   3rd Qu.:42.20  1.3    :0
Max.    :20091                    Max.    :150   Max.    :44.80  1.4    :0
(Other) :0

manufacturer   age      outs          f.miles
Audi      :5   Min.   :1.0   Min.   :0   f.miles-[0.001,5.81]:4
BMW       :0   1st Qu.:2.0   1st Qu.:0   f.miles-(5.81,17.7] :0
Mercedes:0   Median :2.0   Median :0   f.miles-(17.7,34.1] :1
VW        :0   Mean    :1.8   Mean    :0   f.miles-(34.1,119] :0
                           3rd Qu.:2.0   3rd Qu.:0
                           Max.   :2.0   Max.   :0

f.tax      f.mpg      f.age
f.tax-[0,144] :0   f.mpg-muy bajo:5   f.age-[1,2]:5
f.tax-(144,145]:1   f.mpg-bajo   :0   f.age-(2,4]:0
f.tax-(145,155]:4   f.mpg-medio  :0   f.age-(+4) :0
f.tax-(155,205]:0   f.mpg-alto   :0

```

```

f.price      Audi      mout      claKMPCA
f.price-(3.15e+04,1.09e+05]:2   Audi No :0   NoMOut :5   1:5
f.price-(1.95e+04,2.2e+04] :1   Audi Yes:5   YesMOut:0   2:0
f.price-(2.2e+04,2.6e+04] :1                               3:0
f.price-(2.6e+04,3.15e+04] :1
f.price-[899,1.1e+04] :0
f.price-(1.1e+04,1.4e+04] :0
(Other) :0

claHCMC
f.claHCMC-1:5
f.claHCMC-2:0
f.claHCMC-3:0
f.claHCMC-4:0
f.claHCMC-5:0
f.claHCMC-6:0
f.claHCMC-7:0

```

Si recogemos los datos de los parámetros del primer cluster, podemos ver como todos los vehículos son Audi, de cambio automático y gasolina. También tienen consumos muy bajos y tiene entre 1 y 2 años.

13 K-Means Clustering desde MCA

Por último, vamos a realizar un clustering con el algoritmo de K-Means a partir del MCA.

```
ppcc <- res.mca$ind$coord[,1:7];
kc<-kmeans(dist(ppcc),7)
```

En primer lugar, vamos a analizar el porcentaje de distancias que se acumulan con la suma de distancias entre clusters respecto al total.

```
kc$betweenss/kc$totss
```

```
[1] 0.5983192
```

Podemos ver que es de un 61%, lo que a priori es peor que en la clusterización a partir de ACP.

En el caso de las distancias dentro de los clusters, estas suman un 39%.

```
kc$tot.withinss/kc$totss
```

```
[1] 0.4016808
```

13.1 Profiling de la clusterización

Vamos a analizar las cualidades de los distintos clusters que se han creado.

```
df[-11, "claKMMCA"] <- kc$cluster  
df$claKMMCA <- factor(df$claKMMCA)  
km.catdes<-catdes(df[,c("claKMMCA",vars_num,vars_cat, "price")],1)
```

```
km.catdes$quanti[1]
```

```
$'1'  
      v.test Mean in category Overall mean sd in category   Overall sd  
mpg    21.742764       61.941300     53.097238      6.467213    11.166173  
age     2.910341        4.010687     3.798202      1.150456    2.004245  
price   -3.210497      19972.467176  21206.729400     4612.770772 10553.646562  
tax     -8.865280      142.875519    146.865790      6.591551    12.355981  
          p.value  
mpg    8.087805e-105  
age    3.610346e-03  
price  1.325058e-03  
tax    7.631214e-19
```

Para el primer cluster, podemos ver como la variable más determinante ha sido tax, en concreto, se agrupan los individuos que han pagado más impuestos. Podemos ver como el consumo de estos individuos es más alto. Puede que hablamos de vehículos ecológicos con subvenciones.

```
km.catdes$quanti[2]
```

```
$'2'  
      v.test Mean in category Overall mean sd in category   Overall sd  
mpg    -7.49953       49.371895     53.097238      5.9014358 11.166173  
mileage -19.38849      4594.067538  23318.245205     4104.3003896 21708.552196  
age     -22.09821      1.827887     3.798202      0.4746356 2.004245  
          p.value  
mpg    6.404706e-14  
mileage 9.652378e-84  
age    3.288273e-108
```

Para el segundo cluster, podemos ver como la variable más determinante es el precio, que es muy inferior a la media. Podemos destacar también que los vehículos son más viejos y con más kilometraje.

```
km.catdes$quanti[3]
```

```
$'3'  
      v.test Mean in category Overall mean sd in category   Overall sd  
price   9.284864      24782.542113  21206.729400     7556.2721944 10553.646562  
mpg    -10.213411      48.935528     53.097238      7.8554900 11.166173
```

```

mileage -11.365651      14314.528331 23318.245205   9029.5775721 21708.552196
age     -12.735161           2.866769    3.798202      0.9786456   2.004245
          p.value
price    1.619135e-20
mpg     1.726861e-24
mileage 6.200065e-30
age     3.770667e-37

```

En lo que se refiere al tercer cluster, podemos ver como la variable más determinante es el consumo. En concreto aparecen vehículos con un consumo inferior. Estos vehículos también son más nuevos y tienen un kilometraje inferior a la media.

```
km.catdes$quanti[7]
```

```

$'7'
      v.test Mean in category Overall mean sd in category Overall sd
tax     43.328696      172.167472 146.865790      21.655832 12.355981
age     19.569540       5.651854   3.798202      1.627987 2.004245
mileage 16.631263 40381.136852 23318.245205   19479.532576 21708.552196
price   -2.020259 20199.087591 21206.729400    7182.147983 10553.646562
mpg     -12.321526      46.594963 53.097238      6.884532 11.166173
          p.value
tax     0.000000e+00
age     2.812057e-85
mileage 4.137814e-62
price   4.335651e-02
mpg     6.936948e-35

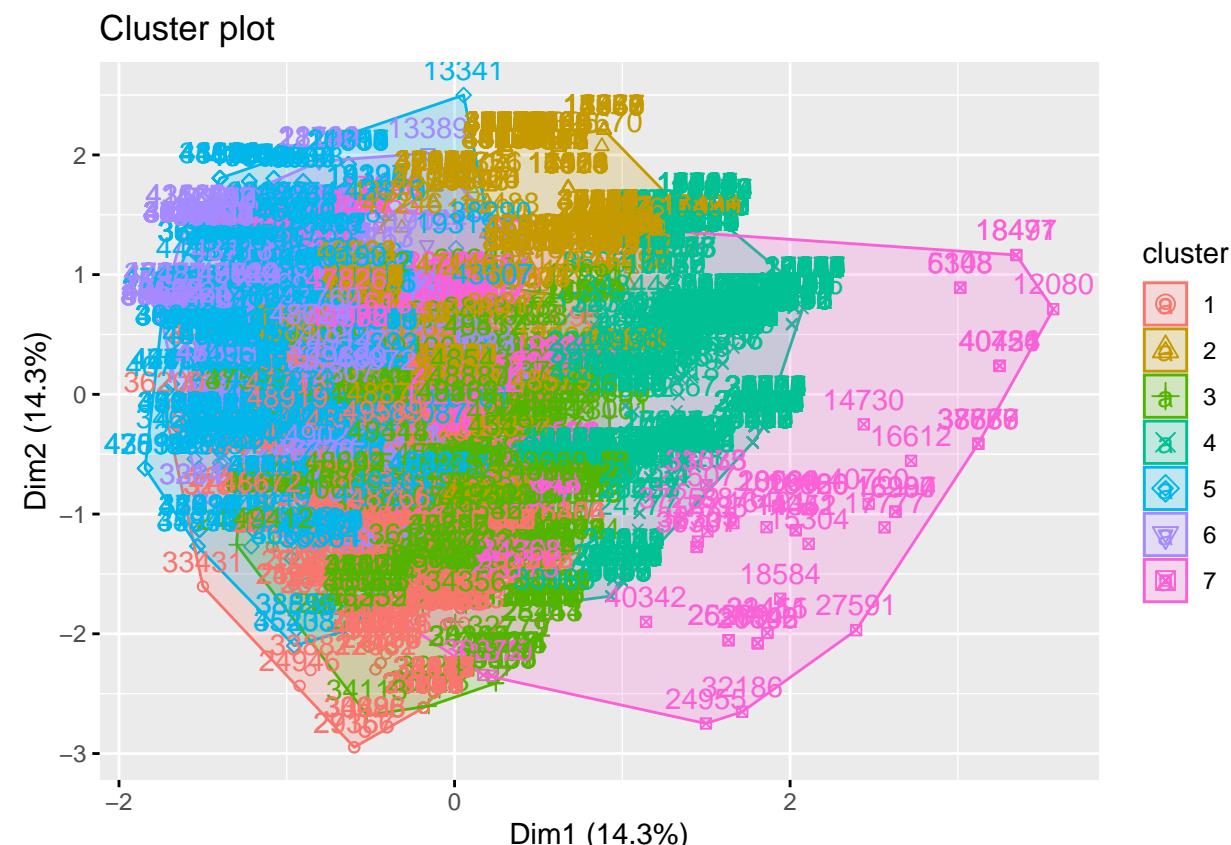
```

Pasando directos al cluster 7, podemos ver como en este caso, aparecen vehículos más baratos y con menor kilometraje e impuestos que la media, aunque en general no están muy alejados del centroide del dataset.

Con estos dos resultados, se puede concluir que la clusterización desde el ACP es de mayor calidad que la generada a partir del ACM.

Este hecho es aún más evidente si lo representamos de manera gráfica:

```
fviz_cluster(kc, data=ppcc)
```

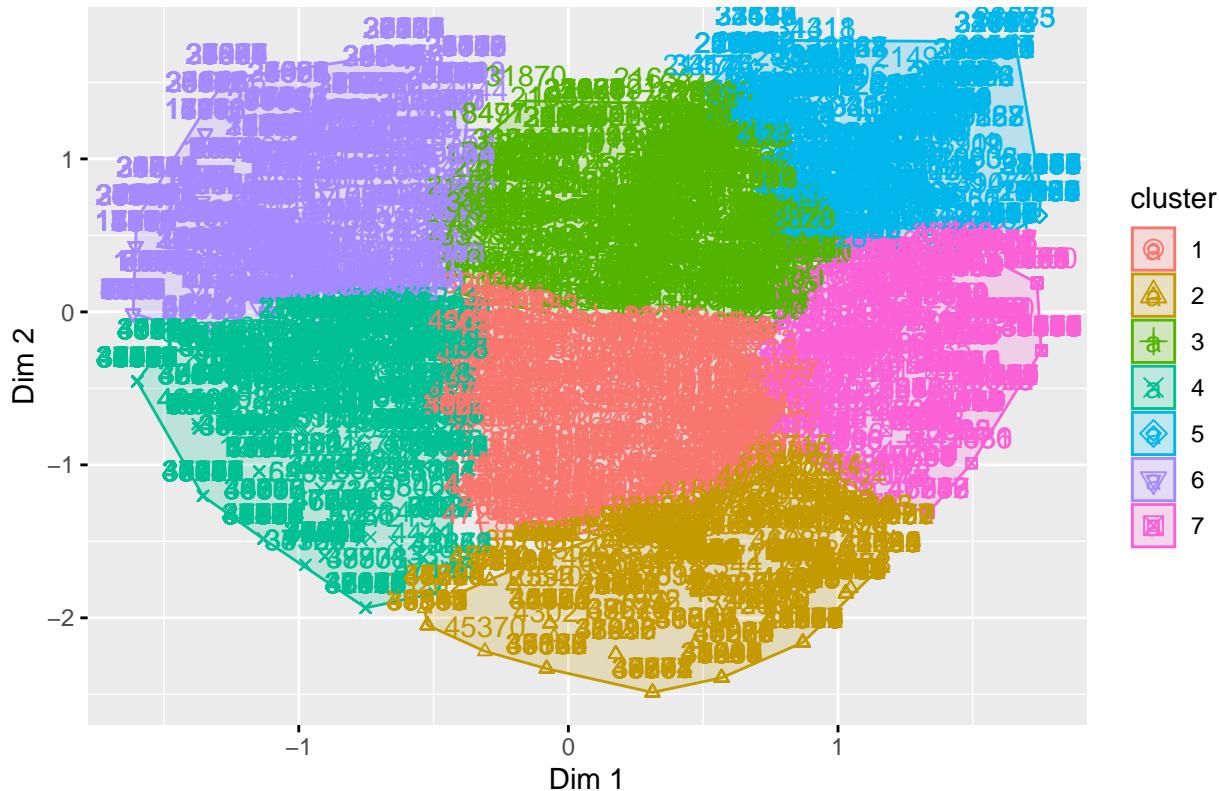


Lamentablemente, al solo poder representar los clusters en 2 dimensiones, el gráfico resultante es prácticamente incomprensible, de modo que vamos a realizar el mismo proceso pero solo aportando las coordenadas pertenecientes a las 2 primeras componentes del MCA, que son las que más variabilidad acumulan (25%).

<Solo por motivos de visualización. Si pudieramos representar las 7 dimensiones veríamos como las clusterización SÍ que tiene sentido>

```
ppcc <- res.mca$ind$coord[,1:2];
kc<-kmeans(dist(ppcc),7)
fviz_cluster(kc, data=ppcc)
```

Cluster plot



En

este caso si que podemos ver bien diferenciados los distintos clusters.

Si analizamos las distancias, podemos ver como en este caso, se crean clusters más diferenciados entre si, pero con individuos más similares. Esto es debido a que solo se están usando 2 dimensiones y la variabilidad que se acumula es muy baja.

```
kc$betweenss/kc$totss;kc$tot.withinss/kc$totss
```

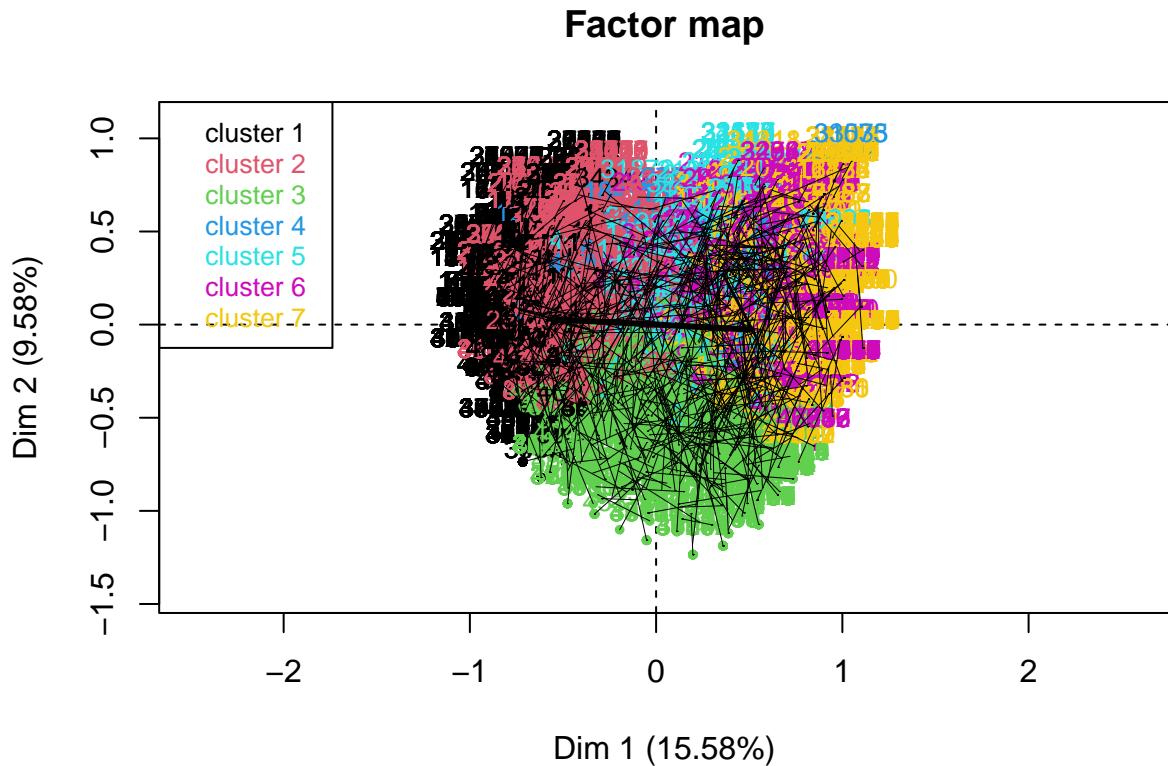
```
[1] 0.8338669
```

```
[1] 0.1661331
```

Podemos ver que la distancia entre clusters acumula un 82% del total, mientras que las distancias dentro de los clusters acumulan solo un 17%. Sería un resultado bastante bueno si con las dos dimensiones que estamos representando acumuláramos mayor variabilidad.

Por último, volvemos a poner el gráfico generado a partir del clustering jerárquico con MCA para comparar las distintas clusterizaciones que se han llevado a cabo según clustering jerárquico y Kmeans.

```
plot.HCPC(res.hcmc, choice="map")
```



13.2 Variables explicativas numéricas

Primero de todo, vamos a empezar aplicando una corrección para los valores de nuestras variables numéricas eliminando los valores 0 para poder evitar algunos errores que podrían salir a posteriori.

```
vars_con
11<-which(df$age==0);11
df$age[11]<-0.5

11<-which(df$tax==0);11
df$tax[11]<-0.5

11<-which(df$mpg==0);11
df$mpg[11]<-0.5

11<-which(df$mileage==0);11
df$mileage[11]<-0.5
```

14 Modelo de regresión lineal

14.1 Variables numéricas

Planteamos nuestro primer modelo basado en las variables numéricas. Con este modelo, pretendemos plantear una regresión lineal que tenga como target la variable price, y que use como variables explicativas mileage, tax, mpg y age.

```
m1<-lm(price~mileage+tax+mpg+age,data=df)
summary(m1)
```

```
Call:
lm(formula = price ~ mileage + tax + mpg + age, data = df)
```

Residuals:

	Min	1Q	Median	3Q	Max
-34504	-4799	-255	3259	69279	

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	5.476e+04	1.627e+03	33.652	< 2e-16 ***
mileage	-4.516e-02	8.015e-03	-5.635	1.85e-08 ***
tax	-2.301e+01	9.385e+00	-2.452	0.0142 *
mpg	-4.138e+02	1.086e+01	-38.098	< 2e-16 ***
age	-1.883e+03	8.658e+01	-21.747	< 2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7386 on 4995 degrees of freedom

Multiple R-squared: 0.5107, Adjusted R-squared: 0.5103

F-statistic: 1303 on 4 and 4995 DF, p-value: < 2.2e-16

En el summary que acabamos de mostrar, podemos apreciar varias cosas:

En primer lugar, podemos ver los errores residuales que se generan en el modelo.

También podemos ver los coeficientes que se plantean para las diferentes variables del modelo así como el término independiente (Intercept) 5.476e+04. Podemos ver como a priori, los coeficientes para todas las variables son significativamente distintos a 0.

Con el valor R-squared, podemos apreciar también que el modelo este modelo explica un 51% de la variabilidad de la variable price.

Por último, podemos ver también como el F-statistic nos indica que nuestra hipótesis nula de que todos los coeficientes sean iguales a 0 se puede rechazar.

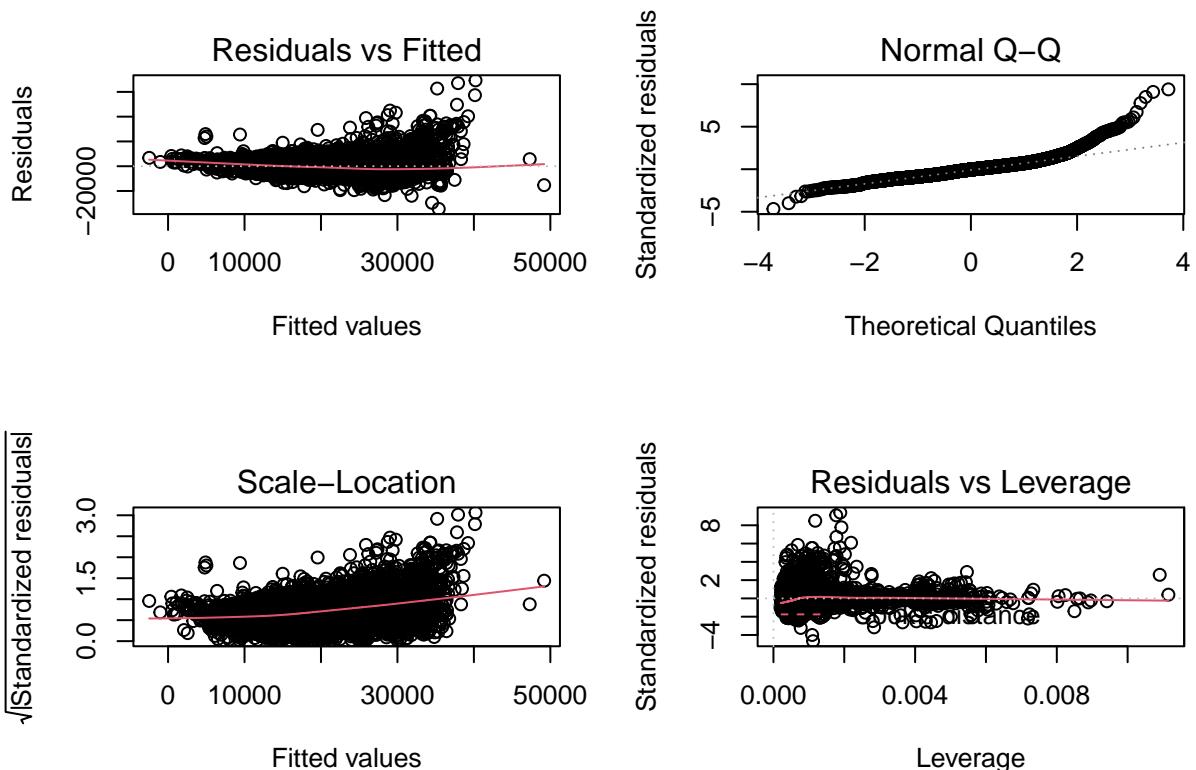
Si analizamos los valores de vif (variable inflation factor), podemos ver que los más altos corresponden a las variables de mileage y age, cercanos a 3. Estos valores nos indican la co-linealidad de las variables, y empezarían a ser preocupantes cuando se acercan al 5, de modo que de momento son correctos.

```
vif(m1)
```

mileage	tax	mpg	age
2.774734	1.232502	1.348349	2.760318

En los siguientes gráficos, podemos apreciar algunas de las características del modelo.

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



En el gráfico de Residuals vs. Fitted podemos apreciar como no existe homocedasticidad en el modelo, ya que los residuos estan distribuidos de manera heterogénea.

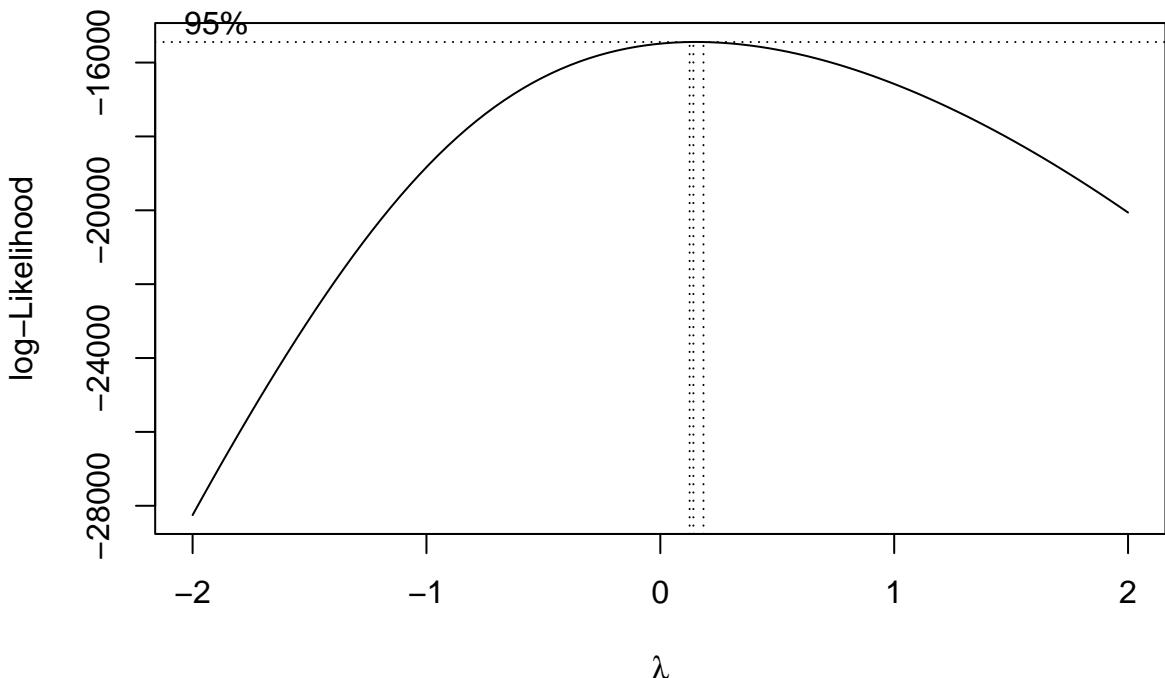
En el gráfico Normal Q–Q podemos ver como el modelo no presenta normalidad, ya que los residuos estandarizados se alejan de la recta que marca la normalidad.

En el gráfico Scale–Location, podemos volver a apreciar la heterosticidad del modelo, analizando la raiz de los residios estandarizados.

En el último gráfico, el de Residuals vs Leverage, podemos apreciar como a priori no parece que existan observaciones influyentes, ya que ninguno de los puntos que se muestran tiene una distancia de Cook que indique sobre-influencia.

A continuación, vamos a determinar si alguna de las variables que hemos introducido en el modelo anterior requiere algún tipo de transformación para lograr un mejor ajuste del modelo.

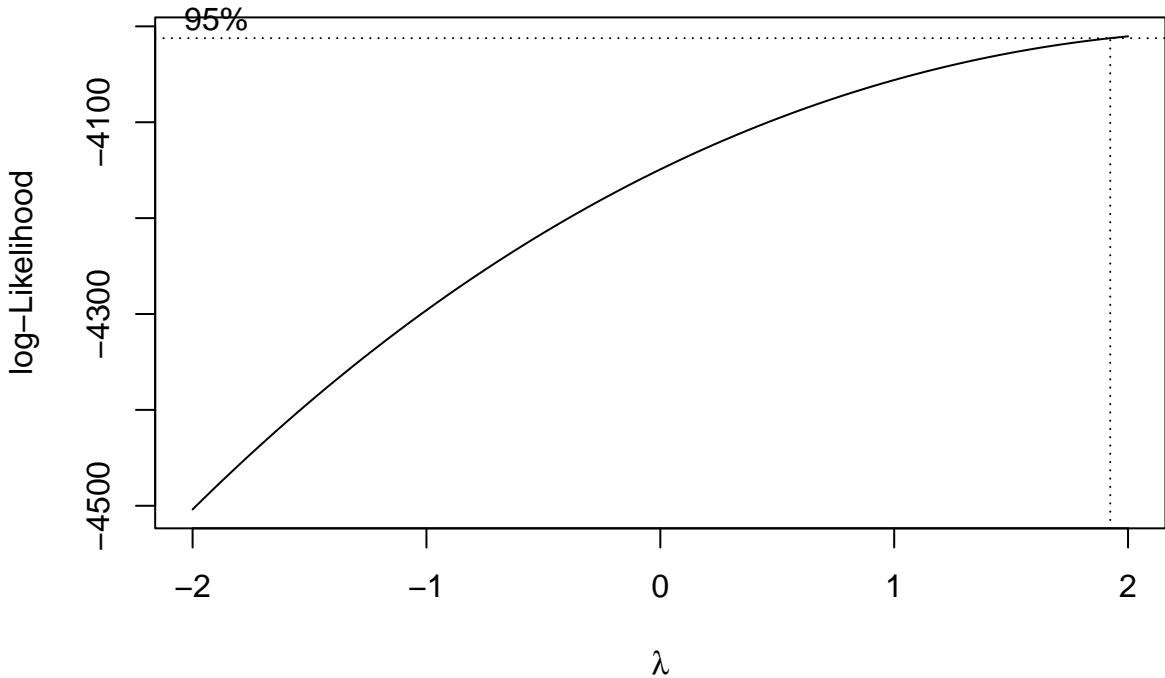
```
par(mfrow=c(1,1))
boxcox(price~mileage+tax+mpg+age, data=df)
```



En el gráfico podemos ver como el intervalo de lambda que aparece es cercano al 0, hecho que indicaría la necesidad de transformar nuestra variable target price a una escala logarítmica.

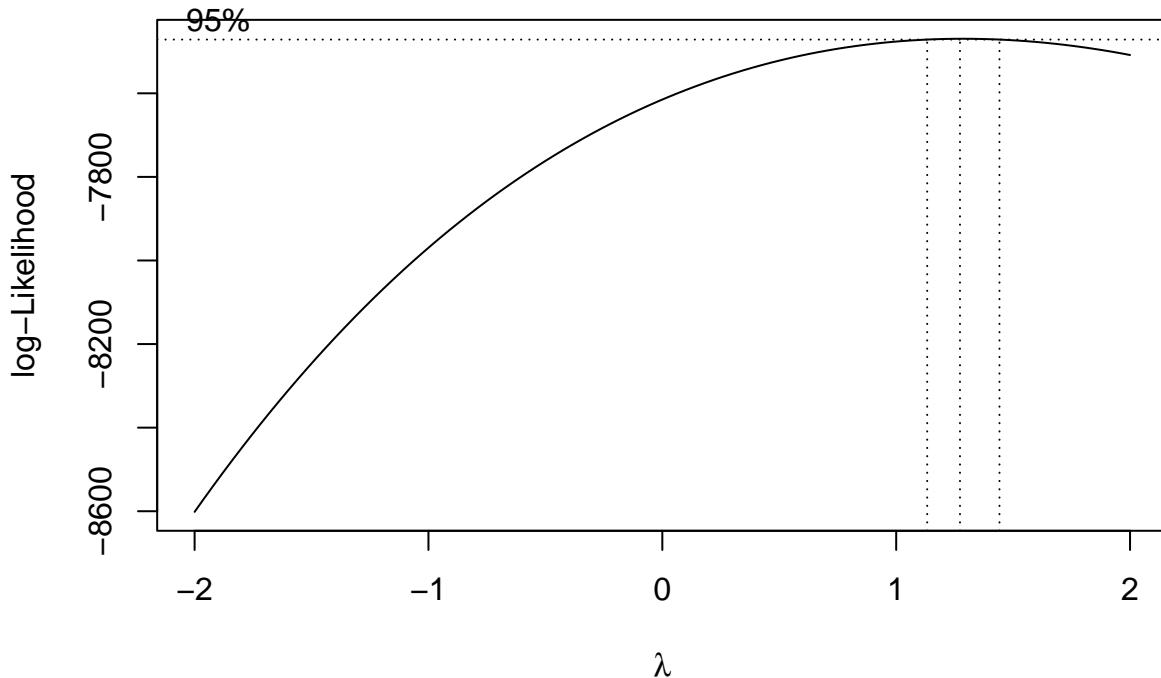
Vamos a volver a evaluar la co-linealidad de nuestra variable target con las dependientes habiendo aplicado la transformación logarítmica.

```
boxcox(log(price)~mileage+tax+mpg+age, data=df)
```



Como podemos ver, en este caso aparece un máximo en la función cerca de lambda = 2, de modo que sería recomendable aplicar una transformación cuadrática al target:

```
boxcox(log(price)^2~mileage+tax+mpg+age, data=df)
```



Vamos a proceder a plantear un segundo modelo que incluya esta transformación.

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

Call:

```
lm(formula = log(price) ~ mileage + tax + mpg + age, data = df)
```

Residuals:

Min	1Q	Median	3Q	Max
-3.6717	-0.1762	0.0232	0.1961	1.2362

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	1.096e+01	6.903e-02	158.851	<2e-16 ***
mileage	-3.355e-06	3.400e-07	-9.868	<2e-16 ***
tax	7.973e-04	3.981e-04	2.003	0.0452 *
mpg	-1.376e-02	4.607e-04	-29.857	<2e-16 ***
age	-1.116e-01	3.673e-03	-30.376	<2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3133 on 4995 degrees of freedom

Multiple R-squared: 0.5789, Adjusted R-squared: 0.5786

F-statistic: 1717 on 4 and 4995 DF, p-value: < 2.2e-16

Para este modelo, podemos apreciar como todos los coeficientes siguen siendo estadísticamente diferentes a 0 según el p-valor que se muestra en la última columna. Cabe destacar que para el caso de la variable tax, el p-valor ha subido y es cercano a 0.1.

Podemos ver también como el valor R-squared ha aumentado a 0.58, indicando que este nuevo modelo acumula más explicabilidad de nuestro target.

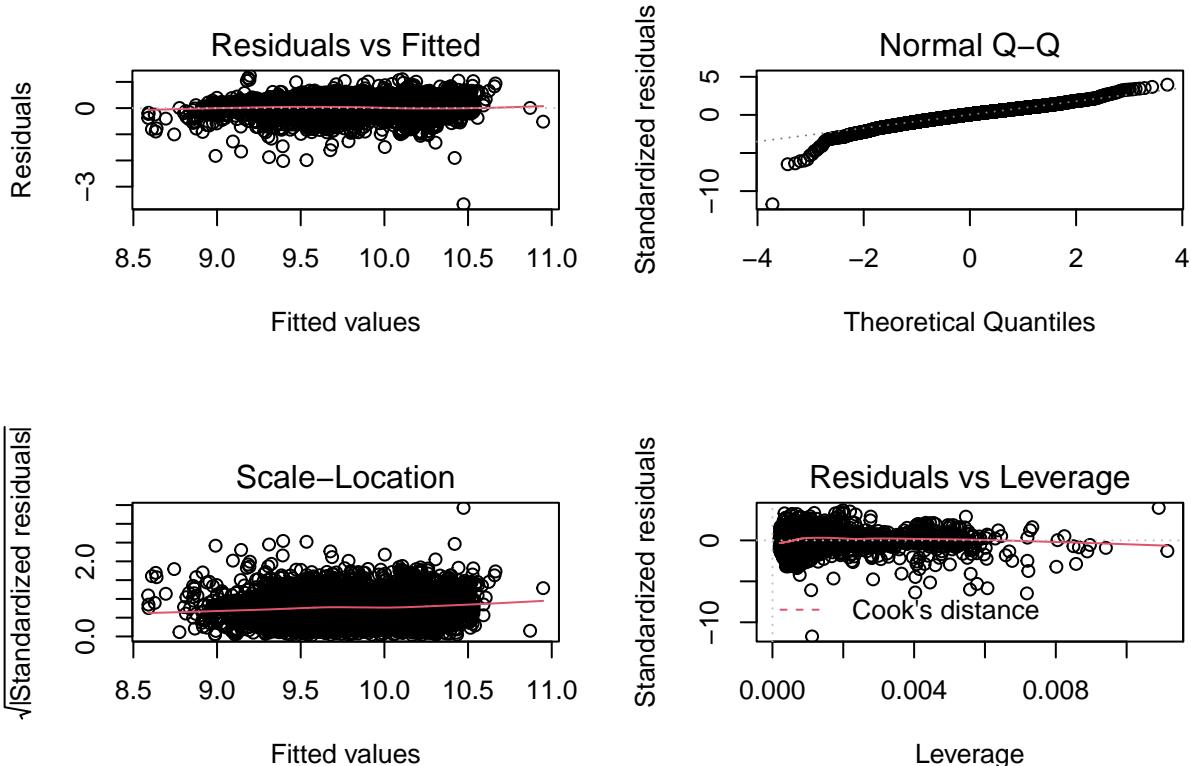
También podemos ver como los valores de vif se mantienen similares a los que aparecían en el modelo anterior, indicando que no parece existir co-linealidad entre las variables.

```
vif(m2)
```

mileage	tax	mpg	age
2.774734	1.232502	1.348349	2.760318

Observemos algunos gráficos para analizar este nuevo modelo que hemos planeado.

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



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

Podemos ver como según el primer gráfico, el modelo ha adquirido homocedasticidad, siendo la distribución de los residuos más homogénea.

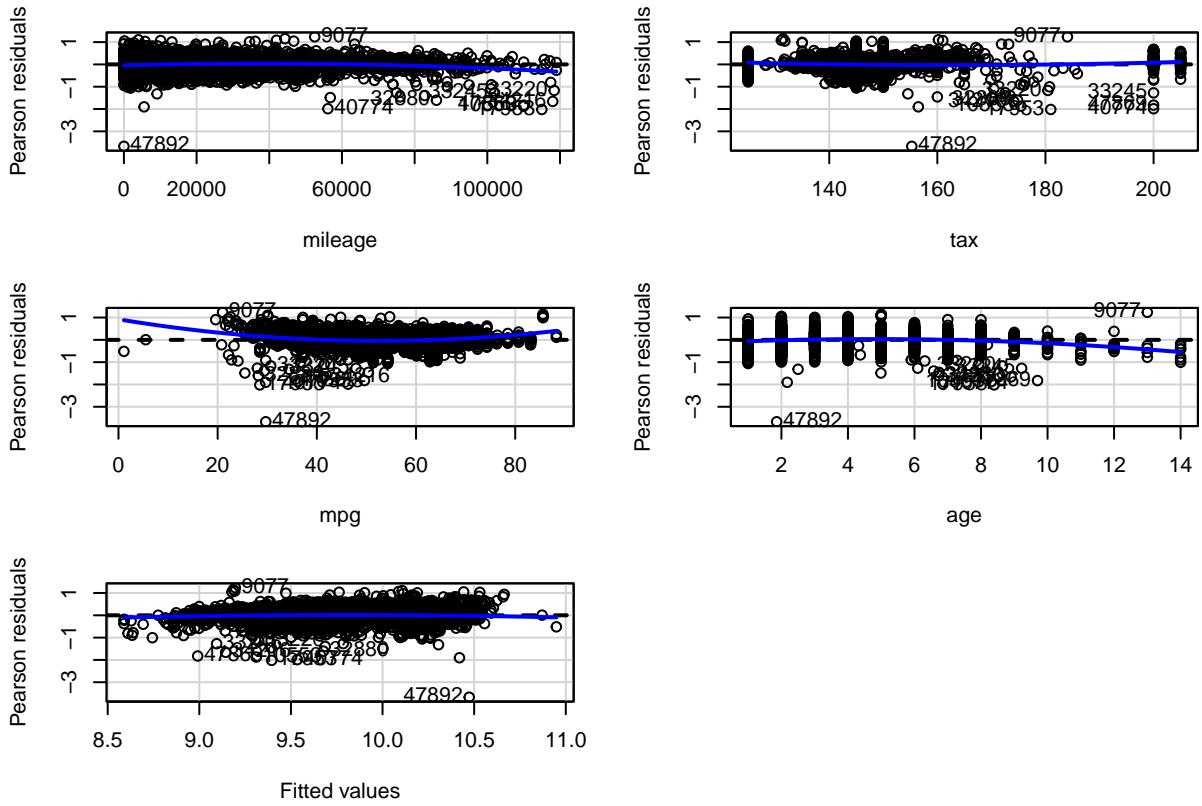
En lo que se refiere a la normalidad, podemos ver como este modelo presenta más normalidad que el anterior, al menos en lo que se refiere a los quantiles superiores. Sin embargo, para los cuantiles inferiores, aún aparece cierta distancia con la recta que describe la normalidad.

Por último, para el caso de la sobre-influencia en el modelo, podemos ver un resultado similar al del anterior modelo.

Vamos a proceder a realizar un análisis más en profundidad del modelo.

En los siguientes gráficos podemos ver como se distribuyen los residuos.

```
residualPlots(m2,id=list(method=cooks.distance(m2),n=10))
```



```

Test stat Pr(>|Test stat|)
mileage   -8.9294      < 2.2e-16 ***
tax        6.7284      1.909e-11 ***
mpg       13.1778      < 2.2e-16 ***
age       -10.5882     < 2.2e-16 ***
Tukey test -2.9525      0.003152 **

Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

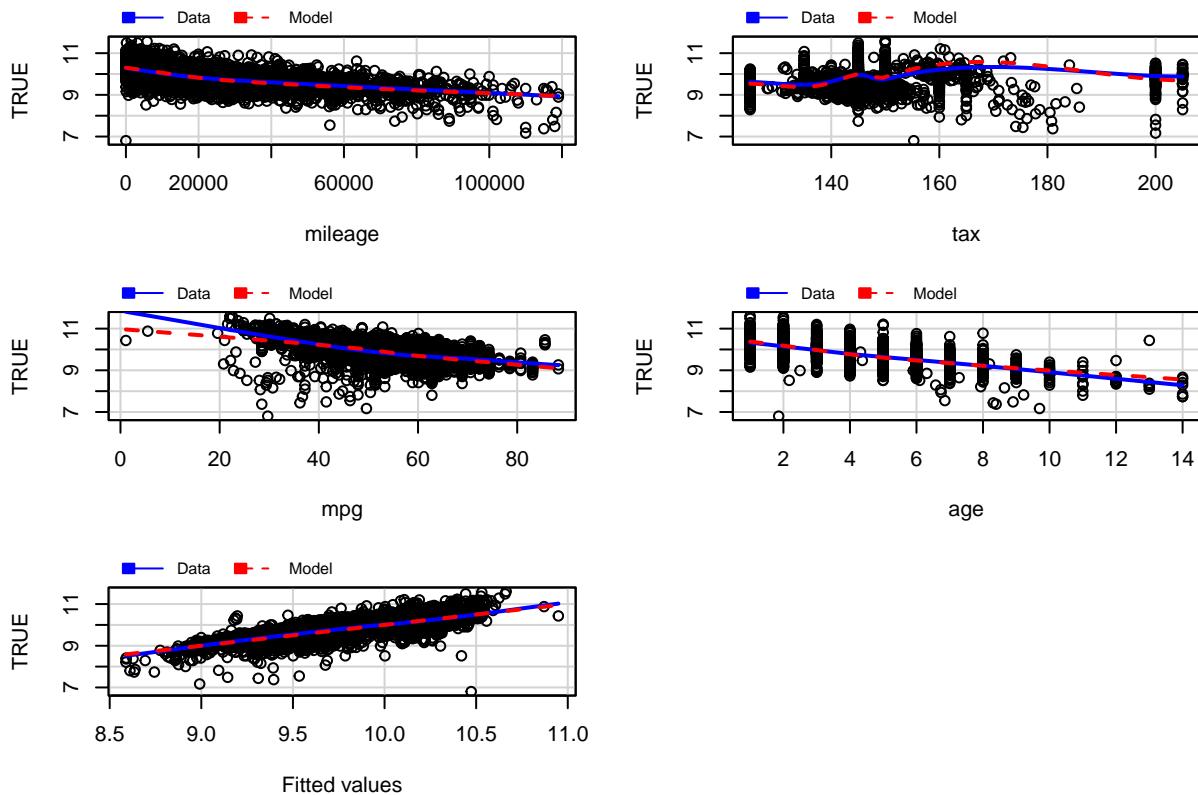
Algunos casos a destacar son el de mileage, donde se puede apreciar una cierta acumulación en los valores bajos a la vez que aparecen puntos que se alejan de la nube y de la curva para valores más altos.

También podemos destacar que en el caso de age, que es una variable que originalmente generamos a partir de la variable year y se puede apreciar su distribución en columnas, aparece una nube de puntos cerca del centro, probablemente debido a la imputación a partir de PCA que se realizó en la primera entrega. Podemos ver algo parecido pero no tan evidente para la variable tax.

Con la siguiente función procederemos a ver el ajuste del modelo con los datos.

```
marginalModelPlots(m2)
```

Marginal Model Plots



Podemos ver como el ajuste para el caso de la variable mileage o age parecen casi perfectos, mientras que para las variables mpg y tax existe una cierta desviación entre las curvas roja y azul, que nos indican que hay falta de linealidad entre la variable target y las variables explicativas.

En el caso de los Fitted values, las rectas se ajustan a la perfección.

Aplicando la función de boxTidwell podemos determinar las transformaciones que deberíamos aplicar a nuestro modelo para mejorarlo.

```
#boxTidwell(log(price)^2~mileage+tax+age+mpg, data=df[!df$mout=="YesMOut",], verbose=TRUE)
```

Sin embargo, cuando ejecutamos la función pasando como entrada el modelo que habíamos planteado, nos encontramos que la función falla. Aplicando el modo verbose, podemos ver como el motivo del fallo es la tendencia a +infinito del exponente de la variable mpg.

Como no sé como proceder, vamos a realizar algunos tests:

En primer lugar, vamos a crear un nuevo modelo sin la variable mpg y lo vamos a evaluar con la función boxTidwell para determinar las transformaciones a aplicar a nuestras variables explicativas (aunque hemos tenido que aumentar el número máximo de iteraciones).

```
m3<-lm(log(price)~mileage+tax+age, data=df)
boxTidwell(log(price)~mileage+tax+age, data=df[!df$mout=="YesMOut",], max.iter=100)
```

```
MLE of lambda Score Statistic (z) Pr(>|z|)
mileage      0.56718      3.7432 0.0001817 ***
tax          4.06382      6.7974 1.065e-11 ***
age          1.12393     -1.6431 0.1003561
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

iterations = 30
```

Como podemos ver, este nuevo modelo que excluye la variable mpg pierde explicabilidad, pero nos permite ejecutar la función boxTidwell para determinar las transformaciones que deberíamos aplicar a nuestras variables.

Vamos a aplicar las transformaciones que aparecen con la función boxTidwell para crear un nuevo modelo.

```
m4<-lm(log(price)~sqrt(mileage)+poly(tax,4)+age, data=df)
summary(m4)
```

Call:
`lm(formula = log(price) ~ sqrt(mileage) + poly(tax, 4) + age,
 data = df)`

Residuals:

Min	1Q	Median	3Q	Max
-3.6437	-0.1893	0.0094	0.1871	1.6992

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	10.5949900	0.0118047	897.521	< 2e-16 ***
sqrt(mileage)	-0.0015588	0.0001148	-13.576	< 2e-16 ***
poly(tax, 4)1	4.9247954	0.3440468	14.314	< 2e-16 ***
poly(tax, 4)2	2.5889209	0.3494831	7.408	1.50e-13 ***
poly(tax, 4)3	-3.8225677	0.3698610	-10.335	< 2e-16 ***
poly(tax, 4)4	1.4936334	0.3362566	4.442	9.11e-06 ***
age	-0.1409327	0.0043431	-32.449	< 2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3354 on 4993 degrees of freedom
 Multiple R-squared: 0.5176, Adjusted R-squared: 0.517
 F-statistic: 892.8 on 6 and 4993 DF, p-value: < 2.2e-16

```
#boxTidwell(log(price)~sqrt(mileage)+poly(tax,4)+age, data=df[!df$mout=="YesMOut",], verbose=TRUE)
```

Volviendo a aplicar la función de boxTidwell, podemos ver como, en este caso vuelve a fallar debido a que algunos de los coeficientes de los monomios que se han generado con la función poly(tax,4) son negativos. Sin embargo, el modelo que excluye la variable mpg, aún aplicando todas las transformaciones recomendadas, tiene un valor de R-squared inferior al original.

Si ejecutamos el test de Clarke que nos permite analizar modelos no anidados, (lo he encontrado por internet) podemos ver que, aparentemente, el modelo original, que incluye la variable mpg pero ninguna transformación a parte de la logarítmica para el target, es mejor que el modelo que excluye esta variable pero incluye las transformaciones.

```
library(games)
clarke(m2, m4)
```

Clarke test for non-nested models

Model 1 log-likelihood: -1289
 Model 2 log-likelihood: -1629
 Observations: 5000
 Test statistic: 3205 (64%)

Model 1 is preferred (p < 2e-16)

Vamos a probar incluyendo la variable mpg en el modelo que ya habíamos planteado aplicando las transformaciones que aparecían con la función boxTidwell.

```
m5<-update(m4, ~.+mpg)
summary(m5)
```

Call:
`lm(formula = log(price) ~ sqrt(mileage) + poly(tax, 4) + age +
 mpg, data = df)`

Residuals:

	Min	1Q	Median	3Q	Max
	-3.6780	-0.1769	0.0206	0.1969	1.3587

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	11.1517010	0.0234285	475.988	< 2e-16 ***
sqrt(mileage)	-0.0007257	0.0001117	-6.499	8.9e-11 ***
poly(tax, 4)1	0.4878986	0.3612542	1.351	0.177
poly(tax, 4)2	2.1950054	0.3269186	6.714	2.1e-11 ***
poly(tax, 4)3	0.5668394	0.3821265	1.483	0.138
poly(tax, 4)4	0.3166057	0.3172552	0.998	0.318
age	-0.1204402	0.0041294	-29.167	< 2e-16 ***
mpg	-0.0140643	0.0005222	-26.935	< 2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3134 on 4992 degrees of freedom

Multiple R-squared: 0.5788, Adjusted R-squared: 0.5782

F-statistic: 979.9 on 7 and 4992 DF, p-value: < 2.2e-16

En este caso, se puede ver como la incorporación de la variable mpg sí que aumenta la explicabilidad del modelo, generando un modelo más completo.

Además, aplicando la función anova, podemos ver como se rechaza la hipótesis de equivalencia, de modo que el nuevo modelo es mejor.

```
anova(m4,m5)
```

Analysis of Variance Table

Model	log(price) ~ sqrt(mileage) + poly(tax, 4) + age	log(price) ~ sqrt(mileage) + poly(tax, 4) + age + mpg
1	4993 561.63	4992 490.36
2		1 71.264 725.48 < 2.2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sin embargo, si lo comparamos con el original:

```
clarke(m2,m5)
```

Clarke test for non-nested models

Model 1 log-likelihood: -1289

Model 2 log-likelihood: -1290

Observations: 5000

Test statistic: 2824 (56%)

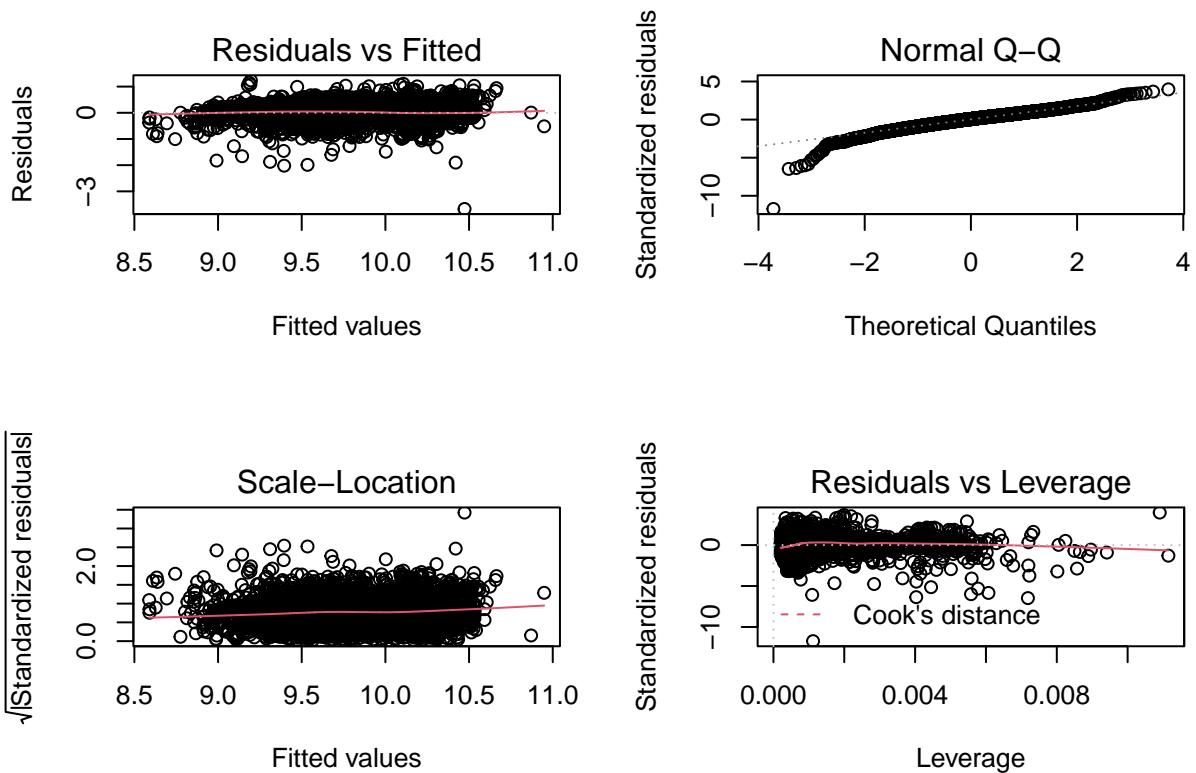
Model 1 is preferred (p < 2e-16)

Podemos ver como el test de clarke para modelos no anidados determina que el primer modelo es preferible.

De modo que seguiremos avanzando con el segundo modelo que hemos planteado, que incluye la variable mpg pero no incluye ninguna transformación a parte de la del target price.

Vamos a proceder a observar algunos gráficos para analizar este modelo:

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



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

Podemos ver como según el primer gráfico, el modelo ha adquirido homocedasticidad, siendo la distribución de los residuos más homogénea.

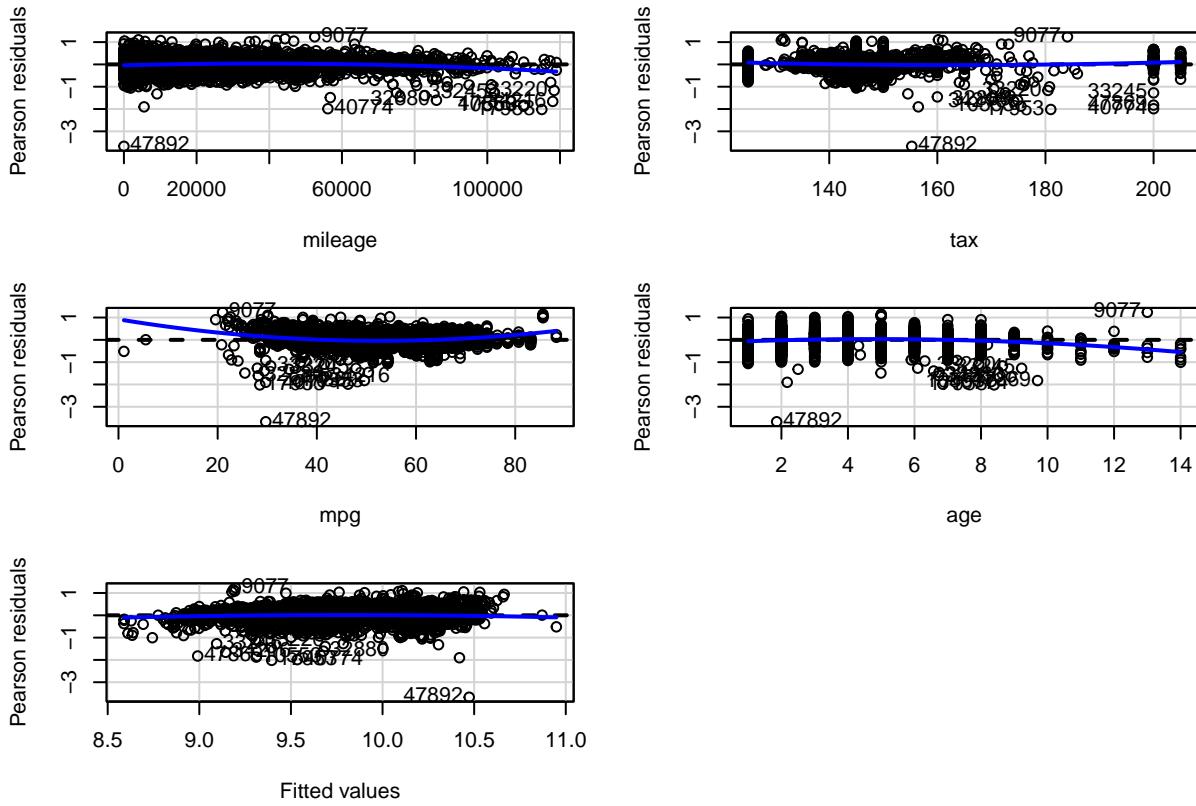
En lo que se refiere a la normalidad, podemos ver como para este modelo tiene más normalidad que el anterior, al menos en lo que se refiere a los cuantiles superiores. Sin embargo, para los cuantiles inferiores, aún aparece cierta distancia con la recta que describe la normalidad.

Por último, para el caso de la sobre-influencia en el modelo, podemos ver un resultado similar al del anterior modelo.

Vamos a proceder a realizar un análisis más en profundidad del modelo.

En los siguientes gráficos podemos ver como se distribuyen los residuos.

```
residualPlots(m2,id=list(method=cooks.distance(m2),n=10))
```



```

Test stat Pr(>|Test stat|)
mileage   -8.9294      < 2.2e-16 ***
tax        6.7284      1.909e-11 ***
mpg       13.1778      < 2.2e-16 ***
age       -10.5882     < 2.2e-16 ***
Tukey test -2.9525      0.003152 **

---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

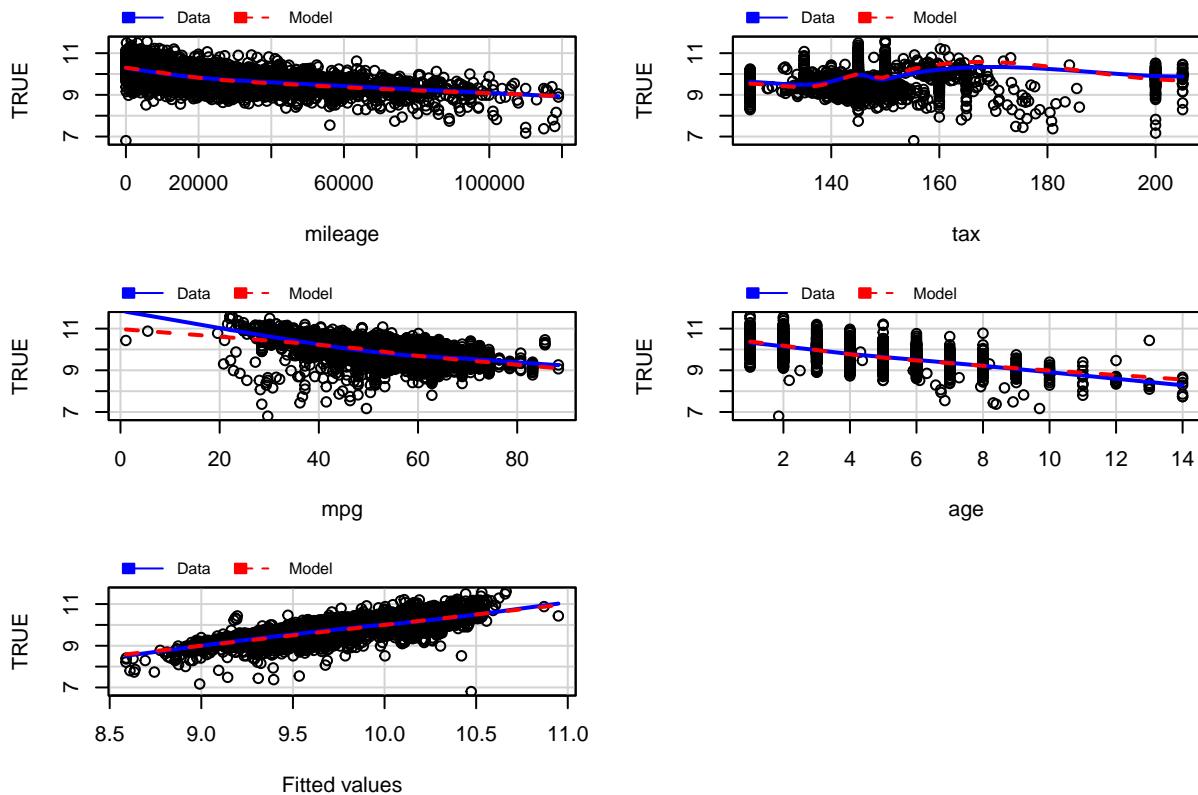
Algunos casos a destacar son el de mileage, donde se puede apreciar una cierta acumulación en los valores bajos, a la vez que aparecen puntos que se alejan de la nube y de la curva para valores más altos.

También podemos destacar que en el caso de age, que es una variable que originalmente generamos a partir de la variable year tiene una distribución en columnas. Aparece una nube de puntos cerca del centro, probablemente debido a la imputación a partir de PCA que se realizó en la primera entrega. Podemos ver algo parecido pero no tan evidente para la variable tax.

Con la siguiente función procederemos a ver el ajuste del modelo con los datos.

```
marginalModelPlots(m2)
```

Marginal Model Plots



Podemos ver como el ajuste para el caso de la variable mileage o age parecen casi perfectos, mientras que para las variables mpg y tax existe una cierta desviación entre las curvas roja y azul.

En el caso de los Fitted values, las rectas se ajustan casi a la perfección.

Vamos a proceder a volver a generar el modelo excluyendo los multivariant outliers.

```
m6<-update(m2, data=df[!df$mout=="YesMOut",])
summary(m6)
```

Call:

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

Residuals:

Min	1Q	Median	3Q	Max
-2.07976	-0.17094	0.02287	0.19110	0.82842

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	1.082e+01	6.468e-02	167.283	< 2e-16 ***
mileage	-2.294e-06	3.654e-07	-6.279	3.71e-10 ***
tax	1.628e-03	3.715e-04	4.381	1.20e-05 ***
mpg	-1.410e-02	4.502e-04	-31.314	< 2e-16 ***
age	-1.077e-01	3.830e-03	-28.117	< 2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2833 on 4805 degrees of freedom

Multiple R-squared: 0.5769, Adjusted R-squared: 0.5766

F-statistic: 1638 on 4 and 4805 DF, p-value: < 2.2e-16

En el caso del R-squared, podemos ver como este ha bajado, aunque infimamente, indicando que el modelo anterior aglutinaba más variabilidad de nuestro target.

Vamos a proceder a estudiar la validez de nuestro tercer modelo.

```
Anova(m6)
```

Anova Table (Type II tests)

```
Response: log(price)
          Sum Sq Df F value    Pr(>F)
mileage     3.16   1 39.424 3.711e-10 ***
tax         1.54   1 19.197 1.204e-05 ***
mpg        78.71   1 980.586 < 2.2e-16 ***
age        63.46   1 790.540 < 2.2e-16 ***
Residuals 385.69 4805
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

En este test podemos ver que todas las variables que aparecen son significativas, de modo que ninguna de estas variables es prescindible.

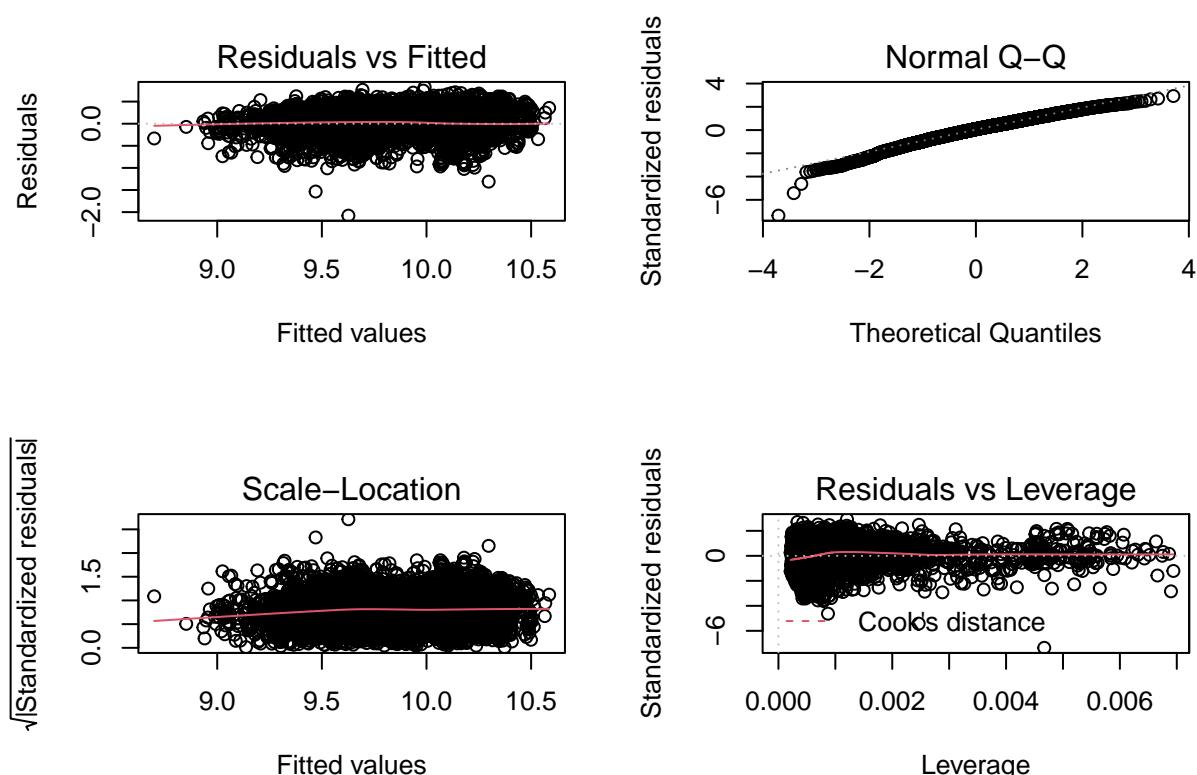
Con la función de VIF podemos ver como, los vifs asociados a mileage y age son mayores que 3, pero de momento no debería importarnos mucho.

```
vif(m6)
```

```
mileage      tax      mpg      age
2.988723 1.179943 1.399036 3.021153
```

Vamos a analizar los gráficos del modelo:

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



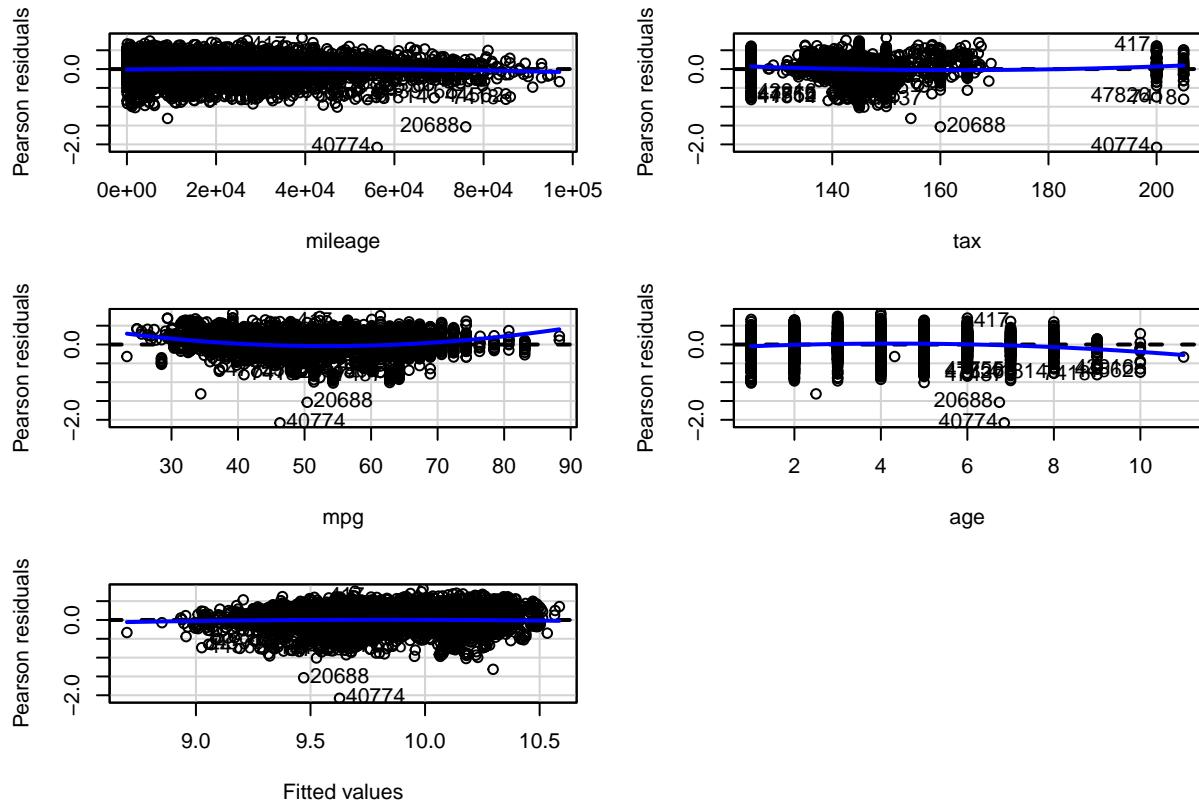
En el de Residuals vs Fitted y Scale-Location podemos ver como el modelo presenta bastante homocedasticidad.

En el de Normal Q-Q, similarmente al m2, podemos apreciar como en los cuantiles superiores aparece normalidad, mientras que para los inferiores, la normalidad de nuestro modelo se aleja de la recta debido a algunas observaciones extrañas.

Por último, en el gráfico de Residuals vs Leverage, podemos ver como para este modelo siguen sin aparecer puntos con una distancia de Cook relevante, de modo que no parece existir sobre-influencia de ninguna observación.

En los siguientes gráficos podemos apreciar como las rectas se ajustan bastante, excepto en el caso de la variable mpg, a la que, dados los problemas que aparecen con la tendencia a infinito de su exponente en las iteraciones de la función boxTidwell, no podemos determinar la transformación que se le debería aplicar para mejorar el modelo.

```
par(mfrow=c(2,3))
residualPlots(m6,id=list(method=cooks.distance(m6),n=10))
```



	Test stat	Pr(> Test stat)
mileage	-2.4985	0.0125 *
tax	6.2939	3.371e-10 ***
mpg	12.8930	< 2.2e-16 ***
age	-6.9379	4.510e-12 ***
Tukey test	-1.4810	0.1386

Signif. codes:	0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1	

Podemos apreciar que hay tres individuos que aparecen constantemente fuera de las nubes de puntos, los 20688, 19848 y 40774.

Vamos a proceder a eliminar los elementos que aparecían en los plots anteriores constantemente alejados de las nubes de puntos así como los multivariant outliers.

```
df2 <- df[!df$mout=="YesMOut",]
df2 <- df2[row.names(df2)!="19848",]
df2 <- df2[row.names(df2)!="40774",]
df2 <- df2[row.names(df2)!="20688",]
```

Procederemos a replantear el modelo excluyendo las observaciones que hemos comentado previamente.

```
m7<-update(m6,data=df2)
summary(m7)
```

Call:
`lm(formula = log(price) ~ mileage + tax + mpg + age, data = df2)`

Residuals:

Min	1Q	Median	3Q	Max
-1.31243	-0.17147	0.02243	0.19090	0.82498

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	1.080e+01	6.421e-02	168.165	< 2e-16 ***
mileage	-2.203e-06	3.626e-07	-6.076	1.32e-09 ***
tax	1.794e-03	3.689e-04	4.862	1.20e-06 ***
mpg	-1.414e-02	4.465e-04	-31.669	< 2e-16 ***
age	-1.078e-01	3.797e-03	-28.393	< 2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2809 on 4803 degrees of freedom

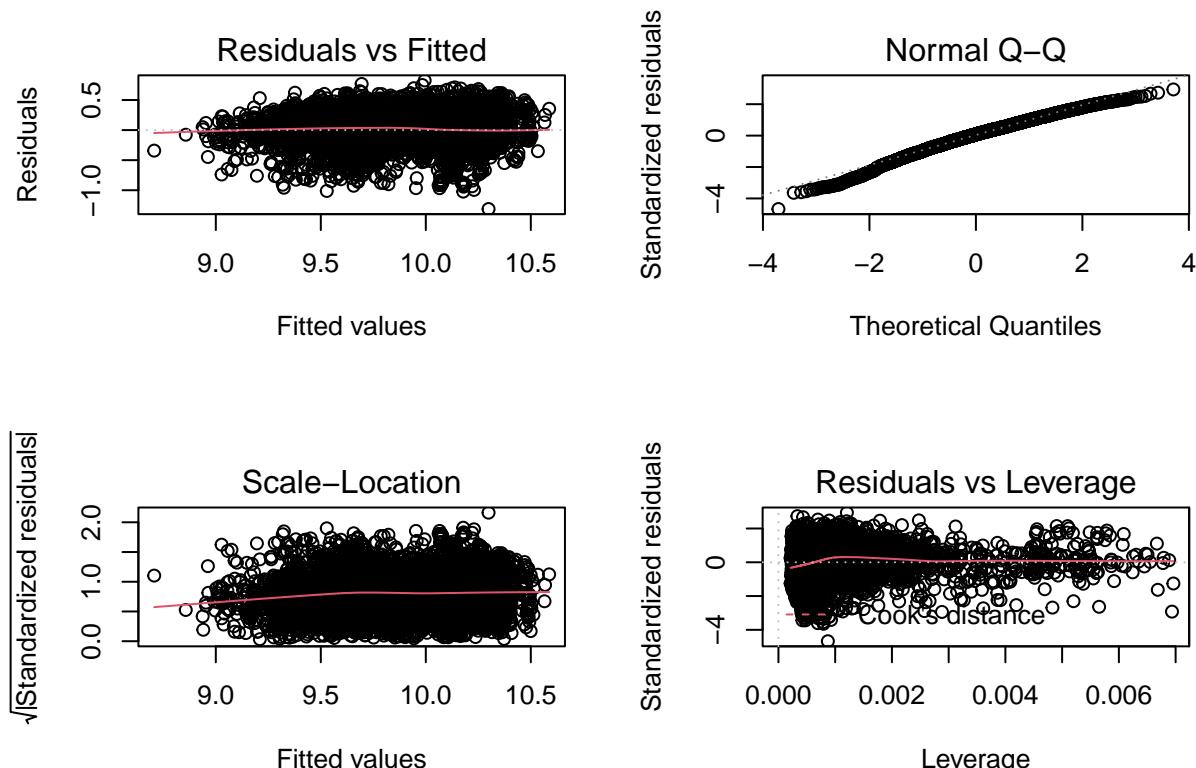
Multiple R-squared: 0.5801, Adjusted R-squared: 0.5798

F-statistic: 1659 on 4 and 4803 DF, p-value: < 2.2e-16

Se puede apreciar como aumentan la explicabilidad pero no hay grandes cambios en la relevancia de los coeficientes.

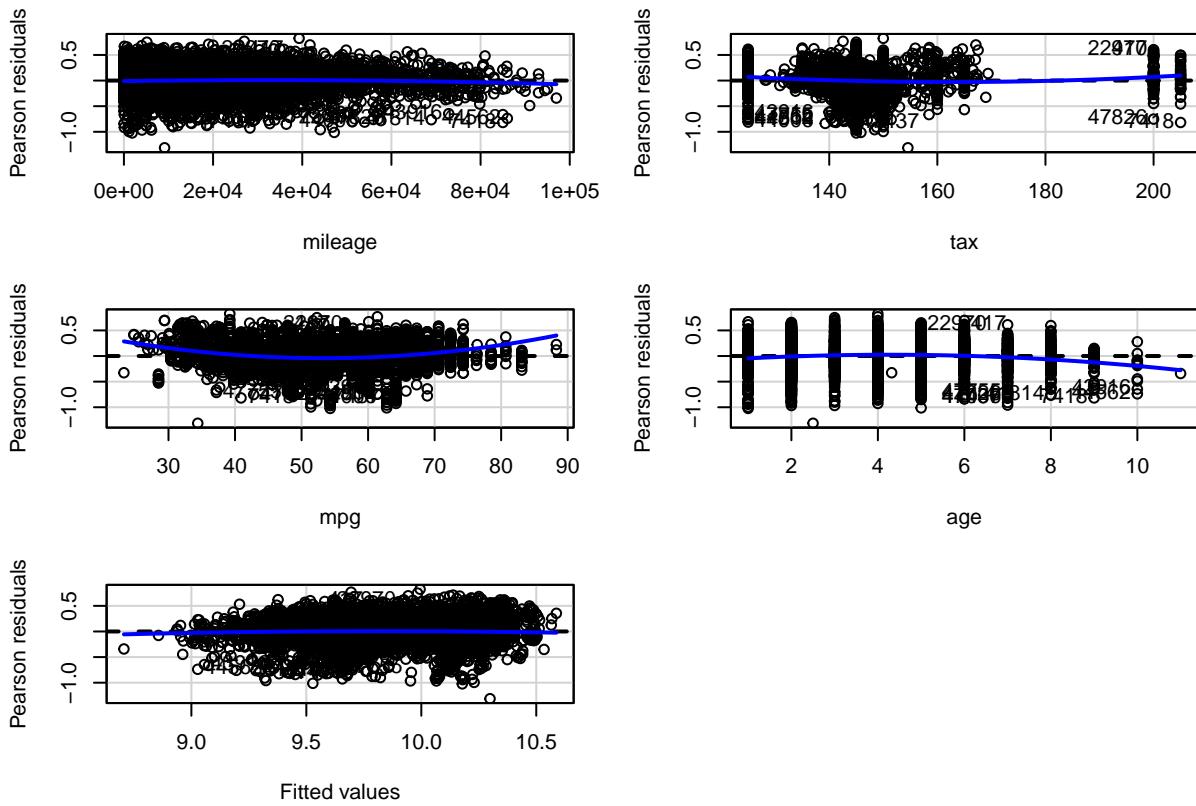
En los siguientes plots podemos ver como este pequeño cambio en el dataset hemos eliminado los individuos que constantemente se desviaban de las nubes de puntos. Este hecho se puede ver especialmente en el gráfico Normal Q-Q, donde hay un mejor ajuste a la recta para los cuantiles inferiores.

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



Si nos fijamos en los gráficos de los residuos, podemos ver como siguen existiendo algunos desajustes en las rectas, sobretodo para las variables mpg y age.

```
par(mfrow=c(2,3))
residualPlots(m7,id=list(method=cooks.distance(m7),n=10))
```



	Test stat	Pr(> Test stat)
mileage	-2.2944	0.02181 *
tax	6.5138	8.078e-11 ***
mpg	12.9057	< 2.2e-16 ***
age	-6.9296	4.780e-12 ***
Tukey test	-1.5075	0.13168

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Si realizamos el test de Breusch-Pagan contra la heteroscedasticidad de nuestro modelo, obtenemos un p-valor de 0.0005, de modo que podemos rechazar la H_0 y confirmar que nuestro modelo es homocedástico.

```
library(lmtest)
```

Loading required package: zoo

Attaching package: 'zoo'

The following objects are masked from 'package:base':

```
as.Date, as.Date.numeric
```

Attaching package: 'lmtest'

The following object is masked from 'package:epiDisplay':

```
lrtest
```

```
bptest(m7)
```

```
studentized Breusch-Pagan test  
data: m7  
BP = 30.178, df = 4, p-value = 4.502e-06
```

Vamos a proceder a mostrar los boxplots de los valores R-student, Hat y distancias de Cook de las observaciones del modelo.

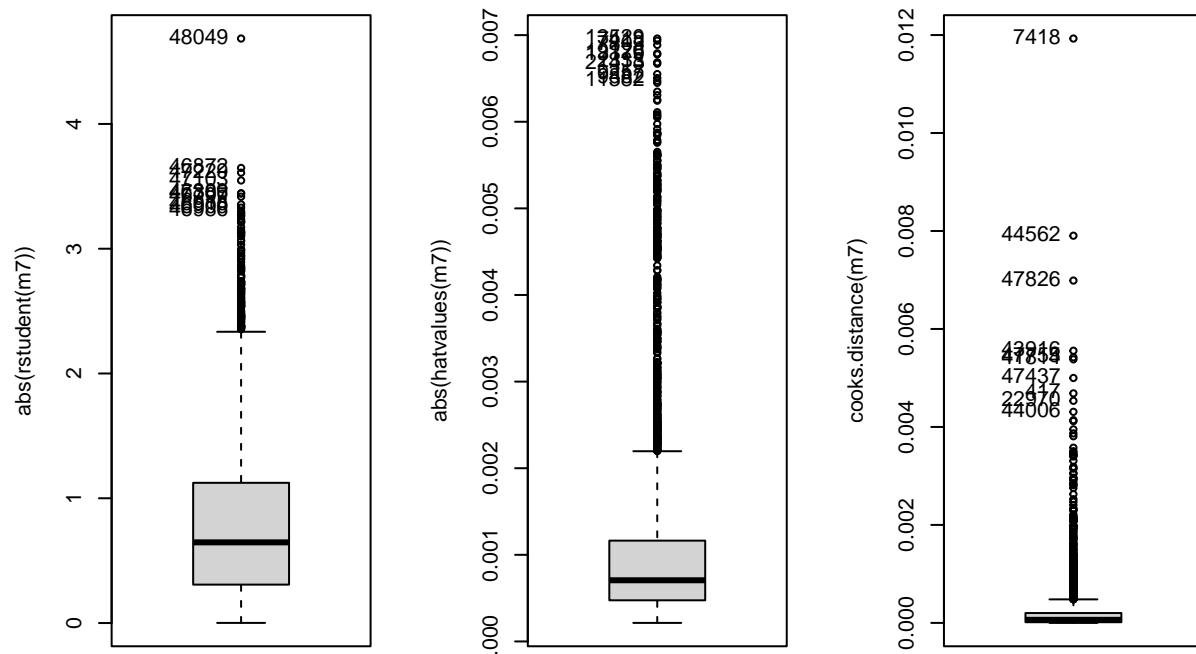
```
par(mfrow=c(1,3))  
Boxplot(abs(rstudent(m7)), id=list(labels=row.names(df2)))
```

```
[1] "48049" "46872" "47229" "47103" "46809" "47386" "46797" "46918" "46909"  
[10] "46986"
```

```
Boxplot(abs(hatvalues(m7)), id=list(labels=row.names(df2)))
```

```
[1] "13529" "7418" "7803" "19126" "13379" "2338" "21413" "9257" "9882"  
[10] "11382"
```

```
Boxplot(cooks.distance(m7), id=list(labels=row.names(df2)))
```



```
[1] "7418" "44562" "47826" "43916" "47755" "41814" "47437" "417" "22970"  
[10] "44006"
```

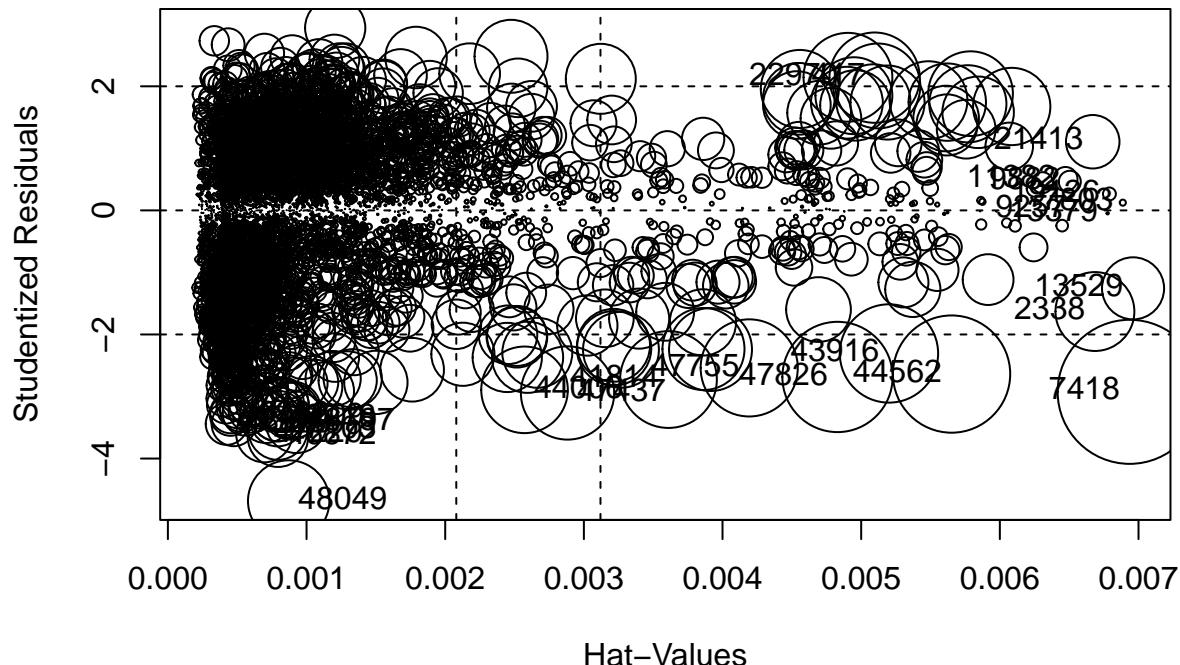
Con estos gráficos podemos detectar los valores para los cuales se rompe la cadena de puntos y podemos categorizar como outliers.

```
stu_out <- which(abs(rstudent(m7))>3.7);  
cook_out <- which(abs(cooks.distance(m7))>0.0065);  
hat_out <- which(abs(hatvalues(m7))>0.007);  
  
outs<-unique(c(stu_out,cook_out,hat_out));outs
```

```
[1] 4648 737 4317 4634
```

Si analizamos el gráfico de influencias, podemos ver como no existe una distribución aglomerada, tal vez en los individuos con valores de Hat muy bajos, pero en general existe basante dispersión.

```
par(mfrow=c(1,1));
outs2 <- influencePlot(m7, id=list(n=10));
```



```
outs2 <- labels(outs2)[[1]];
outs2 <- as.numeric(outs2);
outs3 <- unique(c(outs,outs2));outs3
```

```
[1] 4648 737 4317 4634 417 2338 7418 7803 9257 9882 11382 13379
[13] 13529 19126 21413 22970 41814 43916 44006 44562 46797 46809 46872 46909
[25] 46918 46986 47103 47229 47386 47437 47755 47826 48049
```

Vamos a terminar este análisis generando un modelo excluyendo los individuos que hemos detectado como outliers.

```
m8 <- update(m7, data=df2[-outs3,])
summary(m8)
```

Call:
lm(formula = log(price) ~ mileage + tax + mpg + age, data = df2[-outs3,
])

Residuals:
Min 1Q Median 3Q Max
-1.02127 -0.17225 0.02214 0.19072 0.82166

Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.079e+01 6.420e-02 168.064 < 2e-16 ***
mileage -2.176e-06 3.617e-07 -6.017 1.90e-09 ***

```

tax           1.873e-03  3.692e-04   5.074 4.04e-07 ***
mpg          -1.426e-02  4.454e-04  -32.009 < 2e-16 ***
age          -1.072e-01  3.785e-03  -28.335 < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Residual standard error: 0.2798 on 4797 degrees of freedom
Multiple R-squared: 0.5813, Adjusted R-squared: 0.581
F-statistic: 1665 on 4 and 4797 DF, p-value: < 2.2e-16

Vemos que la explicabilidad del modelo final es del 57,47%.

14.2 Factores

Vamos a empezar añadiendo un solo factor al modelo. En este caso empezaremos con los factores que determinamos que eran más influyentes en el análisis MCA de la entrega anterior. Los tres factores más relevantes eran f.price, transmission y fuelType. No añadiremos f.price ya que es un factor generado a partir de nuestra variable target.

De momento empezamos añadiendo fuelType.

```
m10<- update(m8, ~.+fuelType)
summary(m10)
```

Call:
`lm(formula = log(price) ~ mileage + tax + mpg + age + fuelType,
 data = df2[-outs3,])`

Residuals:

Min	1Q	Median	3Q	Max
-0.85784	-0.12569	0.00999	0.14887	0.74534

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	1.149e+01	5.241e-02	219.325	< 2e-16 ***
mileage	-4.550e-06	2.887e-07	-15.762	< 2e-16 ***
tax	8.432e-04	2.921e-04	2.886	0.00391 **
mpg	-2.219e-02	3.811e-04	-58.221	< 2e-16 ***
age	-8.471e-02	3.015e-03	-28.097	< 2e-16 ***
fuelTypef.Fuel-Petrol	-3.911e-01	7.299e-03	-53.592	< 2e-16 ***
fuelTypef.Fuel-Hybrid	6.008e-02	3.185e-02	1.887	0.05929 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2207 on 4795 degrees of freedom
Multiple R-squared: 0.7396, Adjusted R-squared: 0.7393
F-statistic: 2270 on 6 and 4795 DF, p-value: < 2.2e-16

Como podemos ver ya a simple vista, el R-squared del modelo ha aumentado significativamente, indicando mayor explicabilidad.

Si analizamos los vif, vemos que no ha habido cambios significativos respecto al modelo anterior. Los vifs se mantienen en valores inferiores a 5.

```
vif(m10)
```

	GVIF	Df	GVIF^(1/(2*Df))
mileage	3.048469	1	1.745987
tax	1.184422	1	1.088311
mpg	1.649388	1	1.284285
age	3.066970	1	1.751277
fuelType	1.267793	2	1.061114

Con el test anova podemos ver como todas las variables mantienen su significatividad.

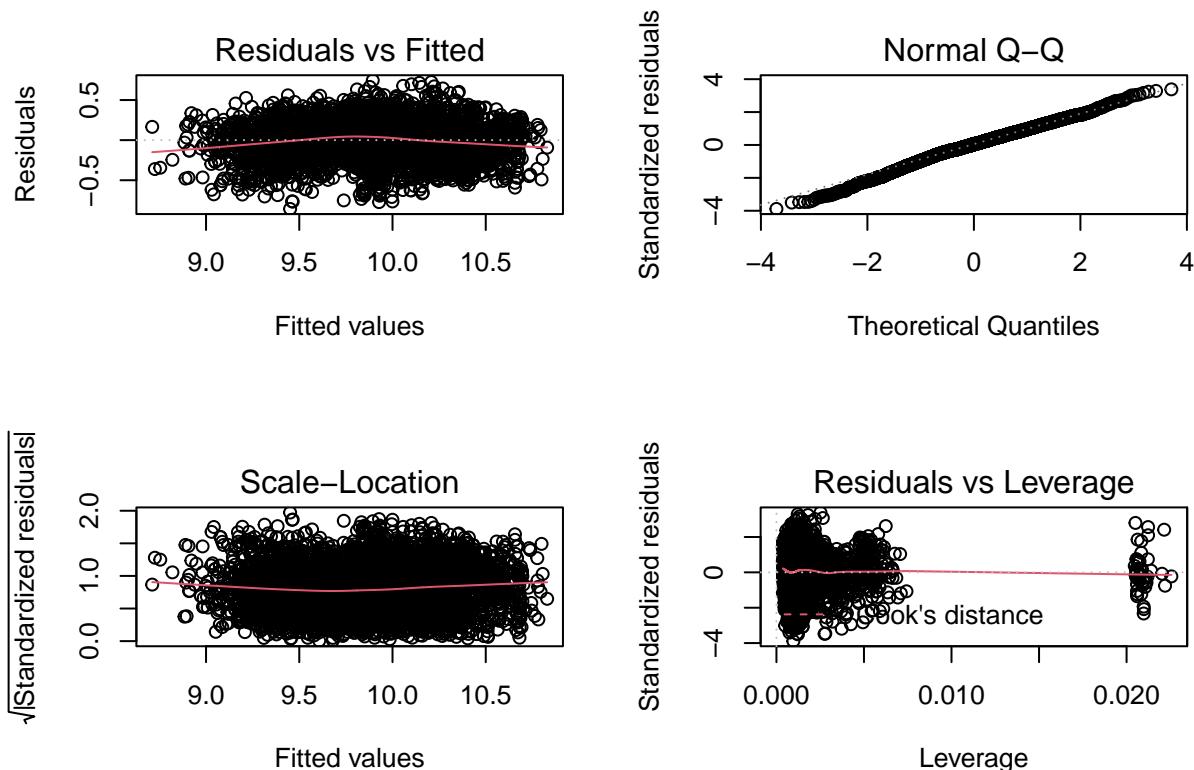
```
Anova(m10)
```

Anova Table (Type II tests)

```
Response: log(price)
          Sum Sq   Df F value    Pr(>F)
mileage    12.099   1 248.4426 < 2.2e-16 ***
tax        0.406   1   8.3311  0.003915 **
mpg       165.073   1 3389.6813 < 2.2e-16 ***
age        38.444   1   789.4280 < 2.2e-16 ***
fuelType   141.924   2 1457.1566 < 2.2e-16 ***
Residuals 233.511 4795
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Si analizmos los plots del modelo, podemos ver como el modelo parece haber perdido homocedasticidad y normalidad en los extremos, y la distribución de las distancias de Cook es algo peculiar, generando diversas acumulaciones de puntos.

```
par(mfrow=c(2,2))
plot(m10, id.n=0)
```



Vamos a proceder a incorporar también el factor transmission.

```
m11 <- update(m10, ~.+transmission)
summary(m11)
```

Call:

```
lm(formula = log(price) ~ mileage + tax + mpg + age + fuelType +
  transmission, data = df2[-outs3, ])
```

Residuals:

```

Min      1Q   Median     3Q    Max
-0.77564 -0.12050  0.00072  0.13119  0.74567

```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	1.114e+01	4.772e-02	233.347	<2e-16 ***
mileage	-4.309e-06	2.569e-07	-16.772	<2e-16 ***
tax	4.196e-04	2.600e-04	1.613	0.107
mpg	-1.824e-02	3.583e-04	-50.912	<2e-16 ***
age	-7.978e-02	2.685e-03	-29.708	<2e-16 ***
fuelTypef.Fuel-Petrol	-3.050e-01	6.961e-03	-43.815	<2e-16 ***
fuelTypef.Fuel-Hybrid	3.877e-03	2.836e-02	0.137	0.891
transmissionf.Trans-SemiAuto	2.449e-01	7.191e-03	34.059	<2e-16 ***
transmissionf.Trans-Automatic	2.207e-01	8.040e-03	27.458	<2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1962 on 4793 degrees of freedom

Multiple R-squared: 0.7942, Adjusted R-squared: 0.7938

F-statistic: 2312 on 8 and 4793 DF, p-value: < 2.2e-16

Con el aumento del R-squared, podemos apreciar un nuevo aumento en la explicabilidad del modelo.

El test de anova nos indica que el nuevo modelo es preferible al anterior.

```
anova(m10,m11)
```

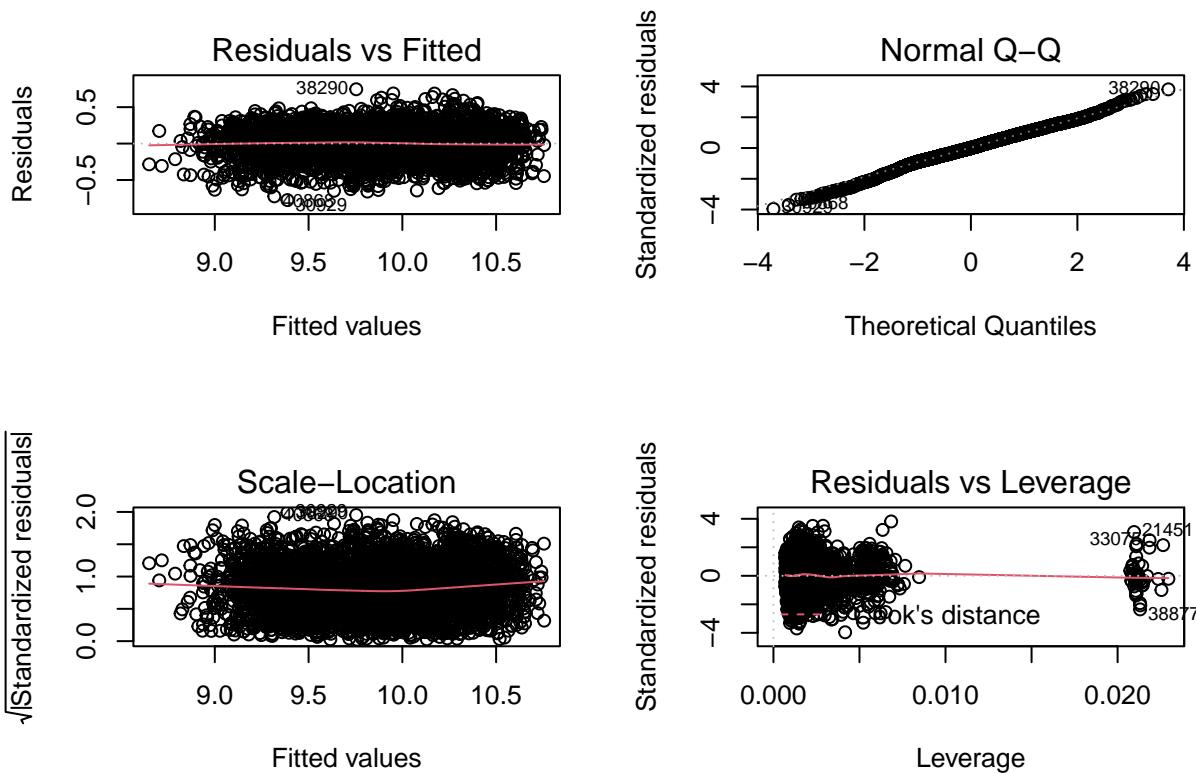
Analysis of Variance Table

Model 1: log(price) ~ mileage + tax + mpg + age + fuelType
Model 2: log(price) ~ mileage + tax + mpg + age + fuelType + transmission
Res.Df RSS Df Sum of Sq F Pr(>F)
1 4795 233.51
2 4793 184.56 2 48.952 635.64 < 2.2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Si analizamos los plots de este modelo, podemos ver como no aparecen cambios significativos. Tal vez podemos apreciar algo más de normalidad, pero la homocedasticidad y las distancias de Cook no parecen haber cambiado mucho.

```
par(mfrow=c(2,2))
plot(m11)
```



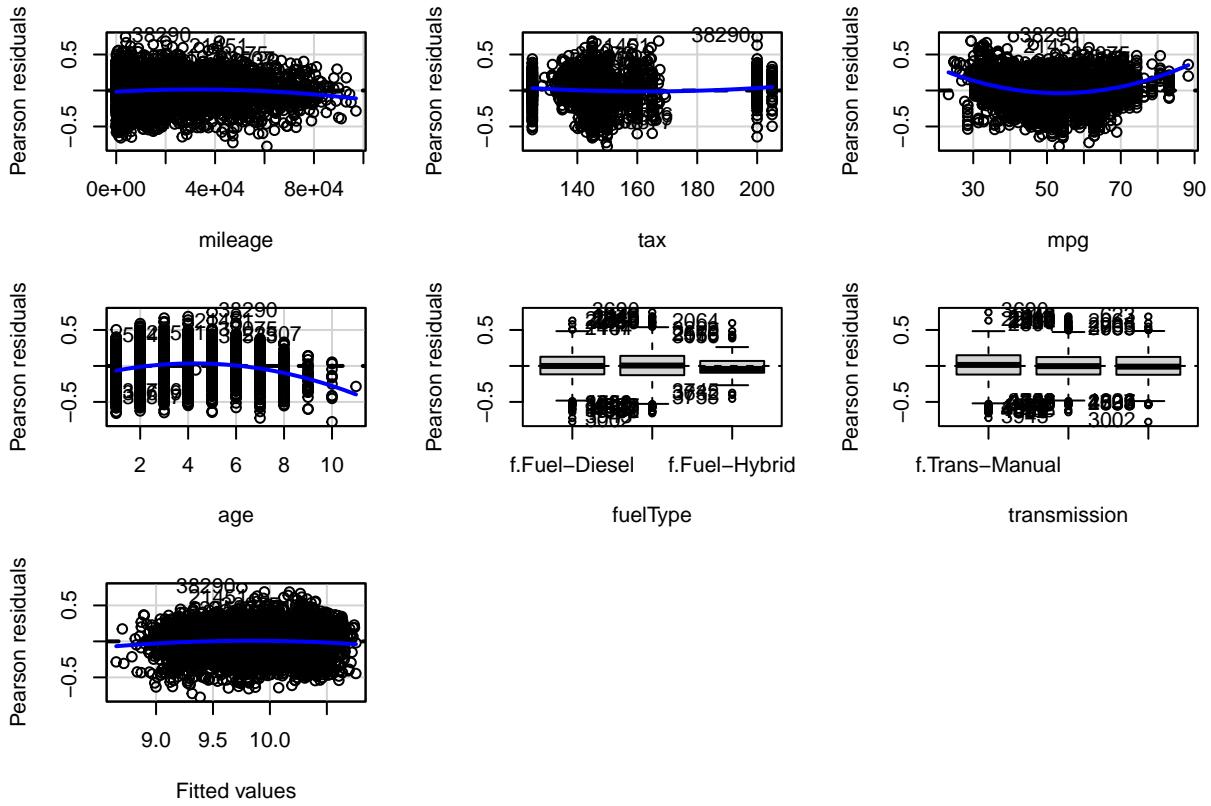
En lo que se refiere a los vifs, no hay cambios significativos, indicando que no aparece co-linealidad en las variables.

```
vif(m11)
```

	GVIF	Df	GVIF ^{(1/(2*Df))}
mileage	3.053952	1	1.747556
tax	1.186901	1	1.089450
mpg	1.843049	1	1.357589
age	3.077388	1	1.754249
fuelType	1.468859	2	1.100893
transmission	1.285463	2	1.064792

Si analizamos los plots de los residuos, para las variables numéricas no apreciamos grandes cambios,

```
par(mfrow=c(1,1))
residualPlots(m11,id=list(method=cooks.distance(m11),n=10))
```



```

Test stat Pr(>|Test stat|)
mileage      -5.2295    1.772e-07 ***
tax          4.4459    8.950e-06 ***
mpg          16.4618   < 2.2e-16 ***
age         -14.4041   < 2.2e-16 ***
fuelType
transmission
Tukey test   -3.4650    0.0005303 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

14.3 Interacciones

14.3.1 Interacciones entre factores

Vamos a proceder a incorporar la interacción entre los factores transmission y fuelType.

```
m13 <- update(m11, ~.+transmission*fuelType)
summary(m13)
```

```
Call:
lm(formula = log(price) ~ mileage + tax + mpg + age + fuelType +
    transmission + fuelType:transmission, data = df2[-outs3,
    ])
```

```
Residuals:
    Min      1Q      Median      3Q      Max 
-0.76008 -0.11925  0.00092  0.13340  0.76630
```

```
Coefficients: (1 not defined because of singularities)
              Estimate Std. Error
(Intercept) 1.115e+01 4.746e-02
mileage     -4.463e-06 2.564e-07
tax          5.024e-04 2.587e-04
```

```

mpg                         -1.805e-02  3.572e-04
age                          -7.916e-02  2.671e-03
fuelTypef.Fuel-Petrol        -3.619e-01  1.009e-02
fuelTypef.Fuel-Hybrid         3.684e-02  4.215e-02
transmissionf.Trans-SemiAuto 2.026e-01  9.433e-03
transmissionf.Trans-Automatic 1.764e-01  1.005e-02
fuelTypef.Fuel-Petrol:transmissionf.Trans-SemiAuto 8.838e-02  1.341e-02
fuelTypef.Fuel-Hybrid:transmissionf.Trans-SemiAuto -3.633e-02  5.674e-02
fuelTypef.Fuel-Petrol:transmissionf.Trans-Automatic 1.031e-01  1.555e-02
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic      NA       NA
t value Pr(>|t|)
(Intercept)                234.868 < 2e-16 ***
mileage                     -17.405 < 2e-16 ***
tax                           1.942  0.0522 .
mpg                          -50.541 < 2e-16 ***
age                          -29.638 < 2e-16 ***
fuelTypef.Fuel-Petrol        -35.872 < 2e-16 ***
fuelTypef.Fuel-Hybrid          0.874  0.3822
transmissionf.Trans-SemiAuto 21.480 < 2e-16 ***
transmissionf.Trans-Automatic 17.545 < 2e-16 ***
fuelTypef.Fuel-Petrol:transmissionf.Trans-SemiAuto 6.592  4.81e-11 ***
fuelTypef.Fuel-Hybrid:transmissionf.Trans-SemiAuto -0.640  0.5221
fuelTypef.Fuel-Petrol:transmissionf.Trans-Automatic 6.631  3.70e-11 ***
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic      NA       NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Residual standard error: 0.195 on 4790 degrees of freedom
 Multiple R-squared: 0.7968, Adjusted R-squared: 0.7963
 F-statistic: 1707 on 11 and 4790 DF, p-value: < 2.2e-16

Podemos ver como el R-squared apenas aumenta y se mantiene la explicabilidad de las variables.

Si realizamos el test de anova, podemos ver como, a pensar de que a partir del summary los modelos parecen similares, no lo son, y en realidad el nuevo modelo es mejor que el anterior.

```
anova(m11,m13)
```

Analysis of Variance Table

```

Model 1: log(price) ~ mileage + tax + mpg + age + fuelType + transmission
Model 2: log(price) ~ mileage + tax + mpg + age + fuelType + transmission +
  fuelType:transmission
  Res.Df   RSS Df Sum of Sq    F    Pr(>F)
1   4793 184.56
2   4790 182.22  3     2.334 20.45 3.651e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

14.3.2 Interacciones Factor-Numéricas

Vamos a probar a añadir las interacciones de la variable numérica age y el factor transmission. Podemos ver como los dos modelos no son equivalentes, y según el test de anova, el nuevo modelo es más completo.

```
m14 <- update(m13, ~.+age*transmission)
anova(m13,m14)
```

Analysis of Variance Table

```

Model 1: log(price) ~ mileage + tax + mpg + age + fuelType + transmission +
  fuelType:transmission
Model 2: log(price) ~ mileage + tax + mpg + age + fuelType + transmission +
  fuelType:transmission

```

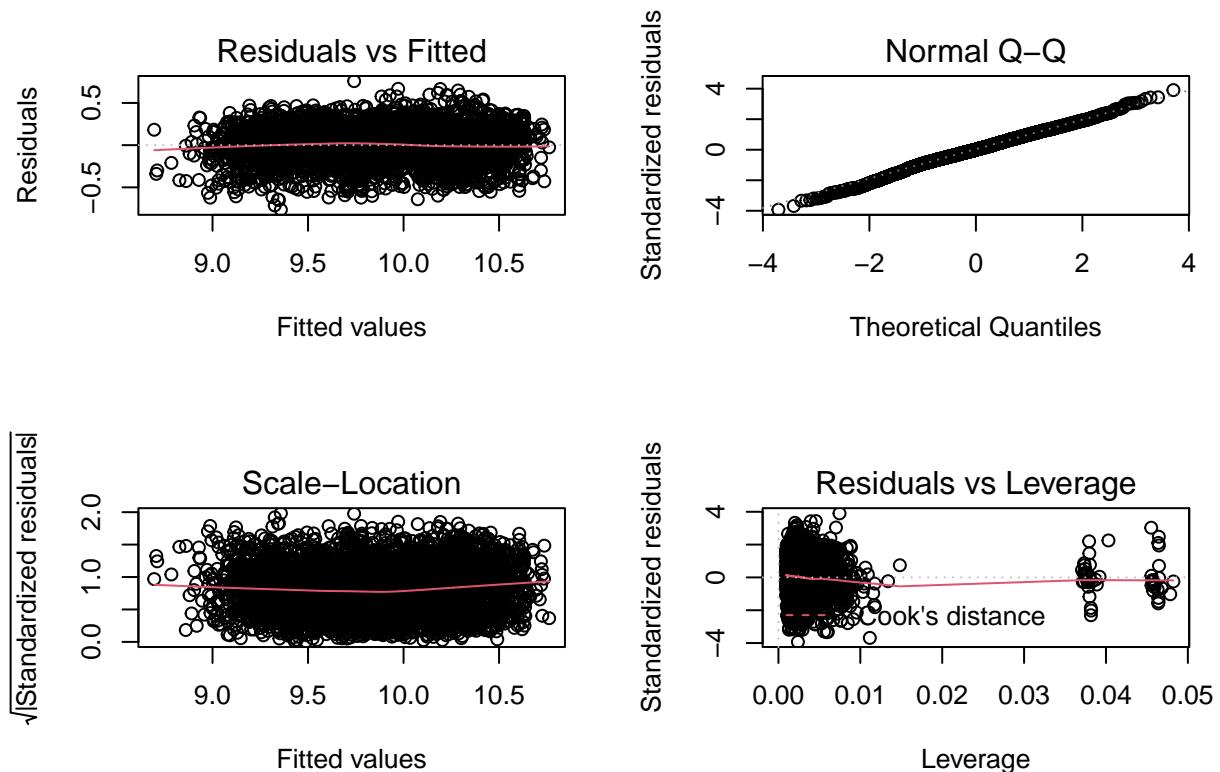
```

fuelType:transmission + age:transmission
Res.Df   RSS Df Sum of Sq      F Pr(>F)
1    4790 182.22
2    4788 181.89  2   0.33656 4.4297 0.01197 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Si analizamos los plots del modelo, podemos ver como existe homocedasticidad y aparece bastante normalidad.

```
par(mfrow=c(2,2))
plot(m14, id.n=0)
```

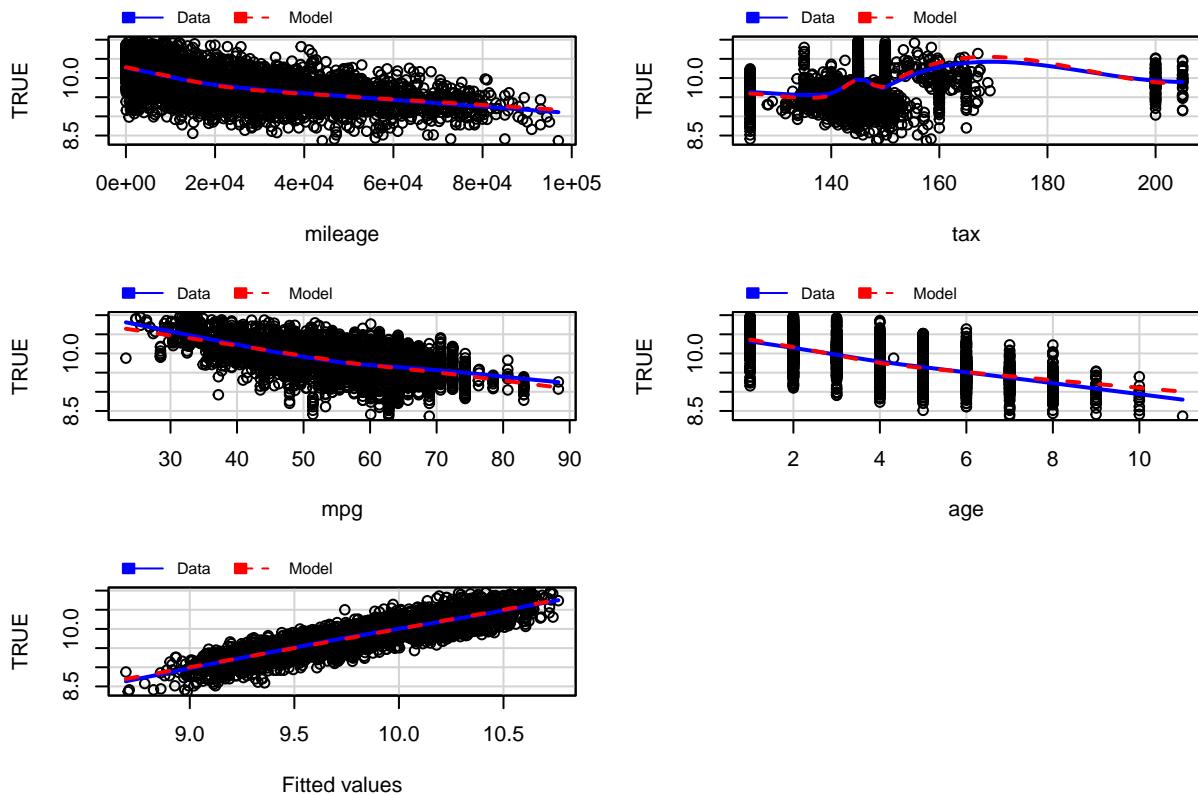


Si analizamos los plots de los residuos, podemos ver como el ajuste de las regresiones entre el modelo y los datos es muy preciso.

```
marginalModelPlots(m14)
```

Warning in mmpls(...): Interactions and/or factors skipped

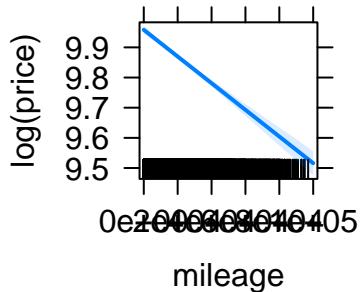
Marginal Model Plots



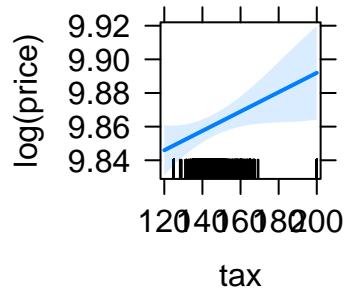
Podemos ver claramente fuerte relación entre la variable mpg y nuestro target numérico.

```
library(effects);
plot(allEffects(m14))
```

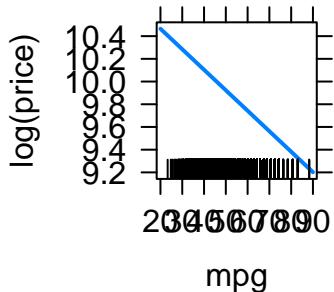
mileage effect plot



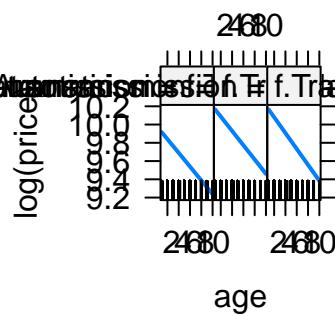
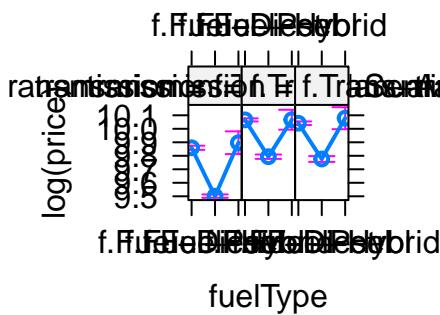
tax effect plot



mpg effect plot



type*transmission effect plot



Además, si nos fijamos en la interacciones de age y transmission, podemos ver como para todos los tipos de transmisión se respeta la relación inversa con el precio: cuantos más años tiene el coche, más barato es, y viceversa. Por el contrario, en el caso de fuelType, podemos ver como los coches de gasolina son más baratos que los diesel o híbridos.

Si hacemos una rápida búsqueda en internet (<https://www.motor.mapfre.es/consejos-practicos/consejos-para-ahorrar/diesel-o-gasolina/>) podemos ver como esto este hecho cuadra con la realidad.

El problema aparece cuando intentamos aplicar la función vif para comprobar la no co-linealidad de las variables del modelo, ya que salta un error que nos indica que si que hay variables co-lineales. Sin embargo, si realizamos un análisis de las varianzas, todas las variables parecen significativas. Las que menos significación adquieren son tax y la interacción entre age y transmission.

```
#vif(m14)
```

```
Anova(m14);
```

```
Note: model has aliased coefficients
      sums of squares computed by model comparison
```

Anova Table (Type II tests)

Response: log(price)

	Sum Sq	Df	F value	Pr(>F)		
mileage	11.265	1	296.5463	< 2.2e-16 ***		
tax	0.185	1	4.8705	0.02737 *		
mpg	97.100	1	2556.0470	< 2.2e-16 ***		
age	33.417	1	879.6713	< 2.2e-16 ***		
fuelType	73.875	2	972.3323	< 2.2e-16 ***		
transmission	48.952	2	644.3048	< 2.2e-16 ***		
fuelType:transmission	1.956	3	17.1663	4.371e-11 ***		
age:transmission	0.337	2	4.4297	0.01197 *		
Residuals	181.888	4788				

Signif. codes:	0 '***'	0.001 '**'	0.01 '*'	0.05 '.'	0.1 ' '	1

Vamos a proceder a aplicar la función step para tratar de eliminar esta co-linealidad.

```
m15 <- step( m14, k=log(nrow(df2)))
```

```
Start: AIC=-15600.15
log(price) ~ mileage + tax + mpg + age + fuelType + transmission +
  fuelType:transmission + age:transmission
```

	Df	Sum of Sq	RSS	AIC
- age:transmission	2	0.337	182.22	-15608
- tax	1	0.185	182.07	-15604
<none>			181.89	-15600
- fuelType:transmission	3	1.956	183.84	-15574
- mileage	1	11.265	193.15	-15320
- mpg	1	97.100	278.99	-13554

```
Step: AIC=-15608.23
```

```
log(price) ~ mileage + tax + mpg + age + fuelType + transmission +
  fuelType:transmission
```

	Df	Sum of Sq	RSS	AIC
- tax	1	0.143	182.37	-15613
<none>			182.22	-15608
- fuelType:transmission	3	2.334	184.56	-15573
- mileage	1	11.525	193.75	-15322
- age	1	33.417	215.64	-14808
- mpg	1	97.176	279.40	-13564

```
Step: AIC=-15612.93
```

```
log(price) ~ mileage + mpg + age + fuelType + transmission +
  fuelType:transmission
```

	Df	Sum of Sq	RSS	AIC
--	----	-----------	-----	-----

```

<none>                               182.37 -15613
- fuelType:transmission  3      2.291 184.66 -15578
- mileage                 1     11.383 193.75 -15331
- age                     1     33.324 215.69 -14816
- mpg                     1    111.861 294.23 -13324

```

```
#vif(m15)
```

Si realizamos el test de anova, podemos ver como el nuevo modelo no es equivalente al anterior, y de hecho, sigue apareciendo co-linealidad en las variables, de modo que el m14 es preferible.

```
anova(m15,m14)
```

Analysis of Variance Table

```

Model 1: log(price) ~ mileage + mpg + age + fuelType + transmission +
          fuelType:transmission
Model 2: log(price) ~ mileage + tax + mpg + age + fuelType + transmission +
          fuelType:transmission + age:transmission
Res.Df   RSS Df Sum of Sq    F   Pr(>F)
1     4791 182.37
2     4788 181.89  3    0.48002 4.212 0.005533 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Si aplicamos la función alias, podemos ver que esta colinealidad parece generada por la aparición de las variables transmission y fuelType. Supongo que es debido a que los coches híbridos suelen tener transmisión automática, de modo que el conjunto Hybrid:SemiAuto queda vacío, mientras que hybrid y hybrid&Automatic son el mismo conjunto.

```
alias(m14)
```

```

Model :
log(price) ~ mileage + tax + mpg + age + fuelType + transmission +
          fuelType:transmission + age:transmission

Complete :
                                              (Intercept) mileage tax mpg
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic 0           0   0   0
                                                    age fuelTypef.Fuel-Petrol
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic 0           0
                                                    fuelTypef.Fuel-Hybrid
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic 1
                                                    transmissionf.Trans-SemiAuto
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic 0
                                                    transmissionf.Trans-Automatic
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic 0
                                                    fuelTypef.Fuel-Petrol:transmissionf.Trans-SemiAuto
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic 0
                                                    fuelTypef.Fuel-Hybrid:transmissionf.Trans-SemiAuto
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic -1
                                                    fuelTypef.Fuel-Petrol:transmissionf.Trans-Automatic
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic 0
                                                    age:transmissionf.Trans-SemiAuto
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic 0
                                                    age:transmissionf.Trans-Automatic
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic 0

```

```
summary(m14)
```

Call:

```

lm(formula = log(price) ~ mileage + tax + mpg + age + fuelType +
   transmission + fuelType:transmission + age:transmission,
   data = df2[-outs3, ])

Residuals:
    Min      1Q  Median      3Q     Max 
-0.76486 -0.11866  0.00022  0.13405  0.75836 

Coefficients: (1 not defined because of singularities)
                                         Estimate Std. Error
(Intercept)                         1.112e+01  4.949e-02
mileage                                -4.423e-06  2.568e-07
tax                                     5.750e-04  2.605e-04
mpg                                      -1.805e-02  3.569e-04
age                                      -7.531e-02  3.321e-03
fuelTypef.Fuel-Petrol                  -3.581e-01  1.023e-02
fuelTypef.Fuel-Hybrid                  3.751e-02  4.212e-02
transmissionf.Trans-SemiAuto          2.181e-01  1.792e-02
transmissionf.Trans-Automatic         2.250e-01  1.957e-02
fuelTypef.Fuel-Petrol:transmissionf.Trans-SemiAuto 8.535e-02  1.369e-02
fuelTypef.Fuel-Hybrid:transmissionf.Trans-SemiAuto -3.635e-02  5.671e-02
fuelTypef.Fuel-Petrol:transmissionf.Trans-Automatic 9.328e-02  1.590e-02
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic      NA      NA
age:transmissionf.Trans-SemiAuto       -3.060e-03  3.772e-03
age:transmissionf.Trans-Automatic     -1.166e-02  3.989e-03
t value Pr(>|t|) 
(Intercept)                      224.593 < 2e-16 ***
mileage                           -17.221 < 2e-16 ***
tax                               2.207  0.02737 *
mpg                               -50.557 < 2e-16 ***
age                               -22.675 < 2e-16 ***
fuelTypef.Fuel-Petrol              -35.014 < 2e-16 ***
fuelTypef.Fuel-Hybrid               0.890  0.37330
transmissionf.Trans-SemiAuto       12.166 < 2e-16 ***
transmissionf.Trans-Automatic     11.496 < 2e-16 ***
fuelTypef.Fuel-Petrol:transmissionf.Trans-SemiAuto 6.236  4.89e-10 ***
fuelTypef.Fuel-Hybrid:transmissionf.Trans-SemiAuto -0.641  0.52156
fuelTypef.Fuel-Petrol:transmissionf.Trans-Automatic 5.867  4.72e-09 ***
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic      NA      NA
age:transmissionf.Trans-SemiAuto   -0.811  0.41730
age:transmissionf.Trans-Automatic  -2.924  0.00347 **

---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1949 on 4788 degrees of freedom
Multiple R-squared:  0.7972,    Adjusted R-squared:  0.7966
F-statistic: 1448 on 13 and 4788 DF,  p-value: < 2.2e-16

```

Finalmente, si echamos un último vistazo al modelo, podemos ver que hemos conseguido aglomerar una explicabilidad del 80%.

15 Modelo de regresión Binaria

Para plantear el modelo de regresión binaria, vamos a proceder primero a separar nuestro dataframe en dos subconjuntos de entrenamiento y de validación.

```

llwork <- sample(1:nrow(df2),round(0.80*nrow(df2),0))

df_train <- df2[llwork,]
df_test <- df2[-llwork,]

```

15.1 Variables numéricas

Vamos a empezar el proceso de generación del modelo de regresión binaria incorporando las variables numéricas y planteando un modelo inicial.

```
m20<-glm(Audi~mileage+tax+mpg+age,family="binomial",data=df_train)
summary(m20)
```

Call:

```
glm(formula = Audi ~ mileage + tax + mpg + age, family = "binomial",
     data = df_train)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-1.1773	-0.7192	-0.6253	-0.4978	2.1495

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	3.090e-01	5.820e-01	0.531	0.5955
mileage	6.565e-06	3.527e-06	1.861	0.0627 .
tax	-3.616e-04	3.263e-03	-0.111	0.9118
mpg	-3.842e-02	4.505e-03	-8.528	<2e-16 ***
age	7.988e-02	3.687e-02	2.167	0.0302 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

```
Null deviance: 3972.7 on 3845 degrees of freedom
Residual deviance: 3878.7 on 3841 degrees of freedom
AIC: 3888.7
```

Number of Fisher Scoring iterations: 4

Como podemos ver, la variable más significativa de nuestro modelo inicial parece ser mpg, mientras que mileage o tax parecen no ser significativas. También podemos ver como el valor AIC es 3810.1, nuestro objetivo será reducirlo.

Si miramos los valores vif de nuestro modelo, podemos ver como no reflejan co-linealidad entre las variables.

```
vif(m20)
```

mileage	tax	mpg	age
3.162933	1.216322	1.411785	3.224172

Vamos a plantear un nuevo modelo que incluya solo las variables mpg y age, que son las que aparecían como significativas en el modelo anterior. Si realizamos el test anova de los dos modelos, podemos ver como los modelos parecen equivalentes, de modo que será preferible trabajar con el más sencillo.

```
m21 <- glm(Audi~ mpg + age,family="binomial",data=df_train)
anova(m21,m20,test = "LR");
```

Analysis of Deviance Table

Model 1: Audi ~ mpg + age	Model 2: Audi ~ mileage + tax + mpg + age			
Resid. Df	Resid. Dev	Df	Deviance	Pr(>Chi)
1	3843	3882.2		
2	3841	3878.7	2	3.4464 0.1785

Probaremos a añadir los polinomios de segundo grado de las variables del modelo que acabamos de generar. El test de anova nos vuelve a indicar que los modelos son equivalentes, de modo que continuaremos trabajando con el más simple.

```
m22 <- glm(Audi ~ poly(mpg, 2) + poly(age, 2), family="binomial", data=df_train)
anova(m21,m22, test = "LR")
```

Analysis of Deviance Table

```
Model 1: Audi ~ mpg + age
Model 2: Audi ~ poly(mpg, 2) + poly(age, 2)
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1      3843    3882.2
2      3841    3879.4  2     2.7707   0.2502
```

Finalmente, este es el modelo con el que continuaremos trabajando, ya que es el más sencillo y equivalente a otros modelos más complejos que hemos planteado.

```
summary(m21)
```

Call:

```
glm(formula = Audi ~ mpg + age, family = "binomial", data = df_train)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-1.1330	-0.7204	-0.6247	-0.5071	2.1498

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	0.155917	0.199967	0.780	0.436
mpg	-0.037504	0.004143	-9.052	< 2e-16 ***
age	0.132835	0.022401	5.930	3.03e-09 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

```
Null deviance: 3972.7 on 3845 degrees of freedom
Residual deviance: 3882.2 on 3843 degrees of freedom
AIC: 3888.2
```

Number of Fisher Scoring iterations: 4

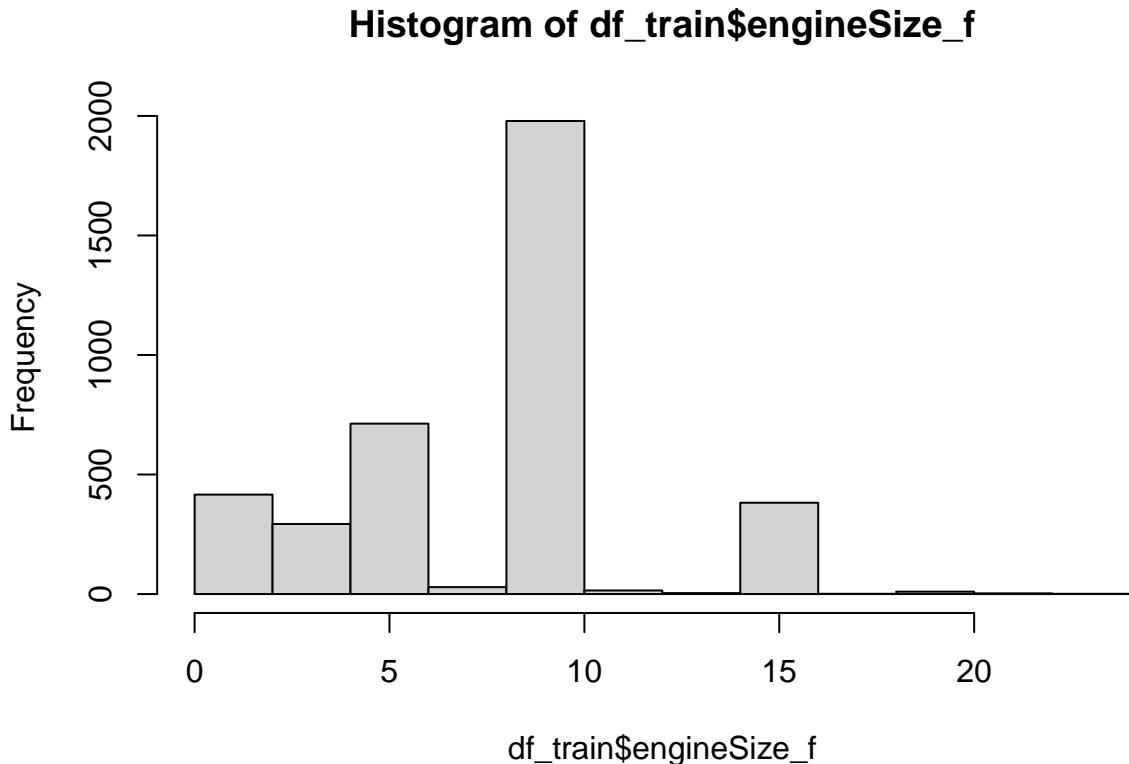
Podemos ver como el AIC de este modelo es 3807.8.

15.2 Factores

Vamos a proceder a añadir los factores que resultaron más significativos en el análisis MCA de la segunda entrega.

Como el factor engineSize tiene muchos niveles, vamos a proceder a transformala para que adquiera solo 3 niveles y poder simplificar el modelo.

```
df_train$engineSize_f <- as.integer(df_train$engineSize)
par(mfrow=c(1,1))
hist(df_train$engineSize_f)
```



```
quantile(df_train$engineSize_f, c(0.3333333, 0.6666666, 1))
```

33.33333%	66.66666%	100%
6	9	23

```
df_train$engineSize_f <- factor(cut(df_train$engineSize_f, breaks = c(0,8,9,20)))
df_test$engineSize_f <- as.integer(df_test$engineSize)
df_test$engineSize_f <- factor(cut(df_test$engineSize_f, breaks = c(0,8,9,20)))
table(df_train$engineSize_f)
```

(0,8]	(8,9]	(9,20]
1451	1657	734

Añadimos los factores fuelType y transmission al modelo. Podemos ver como añadiendo estos dos factores hemos conseguido una reducción del AIC. Además, con la función vif podemos ver que no hay aparente co-linealidad entre las variables explicativas del modelo.

```
m24 <- update(m21, ~.+fuelType+transmission)
summary(m24)
```

Call:
`glm(formula = Audi ~ mpg + age + fuelType + transmission, family = "binomial", data = df_train)`

Deviance Residuals:

Min	1Q	Median	3Q	Max
-1.3733	-0.7328	-0.6125	-0.4412	2.3595

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	1.142132	0.297375	3.841	0.000123 ***
mpg	-0.047459	0.004926	-9.634	< 2e-16 ***
age	0.122257	0.022727	5.379	7.47e-08 ***

```

fuelTypef.Fuel-Petrol      -0.114589  0.094457  -1.213 0.225082
fuelTypef.Fuel-Hybrid      -0.931659  0.603804  -1.543 0.122835
transmissionf.Trans-SemiAuto -0.598076  0.101964  -5.866 4.48e-09 ***
transmissionf.Trans-Automatic -0.599835  0.115009  -5.216 1.83e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

(Dispersion parameter for binomial family taken to be 1)

```

Null deviance: 3972.7  on 3845  degrees of freedom
Residual deviance: 3835.1  on 3839  degrees of freedom
AIC: 3849.1

```

Number of Fisher Scoring iterations: 4

```
vif(m24)
```

	GVIF	Df	GVIF^(1/(2*Df))
mpg	1.662987	1	1.289569
age	1.221182	1	1.105071
fuelType	1.376507	2	1.083165
transmission	1.335989	2	1.075105

Si analizamos la varianza del modelo, podemos ver como todas las variables de nuestro modelo son significativas.

```
Anova(m24, test="LR")
```

Analysis of Deviance Table (Type II tests)

Response: Audi

	LR	Chisq	Df	Pr(>Chisq)
mpg	95.876	1	< 2.2e-16	***
age	28.567	1	9.050e-08	***
fuelType	4.322	2	0.1152	
transmission	40.936	2	1.291e-09	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Con el test anova podemos ver como los modelos no son equivalentes, de modo que nos quedaremos con el que incluye los factores, ya que presenta un AIC inferior.

```
anova(m21,m24, test="LR")
```

Analysis of Deviance Table

Model 1: Audi ~ mpg + age					
Model 2: Audi ~ mpg + age + fuelType + transmission					
Resid.	Df	Resid.	Dev Df	Deviance	Pr(>Chi)
1	3843	3882.2			
2	3839	3835.1	4	47.098	1.455e-09 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Ahora añadiremos el factor derivado de engineSize que hemos generado antes. Podemos ver que este nuevo modelo vuelve a presentar un AIC inferior al del anterior. Además, si usamos la función vif podemos ver como no existe co-linealidad entre las variables.

```
m25 <- update(m24, ~.+engineSize_f)
summary(m25)
```

```

Call:
glm(formula = Audi ~ mpg + age + fuelType + transmission + engineSize_f,
     family = "binomial", data = df_train)

Deviance Residuals:
    Min      1Q  Median      3Q      Max 
-1.5906 -0.7315 -0.5831 -0.3365  2.5436 

Coefficients:
                                         Estimate Std. Error z value Pr(>|z|)    
(Intercept)                         2.564723  0.364598  7.034 2.00e-12 ***
mpg                                -0.070662  0.005855 -12.068 < 2e-16 ***
age                                 0.192331  0.024037  8.002 1.23e-15 ***
fuelTypeef.Fuel-Petrol            -0.592994  0.119091 -4.979 6.38e-07 ***
fuelTypeef.Fuel-Hybrid           -1.407572  0.610139 -2.307 0.021057 *  
transmissionf.Trans-SemiAuto   -0.355643  0.106519 -3.339 0.000841 *** 
transmissionf.Trans-Automatic  -0.321235  0.119360 -2.691 0.007117 **  
engineSize_f(8,9]                 -0.354378  0.115704 -3.063 0.002193 ** 
engineSize_f(9,20]                -1.636599  0.180301 -9.077 < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

(Dispersion parameter for binomial family taken to be 1)

```

Null deviance: 3968.2 on 3841 degrees of freedom
Residual deviance: 3721.7 on 3833 degrees of freedom
(4 observations deleted due to missingness)
AIC: 3739.7

```

Number of Fisher Scoring iterations: 5

```
vif(m25)
```

	GVIF	Df	GVIF^(1/(2*Df))
mpg	2.163591	1	1.470915
age	1.371424	1	1.171078
fuelType	2.130429	2	1.208139
transmission	1.431407	2	1.093807
engineSize_f	2.217359	2	1.220279

Si observamos el análisis de la varianza, vemos que todos los componentes de nuestro modelo son representativos. Por otro lado, no podemos usar las funciones anova ni AIC para comparar ambos modelos, ya que al incluir el factor EngineSize, se han generado algunos missings en los datos que imposibilitan la comparación.

```
Anova(m25, test="LR")
```

Analysis of Deviance Table (Type II tests)

```

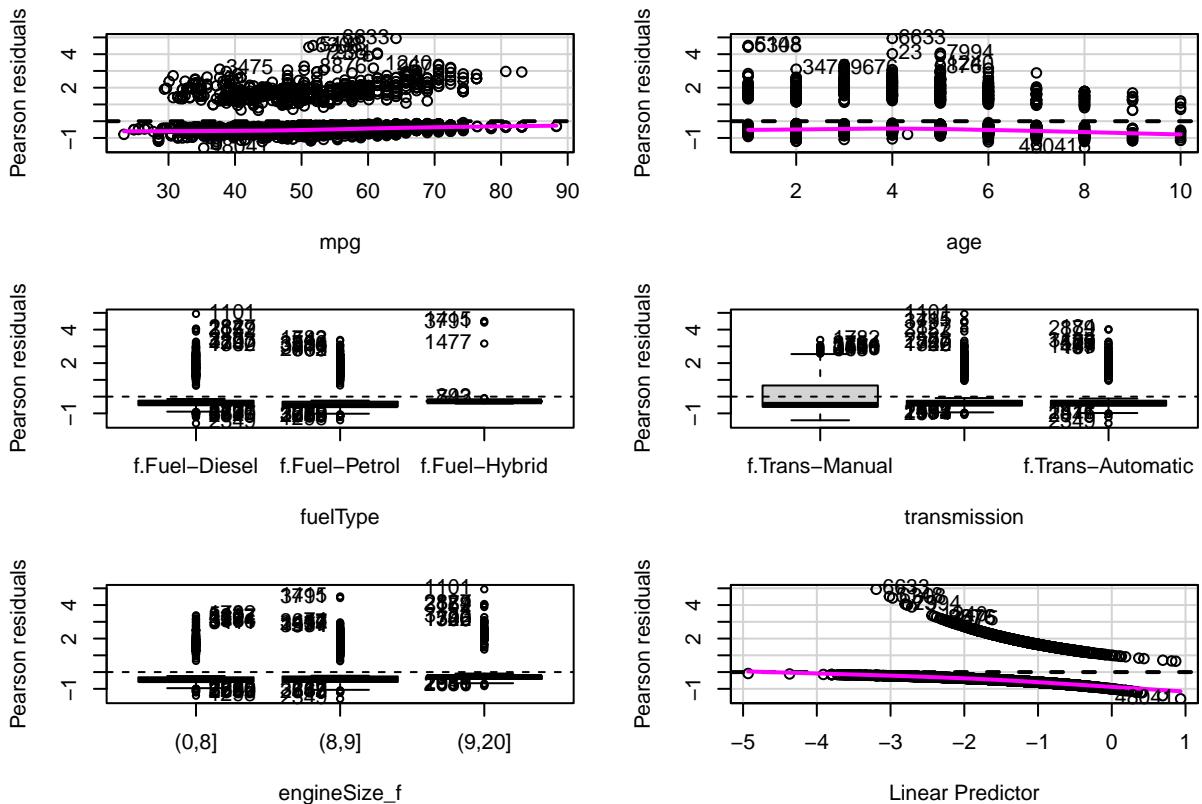
Response: Audi
          LR Chisq Df Pr(>Chisq)
mpg       155.347  1  < 2.2e-16 ***
age        63.596  1  1.528e-15 ***
fuelType   29.608  2  3.722e-07 ***
transmission 12.311  2   0.002122 **
engineSize_f 108.229  2  < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```
#anova(m24,m25, test="LR")      #No se pueden hacer porque han aparecido missings en el dataset cuando heredémoslos
#AIC(m24,m25)
```

Sin embargo, seguiremos avanzando con el nuevo modelo que hemos generado ya que tiene un AIC inferior.

```
residualPlots(m25, id=list(method=cooks.distance(m25), n=10))
```

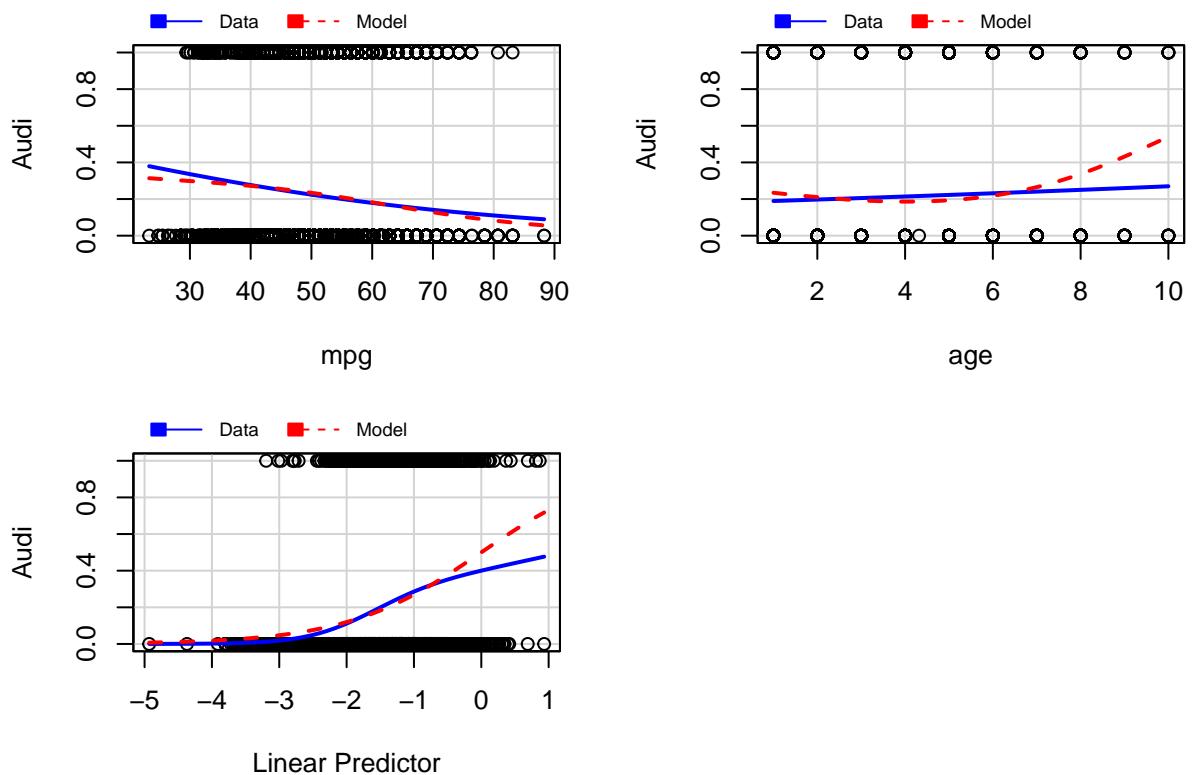


```
Test stat Pr(>|Test stat|)  
mpg 1.9032 0.1677  
age 19.2408 1.152e-05 ***  
fuelType  
transmission  
engineSize_f  
---  
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
marginalModelPlots(m25)
```

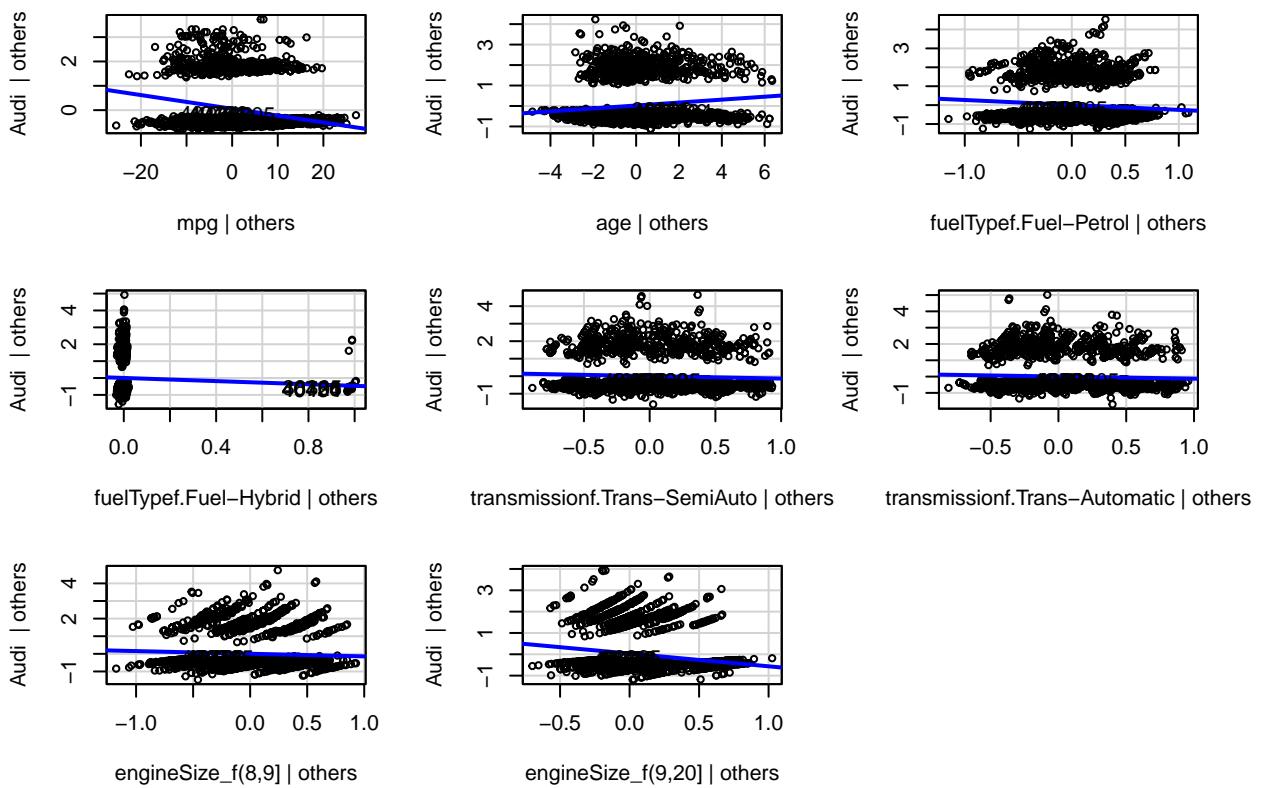
Warning in mmpp(...): Interactions and/or factors skipped

Marginal Model Plots

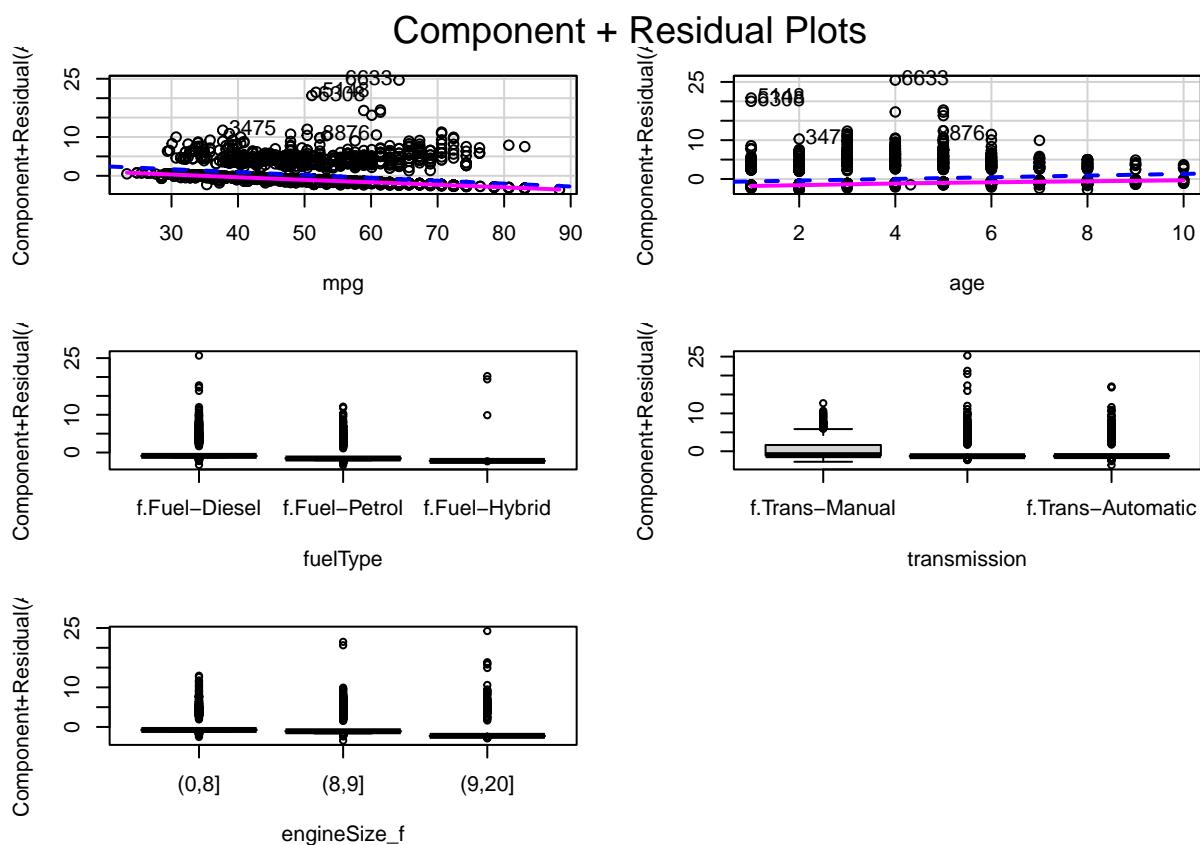


```
avPlots(m25, id=list(method=hatvalues(m25), n=5))
```

Added-Variable Plots

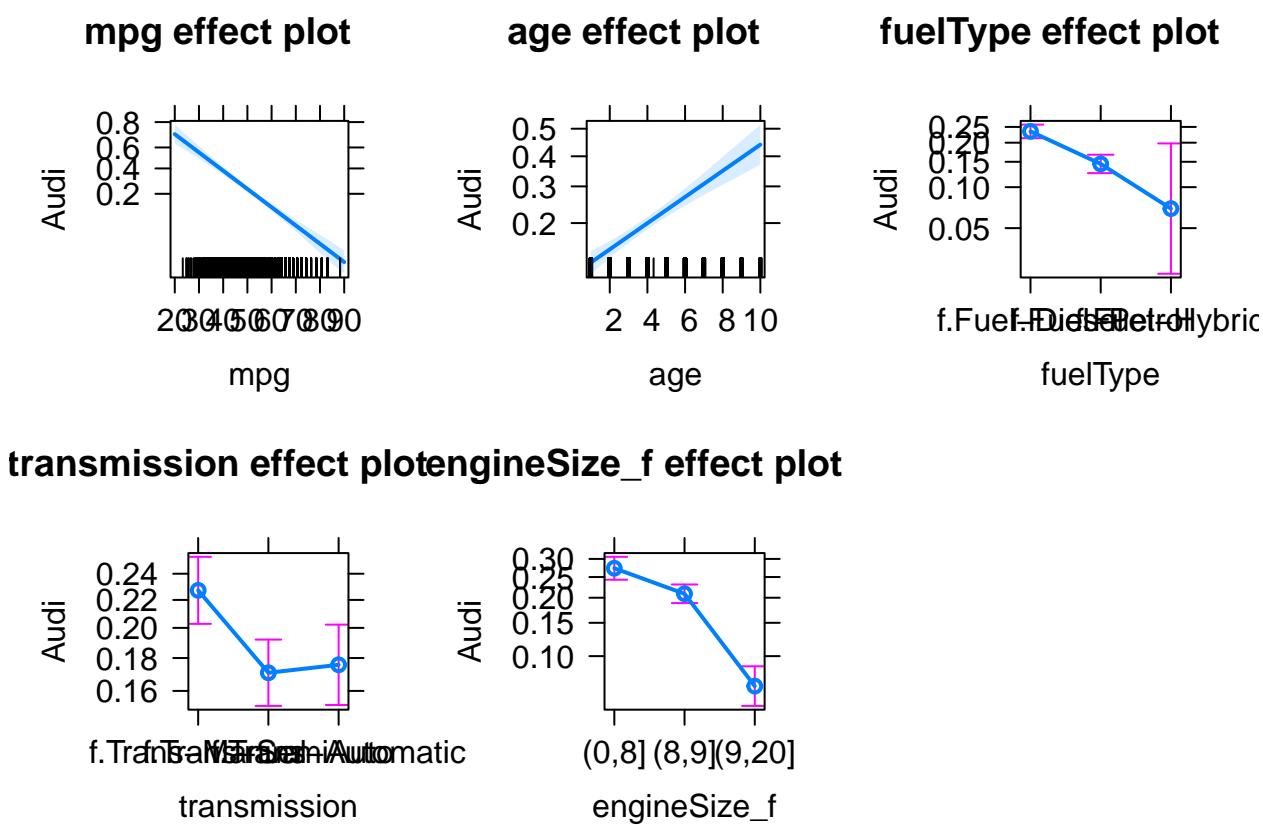


```
crPlots(m25,id=list(method=cooks.distance(m25),n=5))
```



En los siguientes plots podemos ver como varía la probabilidad de que el vehículo sea Audi dependiendo de las distintas variables explicativas que aparecen en el modelo.

```
# library(effects)
plot(allEffects(m25))+theme(axis.text.x = element_text(angle=90))
```



NULL

Para el caso de mpg, parece que hay una relación de proporcionalidad inversa, de manera que cuanto más consumo tenga el coche, más probable será que este sea Audi.

Para la variable age podemos ver que existe una relación de proporcionalidad directa, de modo que cuanto más viejo sea el coche, más probabilidades tendrá de ser Audi.

En lo que se refiere al factor fuelType, podemos ver como el hecho de que el coche sea diesel o gasolina aumenta las probabilidades de que sea Audi.

Por último, la transmisión manual también aumentaría esta probabilidad, así como los engineSize entre 0 y 9, reduciéndose drásticamente la probabilidad de que el coche sea Audi para los coches con engineSize entre 9 y 20.

15.3 Interacciones

Procederemos a incorporar las interacciones entre los factores fuelType y transmission. Podemos ver como el AIC del modelo se reduce inmediatamente. Sin embargo, cuando intentamos usar la función vif para estudiar la co-linealidad de las variables, nos genera un error ya que parece que sí que existe co-linealidad.

```
m26 <- update(m25, ~*(fuelType+transmission)^2, data=df_train)
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
#summary(m26) Omitimos debido a la longitud de la salida  
#vif(m26)
```

Si realizamos un análisis de la varianza del modelo, vemos como hay algunas interacciones que no son significativas, como pueden ser mpg:fuelType:transmission, age:fuelType:transmission o fuelType:transmission.

```
Anova(m26, test="LR")
```

```
Warning: glm.fit: algorithm did not converge
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

```

Analysis of Deviance Table (Type II tests)

Response: Audi

	LR	Chisq	Df	Pr(>Chisq)
mpg		0		
age	54.782	1	1.347e-13	***
fuelType	26.837	2	1.487e-06	***
transmission	10.045	2	0.006587	**
engineSize_f	97.233	2	< 2.2e-16	***
fuelType:transmission	0.899	3	0.825614	
mpg:fuelType	7.416	2	0.024529	*
mpg:transmission	6.208	2	0.044860	*
age:fuelType	8.330	2	0.015529	*
age:transmission	9.840	2	0.007297	**
fuelType:engineSize_f	3.730	4	0.443801	
transmission:engineSize_f	11.617	4	0.020441	*
mpg:fuelType:transmission	1.400	3	0.705568	
age:fuelType:transmission	0.143	3	0.986205	
fuelType:transmission:engineSize_f	3.122	6	0.793342	

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Usaremos la función step para plantear el mejor modelo posible a partir de una simplificación del que ya tenemos. Si intentamos volver a ejecutar la función vif, vuelve a generar un error que indica que sigue existiendo colinealidad entre las variables de nuestro modelo, pero en el summary podemos ver como el AIC de este modelo vuelve a ser inferior al del anterior.

m27 <- step(m26)

```

Start:  AIC=3729.22
Audi ~ mpg + age + fuelType + transmission + engineSize_f + fuelType:transmission +
      mpg:fuelType + mpg:transmission + age:fuelType + age:transmission +
      fuelType:engineSize_f + transmission:engineSize_f + mpg:fuelType:transmission +
      age:fuelType:transmission + fuelType:transmission:engineSize_f

```

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

	Df	Deviance	AIC
- fuelType:transmission:engineSize_f	6	3652.3	3720.3
- age:fuelType:transmission	3	3649.4	3723.4
- mpg:fuelType:transmission	3	3650.6	3724.6
<none>		3649.2	3729.2

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

```

Step: AIC=3720.34
Audi ~ mpg + age + fuelType + transmission + engineSize_f + fuelType:transmission +
      mpg:fuelType + mpg:transmission + age:fuelType + age:transmission +
      fuelType:engineSize_f + transmission:engineSize_f + mpg:fuelType:transmission +
      age:fuelType:transmission

```

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

	Df	Deviance	AIC
- age:fuelType:transmission	3	3652.4	3714.4
- mpg:fuelType:transmission	3	3653.5	3715.5
- fuelType:engineSize_f	4	3656.1	3716.1
<none>		3652.3	3720.3
- transmission:engineSize_f	4	3664.0	3724.0

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

Step: AIC=3714.41

```

Audi ~ mpg + age + fuelType + transmission + engineSize_f + fuelType:transmission +
      mpg:fuelType + mpg:transmission + age:fuelType + age:transmission +
      fuelType:engineSize_f + transmission:engineSize_f + mpg:fuelType:transmission

```

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

	Df	Deviance	AIC
<none>		3652.4	3714.4
- transmission:engineSize_f	4	3664.0	3718.0
- age:fuelType	2	3660.7	3718.7
- age:transmission	2	3662.3	3720.3
- fuelType:engineSize_f	4	3670.9	3724.9
- mpg:fuelType:transmission	3	3673.2	3729.2

#vif(m27)

summary(m27)

Call:

```

glm(formula = Audi ~ mpg + age + fuelType + transmission + engineSize_f +
    fuelType:transmission + mpg:fuelType + mpg:transmission +
    age:fuelType + age:transmission + fuelType:engineSize_f +
    transmission:engineSize_f + mpg:fuelType:transmission, family = "binomial",
    data = df_train)

```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-1.4977	-0.7480	-0.5915	-0.2328	2.7323

Coefficients: (2 not defined because of singularities)

	Estimate	Std. Error
(Intercept)	1.599e+00	7.410e-01
mpg	-5.431e-02	1.222e-02
age	1.691e-01	4.508e-02

fuelTypef.Fuel-Petrol	-9.308e-01	9.579e-01
fuelTypef.Fuel-Hybrid	-4.740e+03	1.046e+05
transmissionf.Trans-SemiAuto	1.622e+00	1.003e+00
transmissionf.Trans-Automatic	1.457e+00	1.002e+00
engineSize_f(8,9]	-6.988e-03	2.023e-01
engineSize_f(9,20]	-1.988e+01	2.587e+03
fuelTypef.Fuel-Petrol:transmissionf.Trans-SemiAuto	-1.012e+00	1.273e+00
fuelTypef.Fuel-Hybrid:transmissionf.Trans-SemiAuto	3.925e+03	9.034e+04
fuelTypef.Fuel-Petrol:transmissionf.Trans-Automatic	-1.330e+00	1.411e+00
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic	NA	NA
mpg:fuelTypef.Fuel-Petrol	1.008e-02	1.711e-02
mpg:fuelTypef.Fuel-Hybrid	8.690e+01	1.902e+03
mpg:transmissionf.Trans-SemiAuto	-4.439e-02	1.702e-02
mpg:transmissionf.Trans-Automatic	-3.071e-02	1.700e-02
age:fuelTypef.Fuel-Petrol	-1.740e-02	5.083e-02
age:fuelTypef.Fuel-Hybrid	-8.772e+01	1.975e+03
age:transmissionf.Trans-SemiAuto	1.268e-01	5.824e-02
age:transmissionf.Trans-Automatic	-7.551e-02	6.244e-02
fuelTypef.Fuel-Petrol:engineSize_f(8,9]	-3.080e-01	2.685e-01
fuelTypef.Fuel-Hybrid:engineSize_f(8,9]	6.183e+00	1.414e+04
fuelTypef.Fuel-Petrol:engineSize_f(9,20]	2.696e-01	4.318e-01
fuelTypef.Fuel-Hybrid:engineSize_f(9,20]	-2.617e+02	1.118e+04
transmissionf.Trans-SemiAuto:engineSize_f(8,9]	-2.988e-01	3.018e-01
transmissionf.Trans-Automatic:engineSize_f(8,9]	-6.937e-02	3.424e-01
transmissionf.Trans-SemiAuto:engineSize_f(9,20]	1.810e+01	2.587e+03
transmissionf.Trans-Automatic:engineSize_f(9,20]	1.859e+01	2.587e+03
mpg:fuelTypef.Fuel-Petrol:transmissionf.Trans-SemiAuto	2.432e-02	2.452e-02
mpg:fuelTypef.Fuel-Hybrid:transmissionf.Trans-SemiAuto	-6.908e+01	1.570e+03
mpg:fuelTypef.Fuel-Petrol:transmissionf.Trans-Automatic	3.279e-02	2.880e-02
mpg:fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic	NA	NA
z	value	Pr(> z)
(Intercept)	2.158	0.030895 *
mpg	-4.445	8.8e-06 ***
age	3.751	0.000176 ***
fuelTypef.Fuel-Petrol	-0.972	0.331193
fuelTypef.Fuel-Hybrid	-0.045	0.963858
transmissionf.Trans-SemiAuto	1.617	0.105935
transmissionf.Trans-Automatic	1.454	0.145863
engineSize_f(8,9]	-0.035	0.972441
engineSize_f(9,20]	-0.008	0.993867
fuelTypef.Fuel-Petrol:transmissionf.Trans-SemiAuto	-0.795	0.426695
fuelTypef.Fuel-Hybrid:transmissionf.Trans-SemiAuto	0.043	0.965340
fuelTypef.Fuel-Petrol:transmissionf.Trans-Automatic	-0.943	0.345855
fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic	NA	NA
mpg:fuelTypef.Fuel-Petrol	0.589	0.555785
mpg:fuelTypef.Fuel-Hybrid	0.046	0.963565
mpg:transmissionf.Trans-SemiAuto	-2.609	0.009091 **
mpg:transmissionf.Trans-Automatic	-1.807	0.070818 .
age:fuelTypef.Fuel-Petrol	-0.342	0.732079
age:fuelTypef.Fuel-Hybrid	-0.044	0.964569
age:transmissionf.Trans-SemiAuto	2.178	0.029415 *
age:transmissionf.Trans-Automatic	-1.209	0.226550
fuelTypef.Fuel-Petrol:engineSize_f(8,9]	-1.147	0.251267
fuelTypef.Fuel-Hybrid:engineSize_f(8,9]	0.000	0.999651
fuelTypef.Fuel-Petrol:engineSize_f(9,20]	0.624	0.532383
fuelTypef.Fuel-Hybrid:engineSize_f(9,20]	-0.023	0.981325
transmissionf.Trans-SemiAuto:engineSize_f(8,9]	-0.990	0.322265
transmissionf.Trans-Automatic:engineSize_f(8,9]	-0.203	0.839456
transmissionf.Trans-SemiAuto:engineSize_f(9,20]	0.007	0.994415
transmissionf.Trans-Automatic:engineSize_f(9,20]	0.007	0.994265
mpg:fuelTypef.Fuel-Petrol:transmissionf.Trans-SemiAuto	0.992	0.321091
mpg:fuelTypef.Fuel-Hybrid:transmissionf.Trans-SemiAuto	-0.044	0.964895
mpg:fuelTypef.Fuel-Petrol:transmissionf.Trans-Automatic	1.139	0.254838
mpg:fuelTypef.Fuel-Hybrid:transmissionf.Trans-Automatic	NA	NA

```

---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 3968.2 on 3841 degrees of freedom
Residual deviance: 3652.4 on 3811 degrees of freedom
(4 observations deleted due to missingness)
AIC: 3714.4
```

Number of Fisher Scoring iterations: 19

Si realizamos un análisis de la varianza, podemos ver como algunas interacciones parecen no ser significativas.

```
Anova(m27, test="LR")
```

```

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

Analysis of Deviance Table (Type II tests)

Response: Audi

	LR	Chisq	Df	Pr(>Chisq)
mpg	147.679	1	< 2.2e-16	***
age	54.063	1	1.942e-13	***
fuelType	26.837	2	1.487e-06	***
transmission	10.045	2	0.0065873	**
engineSize_f	99.617	2	< 2.2e-16	***
fuelType:transmission	0.899	3	0.8256144	
mpg:fuelType	6.781	2	0.0336937	*
mpg:transmission	5.927	2	0.0516334	.
age:fuelType	8.322	2	0.0155898	*
age:transmission	9.918	2	0.0070197	**
fuelType:engineSize_f	18.513	4	0.0009794	***
transmission:engineSize_f	11.631	4	0.0203169	*
mpg:fuelType:transmission	20.835	3	0.0001139	***

```

---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Vamos a proceder a generar un nuevo modelo que excluya las interacciones no significativas del modelo anterior. Nos encontramos que cuando analizamos este modelo, vuelve a aparecer co-linealidad en las variables, de modo que realizaremos una step regression.

```
m28 <- update(m27, ~.-fuelType:transmission - age:fuelType)
#vif(m28) -> Salta error
```

En este caso, si que podemos ver que ya ha desaparecido la co-linealidad en las variables, y si realizamos el test de anova para los dos modelos, podemos ver que son equivalentes.

```
m29 <- step(m28)
```

```

Start: AIC=3728.84
Audi ~ mpg + age + fuelType + transmission + engineSize_f + mpg:fuelType +
     mpg:transmission + age:transmission + fuelType:engineSize_f +
```

```
transmission:engineSize_f + mpg:fuelType:transmission
```

	Df	Deviance	AIC
- mpg:fuelType:transmission	3	3678.5	3724.5
- fuelType:engineSize_f	4	3680.7	3724.7
<none>		3676.8	3728.8
- transmission:engineSize_f	4	3687.9	3731.9
- age:transmission	2	3685.3	3733.3

Step: AIC=3724.49

```
Audi ~ mpg + age + fuelType + transmission + engineSize_f + mpg:fuelType +
    mpg:transmission + age:transmission + fuelType:engineSize_f +
    transmission:engineSize_f
```

	Df	Deviance	AIC
- fuelType:engineSize_f	4	3681.6	3719.6
- mpg:fuelType	2	3681.7	3723.7
<none>		3678.5	3724.5
- age:transmission	2	3687.7	3729.7
- transmission:engineSize_f	4	3692.9	3730.9
- mpg:transmission	2	3691.0	3733.0

Step: AIC=3719.64

```
Audi ~ mpg + age + fuelType + transmission + engineSize_f + mpg:fuelType +
    mpg:transmission + age:transmission + transmission:engineSize_f
```

	Df	Deviance	AIC
<none>		3681.6	3719.6
- mpg:fuelType	2	3687.2	3721.2
- age:transmission	2	3691.2	3725.2
- transmission:engineSize_f	4	3697.0	3727.0
- mpg:transmission	2	3694.5	3728.5

```
vif(m29)
```

	GVIF	Df	GVIF^(1/(2*Df))
mpg	4.746399e+00	1	2.178623
age	3.123946e+00	1	1.767469
fuelType	3.185770e+03	2	7.512831
transmission	1.262822e+03	2	5.961226
engineSize_f	9.730630e+06	2	55.851551
mpg:fuelType	3.047992e+03	2	7.430250
mpg:transmission	1.211622e+03	2	5.899861
age:transmission	3.718803e+01	2	2.469453
transmission:engineSize_f	6.427212e+07	4	9.462433

```
anova(m29,m28, test="LR")
```

Analysis of Deviance Table

Model 1: Audi ~ mpg + age + fuelType + transmission + engineSize_f + mpg:fuelType +
 mpg:transmission + age:transmission + transmission:engineSize_f

Model 2: Audi ~ mpg + age + fuelType + transmission + engineSize_f + mpg:fuelType +
 mpg:transmission + age:transmission + fuelType:engineSize_f +
 transmission:engineSize_f + mpg:fuelType:transmission

Resid. Df Resid. Dev Df Deviance Pr(>Chi)

1	3823	3681.6		
2	3816	3676.8	7	4.8001 0.6843

15.4 Diagnóstico

En primer lugar, vamos a ver el summary de nuestro modelo.

```
summary(m29)
```

Call:

```
glm(formula = Audi ~ mpg + age + fuelType + transmission + engineSize_f +
    mpg:fuelType + mpg:transmission + age:transmission + transmission:engineSize_f,
    family = "binomial", data = df_train)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-1.4839	-0.7390	-0.5901	-0.2725	2.6791

Coefficients:

	Estimate	Std. Error	z value
(Intercept)	1.739720	0.485501	3.583
mpg	-0.055637	0.008661	-6.424
age	0.156078	0.036655	4.258
fuelTypef.Fuel-Petrol	-1.565899	0.466743	-3.355
fuelTypef.Fuel-Hybrid	-6.420486	6.020889	-1.066
transmissionf.Trans-SemiAuto	1.550525	0.576323	2.690
transmissionf.Trans-Automatic	1.474629	0.626203	2.355
engineSize_f(8,9]	-0.037455	0.163308	-0.229
engineSize_f(9,20]	-14.764485	213.637592	-0.069
mpg:fuelTypef.Fuel-Petrol	0.020356	0.009243	2.202
mpg:fuelTypef.Fuel-Hybrid	0.094942	0.109945	0.864
mpg:transmissionf.Trans-SemiAuto	-0.039864	0.011492	-3.469
mpg:transmissionf.Trans-Automatic	-0.027505	0.012284	-2.239
age:transmissionf.Trans-SemiAuto	0.129696	0.056742	2.286
age:transmissionf.Trans-Automatic	-0.061159	0.059916	-1.021
transmissionf.Trans-SemiAuto:engineSize_f(8,9]	-0.508671	0.215095	-2.365
transmissionf.Trans-Automatic:engineSize_f(8,9]	-0.343384	0.265313	-1.294
transmissionf.Trans-SemiAuto:engineSize_f(9,20]	12.901525	213.637697	0.060
transmissionf.Trans-Automatic:engineSize_f(9,20]	13.278980	213.637754	0.062
	Pr(> z)		
(Intercept)	0.000339 ***		
mpg	1.33e-10 ***		
age	2.06e-05 ***		
fuelTypef.Fuel-Petrol	0.000794 ***		
fuelTypef.Fuel-Hybrid	0.286257		
transmissionf.Trans-SemiAuto	0.007137 **		
transmissionf.Trans-Automatic	0.018529 *		
engineSize_f(8,9]	0.818596		
engineSize_f(9,20]	0.944902		
mpg:fuelTypef.Fuel-Petrol	0.027638 *		
mpg:fuelTypef.Fuel-Hybrid	0.387843		
mpg:transmissionf.Trans-SemiAuto	0.000522 ***		
mpg:transmissionf.Trans-Automatic	0.025152 *		
age:transmissionf.Trans-SemiAuto	0.022270 *		
age:transmissionf.Trans-Automatic	0.307372		
transmissionf.Trans-SemiAuto:engineSize_f(8,9]	0.018037 *		
transmissionf.Trans-Automatic:engineSize_f(8,9]	0.195576		
transmissionf.Trans-SemiAuto:engineSize_f(9,20]	0.951845		
transmissionf.Trans-Automatic:engineSize_f(9,20]	0.950438		

Signif. codes:	0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1		

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 3968.2 on 3841 degrees of freedom

Residual deviance: 3681.6 on 3823 degrees of freedom

(4 observations deleted due to missingness)

AIC: 3719.6

Number of Fisher Scoring iterations: 14

Como podemos ver, su valor de AIC asociado es de 3639.7. Aunque no es el mejor resultado que hemos conseguido, ya que el m28 tenia 3646.8, este no presenta co-linealidad.

Como hemos comentado anteriormente, con la función vif podemos ver como no aparece co-linealidad entre las variables del modelo. Si analizamos la varianza del modelo, veremos como todas las variables que aparecen son significativas.

```
vif(m29)
```

	GVIF	Df	GVIF^(1/(2*Df))
mpg	4.746399e+00	1	2.178623
age	3.123946e+00	1	1.767469
fuelType	3.185770e+03	2	7.512831
transmission	1.262822e+03	2	5.961226
engineSize_f	9.730630e+06	2	55.851551
mpg:fuelType	3.047992e+03	2	7.430250
mpg:transmission	1.211622e+03	2	5.899861
age:transmission	3.718803e+01	2	2.469453
transmission:engineSize_f	6.427212e+07	4	9.462433

```
Anova(m29)
```

Analysis of Deviance Table (Type II tests)

Response: Audi

	LR	Chisq	Df	Pr(>Chisq)
mpg		148.977	1	< 2.2e-16 ***
age		54.012	1	1.993e-13 ***
fuelType		26.837	2	1.487e-06 ***
transmission		11.322	2	0.003480 **
engineSize_f		111.158	2	< 2.2e-16 ***
mpg:fuelType		5.539	2	0.062683 .
mpg:transmission		12.843	2	0.001627 **
age:transmission		9.594	2	0.008256 **
transmission:engineSize_f		15.318	4	0.004085 **

Signif. codes:	0	'***'	0.001	'**'
	0.01	'*'	0.05	'. '
	0.1	'	1	'

Vamos a proceder a analizar algunos gráficos del modelo que hemos generado.

Si analizamos los boxplots de los los valores de Hat y las distancias de Cook, podemos ver como hay una serie de elementos que aparecen muy separados, algunos coinciden en los tres gráficos.

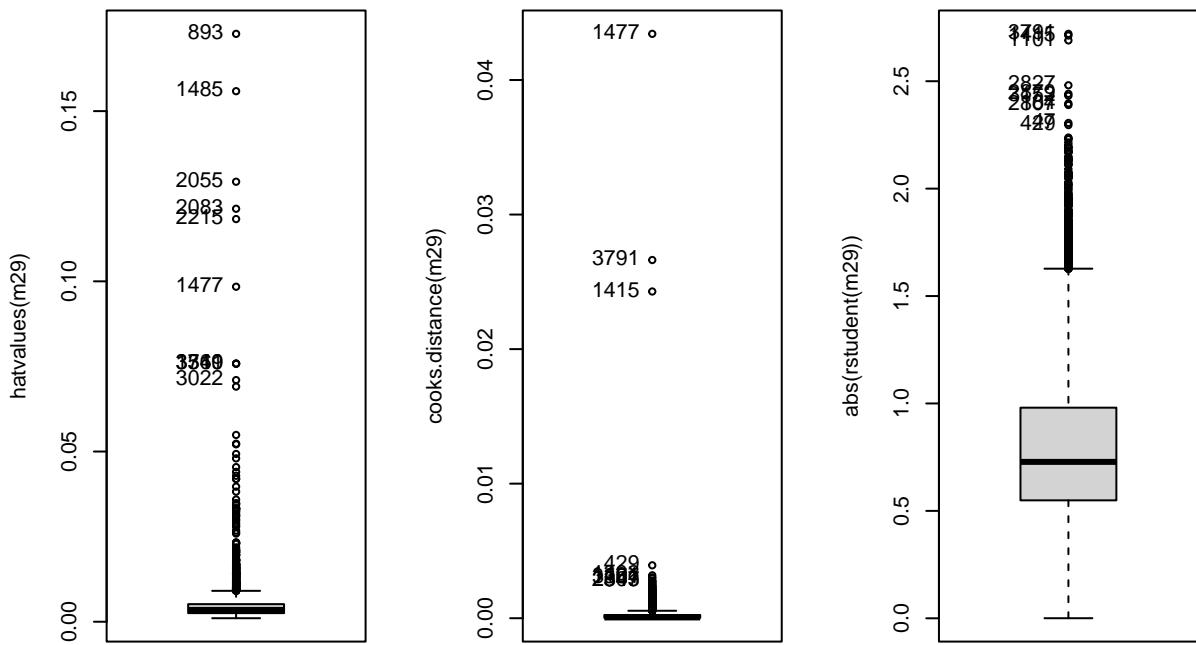
```
par(mfrow=c(1,3))
Boxplot(hatvalues(m29), id=c(labels=row.names(df_train)))
```

```
[1] 893 1485 2055 2083 2215 1477 1361 3540 3719 3022
```

```
Boxplot(cooks.distance(m29), id=c(labels=row.names(df_train)))
```

```
[1] 1477 3791 1415 429 1302 1101 3381 2807 2349 543
```

```
Boxplot(abs(rstudent(m29)), id=c(labels=row.names(df_train)))
```



```
[1] 3791 1415 1101 2827 2879 3152 184 2807 47 429
```

En primer lugar, vamos a observar los valores de los residuos de student. Al haber generado el gráfico con el valor absoluto, solo tendremos que filtrar la parte superior del boxplot. Podemos ver como la cadena de puntos se rompe aproximadamente cerca del 2.3, de manera que vamos a considerar los individuos con residuo de student fuera del intervalo [-2.3, 2.3] como outliers. Haremos lo mismo para las observaciones con distancias de Cook superiores a 0.0035 y Hat values superiores a 0.07.

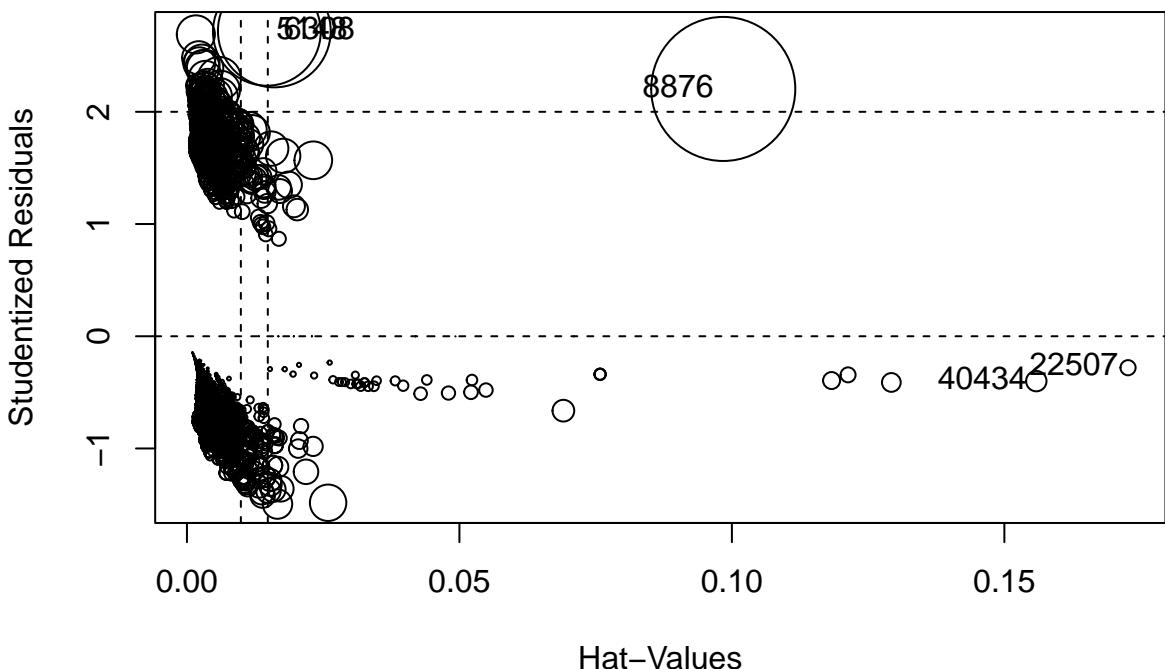
```
stu_out <- which(abs(rstudent(m29))>2.3);
cook_out <- which(abs(cooks.distance(m29))>0.0035);
hat_out <- which(abs(hatvalues(m29))>0.07);

outs<-unique(c(stu_out,cook_out,hat_out));outs
```

```
[1] 47 184 1101 1415 2807 2827 2879 3152 3791 429 1477 893 1361 1485 2055
[16] 2083 2215 3022 3540 3719
```

Si echamos un vistazo al plot que nos marca la influencia que tienen las distintas observaciones hacia el modelo, podemos ver como aparecen algunos puntos bastante alejados de las nubes que se crean, algunos con bastante influencia.

```
par(mfrow=c(1,1));
outs2 <- influencePlot(m29, id=c(labels=row.names(df_train)));
```



```
outs2 <- labels(outs2)[[1]];
outs2 <- as.numeric(outs2);
outs3 <- unique(c(outs,outs2));outs3
```

```
[1]     47    184   1101   1415   2807   2827   2879   3152   3791   429   1477   893
[13]   1361   1485   2055   2083   2215   3022   3540   3719  22507   5148   8876  40434
[25]   6308
```

Vamos a proceder a crear un nuevo modelo que excluya las observaciones que hemos considerado como outliers.

```
m30 <- update(m29,data=df_train[-outs3,])
summary(m30)
```

Call:
`glm(formula = Audi ~ mpg + age + fuelType + transmission + engineSize_f +
 mpg:fuelType + mpg:transmission + age:transmission + transmission:engineSize_f,
 family = "binomial", data = df_train[-outs3,])`

Deviance Residuals:

Min	1Q	Median	3Q	Max
-1.5174	-0.7414	-0.5873	-0.2625	2.6290

Coefficients:

	Estimate	Std. Error	z value
(Intercept)	1.851761	0.488361	3.792
mpg	-0.057925	0.008747	-6.622
age	0.160835	0.036975	4.350
fuelTypef.Fuel-Petrol	-1.672794	0.469095	-3.566
fuelTypef.Fuel-Hybrid	-7.145769	6.725512	-1.062
transmissionf.Trans-SemiAuto	1.566644	0.580477	2.699
transmissionf.Trans-Automatic	1.600124	0.635129	2.519
engineSize_f(8,9]	-0.048438	0.164221	-0.295
engineSize_f(9,20]	-14.766050	213.251070	-0.069
mpg:fuelTypef.Fuel-Petrol	0.022324	0.009292	2.402

```

mpg:fuelTypef.Fuel-Hybrid          0.108826  0.123017  0.885
mpg:transmissionf.Trans-SemiAuto -0.039963  0.011608 -3.443
mpg:transmissionf.Trans-Automatic -0.030966  0.012538 -2.470
age:transmissionf.Trans-SemiAuto   0.125580  0.057064  2.201
age:transmissionf.Trans-Automatic -0.057610  0.060452 -0.953
transmissionf.Trans-SemiAuto:engineSize_f(8,9] -0.493400  0.215760 -2.287
transmissionf.Trans-Automatic:engineSize_f(8,9] -0.320307  0.267855 -1.196
transmissionf.Trans-SemiAuto:engineSize_f(9,20] 12.860481 213.251178  0.060
transmissionf.Trans-Automatic:engineSize_f(9,20] 13.239236 213.251238  0.062
Pr(>|z|)
(Intercept) 0.000150 ***
mpg 3.53e-11 ***
age 1.36e-05 ***
fuelTypef.Fuel-Petrol 0.000362 ***
fuelTypef.Fuel-Hybrid 0.288015
transmissionf.Trans-SemiAuto 0.006957 **
transmissionf.Trans-Automatic 0.011757 *
engineSize_f(8,9] 0.768028
engineSize_f(9,20] 0.944797
mpg:fuelTypef.Fuel-Petrol 0.016285 *
mpg:fuelTypef.Fuel-Hybrid 0.376350
mpg:transmissionf.Trans-SemiAuto 0.000576 ***
mpg:transmissionf.Trans-Automatic 0.013518 *
age:transmissionf.Trans-SemiAuto 0.027758 *
age:transmissionf.Trans-Automatic 0.340597
transmissionf.Trans-SemiAuto:engineSize_f(8,9] 0.022207 *
transmissionf.Trans-Automatic:engineSize_f(8,9] 0.231766
transmissionf.Trans-SemiAuto:engineSize_f(9,20] 0.951911
transmissionf.Trans-Automatic:engineSize_f(9,20] 0.950497
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

(Dispersion parameter for binomial family taken to be 1)

```

Null deviance: 3942.9  on 3821  degrees of freedom
Residual deviance: 3645.7  on 3803  degrees of freedom
(4 observations deleted due to missingness)
AIC: 3683.7

```

Number of Fisher Scoring iterations: 14

Podemos ver como el valor AIC del modelo ha disminuido.

Con la función vif vemos que no existen variables co-lineales en el modelo. Si analizamos la varianza, vemos que todas las variables son significativas.

```
vif(m30)
```

	GVIF	Df	GVIF^(1/(2*Df))
mpg	4.748679e+00	1	2.179146
age	3.143569e+00	1	1.773011
fuelType	3.963578e+03	2	7.934542
transmission	1.291403e+03	2	5.994672
engineSize_f	9.502591e+06	2	55.521414
mpg:fuelType	3.797001e+03	2	7.849829
mpg:transmission	1.242336e+03	2	5.936901
age:transmission	3.720192e+01	2	2.469684
transmission:engineSize_f	6.295735e+07	4	9.438018

```
Anova(m30, test="LR")
```

Analysis of Deviance Table (Type II tests)

```
Response: Audi
```

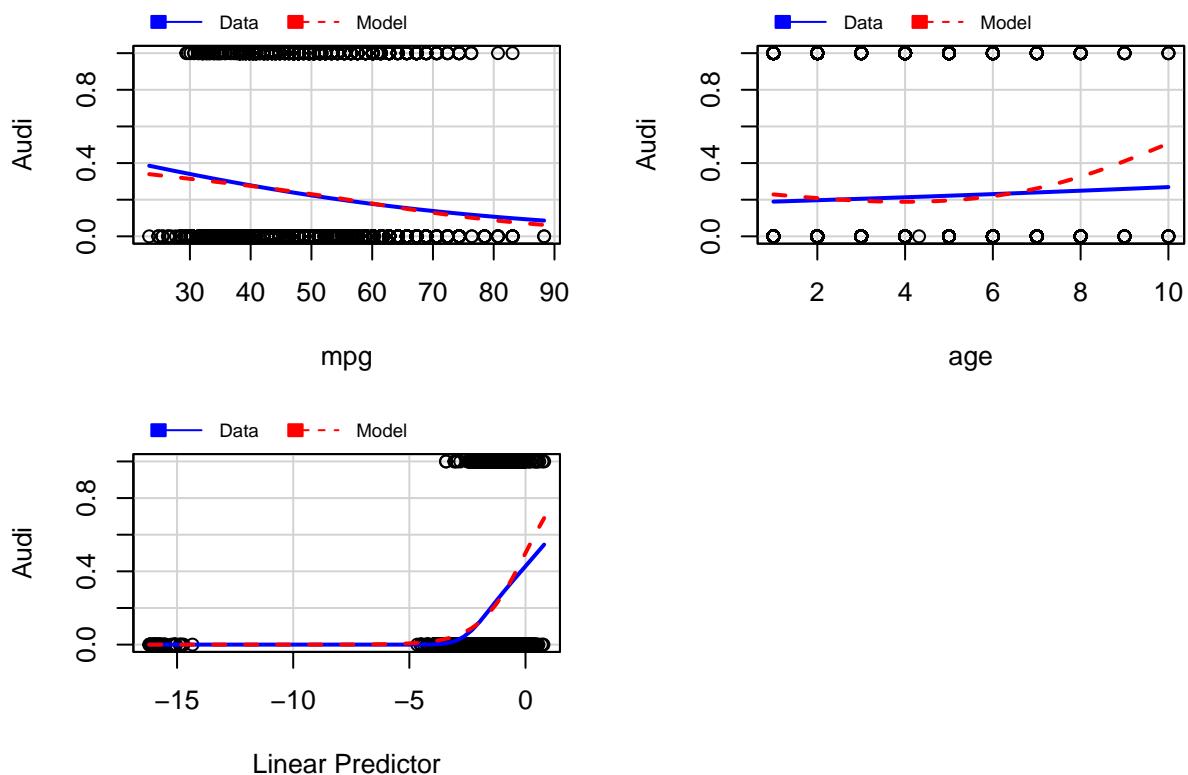
	LR	Chisq	Df	Pr(>Chisq)							
mpg		156.907	1	< 2.2e-16	***						
age		56.150	1	6.714e-14	***						
fuelType		27.433	2	1.104e-06	***						
transmission		11.687	2	0.002899	**						
engineSize_f		115.949	2	< 2.2e-16	***						
mpg:fuelType		6.579	2	0.037266	*						
mpg:transmission		13.177	2	0.001376	**						
age:transmission		8.778	2	0.012413	*						
transmission:engineSize_f		14.457	4	0.005971	**						

Signif. codes:	0	'***'	0.001	'**'	0.01	'*'	0.05	'.'	0.1	',	1

```
marginalModelPlots(m30)
```

Warning in mmpls(...): Interactions and/or factors skipped

Marginal Model Plots



```
m0<-glm(Audi ~ 1, family="binomial", data=df_train[-outs3,])
```

15.5 Bondad del ajuste y capacidad de predicción

Vamos a estudiar la bondad de nuestro modelo y su capacidad de predicción.

En primer lugar, vamos a empezar planteando la distribución del modelo de forma asimptótica con el test de chi-cuadrado.

```
Anova(m30)
```

Analysis of Deviance Table (Type II tests)

```
Response: Audi
```

```
LR Chisq Df Pr(>Chisq)
```

```

mpg                      156.907  1  < 2.2e-16 ***
age                     56.150   1  6.714e-14 ***
fuelType                 27.433   2  1.104e-06 ***
transmission              11.687   2   0.002899 **
engineSize_f                115.949   2  < 2.2e-16 ***
mpg:fuelType                  6.579   2   0.037266 *
mpg:transmission                13.177   2   0.001376 **
age:transmission                  8.778   2   0.012413 *
transmission:engineSize_f        14.457   4   0.005971 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```
1-pchisq(m30$deviance, m30$df.residual)
```

```
[1] 0.9656937
```

Como podemos ver, el p-valor de nuestra hipótesis nula es de 0.95, de modo que podemos aceptarla y afirmar que, en efecto, el modelo se ajusta bien a los datos.

Similarmente, si planteamos el estadístico de Pearson X², nos encontramos que en este caso, aplicando un intervalo de confianza del 95%, deberíamos aceptar nuestra hipótesis nula y afirmar que el modelo se ajusta bien a los datos.

```
X2m30<-sum((resid(m30,"pearson")^2))
1-pchisq( X2m30, m30$df.res)
```

```
[1] 0.9808711
```

Si aplicamos el test de Pseudo R², que tiene un rol similar a la suma de los cuadrados de los residuos en una regresión clásica, podemos ver como existen claras discrepancias entre si podemos o no aceptar nuestra hipótesis nula.

```
library(DescTools)
```

```
Attaching package: 'DescTools'
```

```
The following object is masked from 'package:games':
```

```
Mode
```

```
The following object is masked from 'package:car':
```

```
Recode
```

```
PseudoR2(m30, which='all')
```

	McFadden	McFaddenAdj	CoxSnell	Nagelkerke	AldrichNelson
VeallZimmermann	7.536428e-02	6.572669e-02	7.480260e-02	1.162298e-01	7.213945e-02
		Efron	McKelveyZavoina	Tjur	AIC
	1.420670e-01	6.446708e-02	4.685963e-01	6.864508e-02	3.683742e+03
	BIC	logLik	logLik0	G2	
	3.802464e+03	-1.822871e+03	-1.971448e+03	2.971535e+02	

Sin embargo, debemos recordar que estos test no funcionan con conjuntos de datos agrupados, como pueden ser los que aparecen en nuestra variable engineSize, o en los factores que hemos incluido en nuestro modelo.

Si planteamos el test de Hoslem, podemos ver como el p-valor de la hipótesis nula es de 0.1336, y podemos aceptar nuestra hipótesis nula, afirmando que el modelo SÍ que se ajusta bien a los datos.

```
library(ResourceSelection)
```

ResourceSelection 0.3-5 2019-07-22

```
ll <- which( is.finite(df_test$engineSize_f) )
pred_test <- predict(m30, newdata=df_test[ll,], type="response")
ht <- hoslem.test(as.numeric(df_test$Audi[ll])-1, pred_test)
ht
```

Hosmer and Lemeshow goodness of fit (GOF) test

```
data: as.numeric(df_test$Audi[ll]) - 1, pred_test
X-squared = 12.334, df = 8, p-value = 0.1369
```

A continuación vamos a general la curva de ROC que nos ayudará a ver de manera gráfica la bondad del ajuste del modelo.

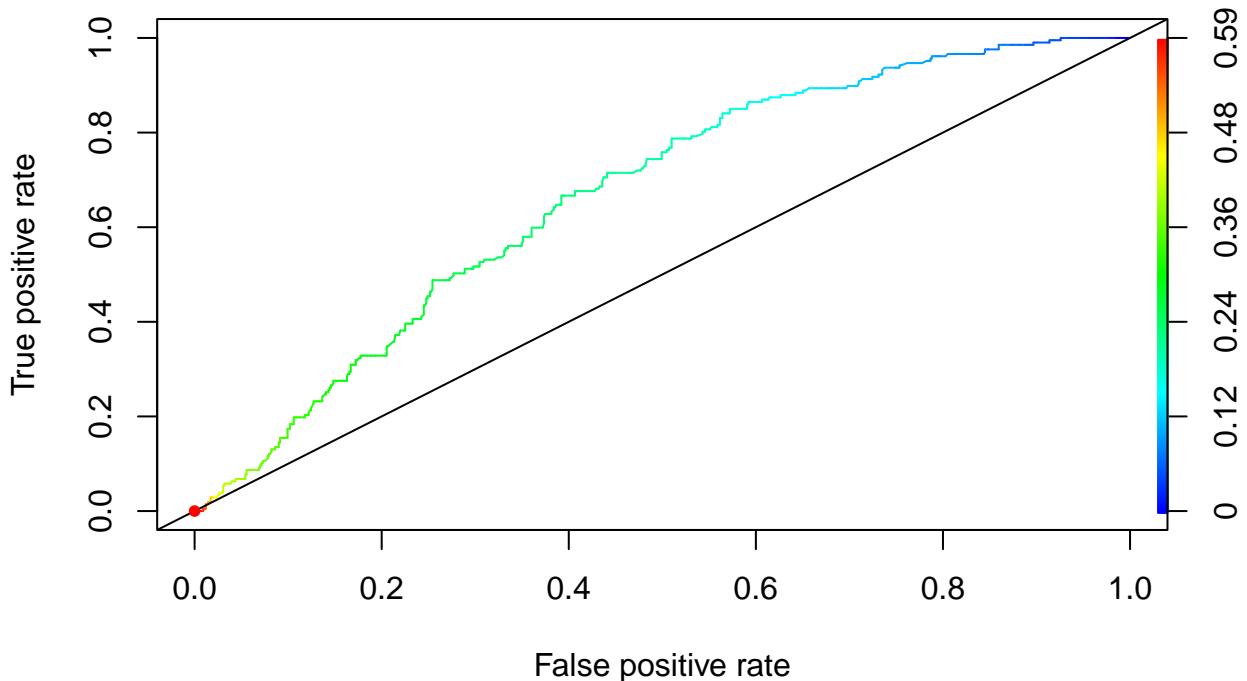
De manera indicativa, un modelo excelente, se acercaría mucho al punto (0,1), mientras que un modelo que se acerca a la recta con $y = x$ sería un modelo malo.

```
pred <- prediction(pred_test, df_test$Audi[ll])
perf <- performance(pred, measure="tpr", x.measure="fpr")

plot(perf, colorize=TRUE, type="l")
abline(a=0, b=1)

# Área bajo la curva
AUC <- performance(pred, measure="auc")
AUCcultura <- AUC@y.values

# Punto de corte óptimo
cost.perf <- performance(pred, measure ="cost")
opt.cut <- pred@cutoffs[[1]][which.min(cost.perf@y.values[[1]])]
#coordenadas del punto de corte óptimo
x<-perf@x.values[[1]][which.min(cost.perf@y.values[[1]])]
y<-perf@y.values[[1]][which.min(cost.perf@y.values[[1]])]
points(x,y, pch=20, col="red")
```



Como podemos ver, nuestro modelo se acerca más a la recta $x=y$ que al punto $(0,1)$, indicando que es bastante mejorable.

Vamos a analizar algunos valores característicos de esta curva.

```
# Área bajo la curva
AUC      <- performance(pred, measure="auc")
AUCcultura <- AUC@y.values

cat("AUC:", AUCcultura[[1]]);
```

AUC: 0.6670666

```
cat("Punto de corte óptimo:", opt.cut)
```

Punto de corte óptimo: Inf

Podemos ver que el área bajo la curva es de 0.687, indicando que es un modelo bastante malo. Además, el punto de corte óptimo se sitúa en el $(0,0)$ (No se muy bien como interpretar este resultado, pero muy positivo no debe ser...)

15.6 Matriz de confusión

Vamos a generar la matriz de confusión del modelo que hemos planteado.

```
audi.est <- ifelse(pred_test<0.4, 0, 1)
tt<-table(audi.est,df_test$Audi[11]); tt;
```

	Audi	No Audi
0	727	195
1	28	12

Si aplicamos las definiciones: Sensibilidad: $12 / (12 + 184) = 0.06$ Especificidad: $726 / (28 + 721) = 0.97$

Con estos dos conceptos podemos concluir que el modelo responde que NO a casi todo. Vemos como de las 210 observaciones que SÍ que eran Audi, solo ha respondido correctamente al 5%. Por otro lado, vemos como ha acertado casi todos los NO Audi...

Finalmente, si estudiamos la tasa de acierto de nuestro modelo, podemos ver que con el conjunto de validación ha acertado el 77.68% de las veces.

```
100*sum(diag(tt))/sum(tt)
```

```
[1] 76.81913
```