

Pricing

2023-01-08

Realizado por : Jose Delgado Serrano (Grupo 1) GitHub: <https://github.com/Josdelser/PreciosMadrid-FID>
Kaggle: <https://www.kaggle.com/datasets/mapecode/madrid-province-rent-data>

Paquetes y librerías

```
# libraries
```

```
library(rpart)
```

```
library(rpart.plot)
```

```
library(rattle)
```

```
## Loading required package: tibble
```

```
## Loading required package: bitops
```

```
## Rattle: A free graphical interface for data science with R.
```

```
## Versión 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
```

```
## Escriba 'rattle()' para agitar, sacudir y rotar sus datos.
```

```
library(dplyr)
```

```
##
```

```
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
```

```
##
```

```
## filter, lag
```

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

```
##
```

```
## intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
library(RColorBrewer)
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
library(ggfortify)
```

```
library(readr)
```

```
library(factoextra)
```

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

```
require(corrplot)
```

```
## Loading required package: corrplot
```

```
## corrplot 0.92 loaded
```

```
set.seed(1)
```

Lectura del dataset

```
precios_madrid <- read.csv("PreciosMadrid.csv")  
head(precios_madrid)
```

```
##      web_id                                     url  
## 1 99439319 https://www.idealista.com/en/inmueble/99439319/  
## 2 99439586 https://www.idealista.com/en/inmueble/99439586/  
## 3 99439169 https://www.idealista.com/en/inmueble/99439169/  
## 4 26925909 https://www.idealista.com/en/inmueble/26925909/  
## 5 99440018 https://www.idealista.com/en/inmueble/99440018/  
## 6 99440142 https://www.idealista.com/en/inmueble/99440142/  
##                                     title      type price deposit  
## 1      Flat / apartment for rent in pablo luna, 4      Flat  1400      NA  
## 2      Penthouse for rent in calle de Bolivia Penthouse  1300      1  
## 3      Duplex for rent in calle de la constancia Duplex   950      1  
## 4 Flat / apartment for rent in Urb. el viso, El Viso      Flat  2975      1  
## 5      Studio flat for rent in luis cabrera      Studio   650      1  
## 6 Flat / apartment for rent in calle de Nieremberg      Flat  1200      NA  
## private_owner professional_name floor_built floor_area floor year_built  
## 1      False Silcasas Ochocientas      60      NA      3rd      1954  
## 2      False      Cruzity      77      NA      6th      1961  
## 3      False      Mm Home      72      68      3rd      1999  
## 4      False      B&H Partners      160      NA      3rd      NA  
## 5      False Madrid en Propiedad      30      NA      4th      NA  
## 6      False Extra Inmobiliaria      54      47      4th      2009  
## orientation bedrooms bathrooms second_hand lift garage_included furnished  
## 1      2      1      True True      False      True  
## 2      2      2      True True      False      True  
## 3      east      1      1      True True      False      True  
## 4      west      4      3      True True      True      True  
## 5      0      1      True True      False      True  
## 6      west      1      1      True True      False      True  
## equipped_kitchen fitted_wardrobes air_conditioning terrace balcony storeroom  
## 1      True      True      True      True      False      False  
## 2      True      False      False      False      False      False  
## 3      True      True      True      True      False      False  
## 4      True      True      True      True      False      False  
## 5      True      True      False      False      True      False  
## 6      True      True      True      False      False      True  
## swimming_pool garden_area  
## 1      False      False  
## 2      False      False  
## 3      False      False
```

```
## 4      True      False
## 5      False     False
## 6      True      False
##
## 1      pablo luna, 4, Subdistrict Castilla, District Chamartín, Madrid, Madrid city,
## 2      Calle de Bolivia, Subdistrict Bernabéu-Hispanoamérica, District Chamartín, Madrid, Madrid city,
## 3      Calle de la constancia, Urb. no, Subdistrict Prosperidad, District Chamartín, Madrid, Madrid city,
## 4      Urb. el viso, Subdistrict El Viso, District Chamartín, Madrid, Madrid city,
## 5      luis cabrera, Subdistrict Prosperidad, District Chamartín, Madrid, Madrid city,
## 6      Calle de Nieremberg, Subdistrict Ciudad Jardín, District Chamartín, Madrid, Madrid city,
##      district      subdistrict postalcode last_update
## 1 Chamartín      Castilla      28046 7 November
## 2 Chamartín Bernabéu-Hispanoamérica      28016 7 November
## 3 Chamartín      Prosperidad      28002 7 November
## 4 Chamartín      El Viso      NA 7 November
## 5 Chamartín      Prosperidad      28002 7 November
## 6 Chamartín      Ciudad Jardín      28002 7 November
```

```
colnames(precios_madrid)
```

```
## [1] "web_id"      "url"      "title"
## [4] "type"      "price"      "deposit"
## [7] "private_owner" "professional_name" "floor_built"
## [10] "floor_area" "floor"      "year_built"
## [13] "orientation" "bedrooms" "bathrooms"
## [16] "second_hand" "lift"      "garage_included"
## [19] "furnished" "equipped_kitchen" "fitted_wardrobes"
## [22] "air_conditioning" "terrace" "balcony"
## [25] "storeroom" "swimming_pool" "garden_area"
## [28] "location" "district" "subdistrict"
## [31] "postalcode" "last_update"
```

Despues de analizar el dataset elijo las columnas que parecen mas interesantes

```
predata1 <- select(precios_madrid,price,floor_built,bathrooms,terrace,bedrooms,postalcode,garage_included)
head(predata1)
```

```
##      price floor_built bathrooms terrace bedrooms postalcode garage_included
## 1  1400      60      1      True      2      28046      False
## 2  1300      77      2     False      2      28016      False
## 3   950      72      1      True      1      28002      False
## 4  2975     160      3      True      4         NA      True
## 5   650      30      1     False      0      28002      False
## 6  1200      54      1     False      1      28002      False
```

```
colnames(predata1)
```

```
## [1] "price"      "floor_built" "bathrooms" "terrace"
## [5] "bedrooms" "postalcode" "garage_included"
```

Elimino pisos que cuesten 0, tengan 0 habitaciones,estén repetidos o sean NA. También para acotar el dataset vamos a coger solo los anuncios de 10 postalcode

```
predata2 = subset(predata1, price>0 & bedrooms>1 & postalcode>=28001 & postalcode<=28011)
predata2 <- na.omit(predata2)
predata <- unique(predata2)
```

Convertir las columnas de valores char("True","False") en num(1,0).Tambien estandarizo todo en numeric

```
predata$terrace <- as.numeric(as.logical(predata$terrace))
predata$garage_included <- as.numeric(as.logical(predata$garage_included))
predata$postalcode <- as.numeric(as.integer(predata$postalcode))
predata$price <- as.numeric(as.integer(predata$price))
predata$floor_built <- as.numeric(as.integer(predata$floor_built))
predata$bathrooms <- as.numeric(as.integer(predata$bathrooms))
predata$bedrooms <- as.numeric(as.integer(predata$bedrooms))
```

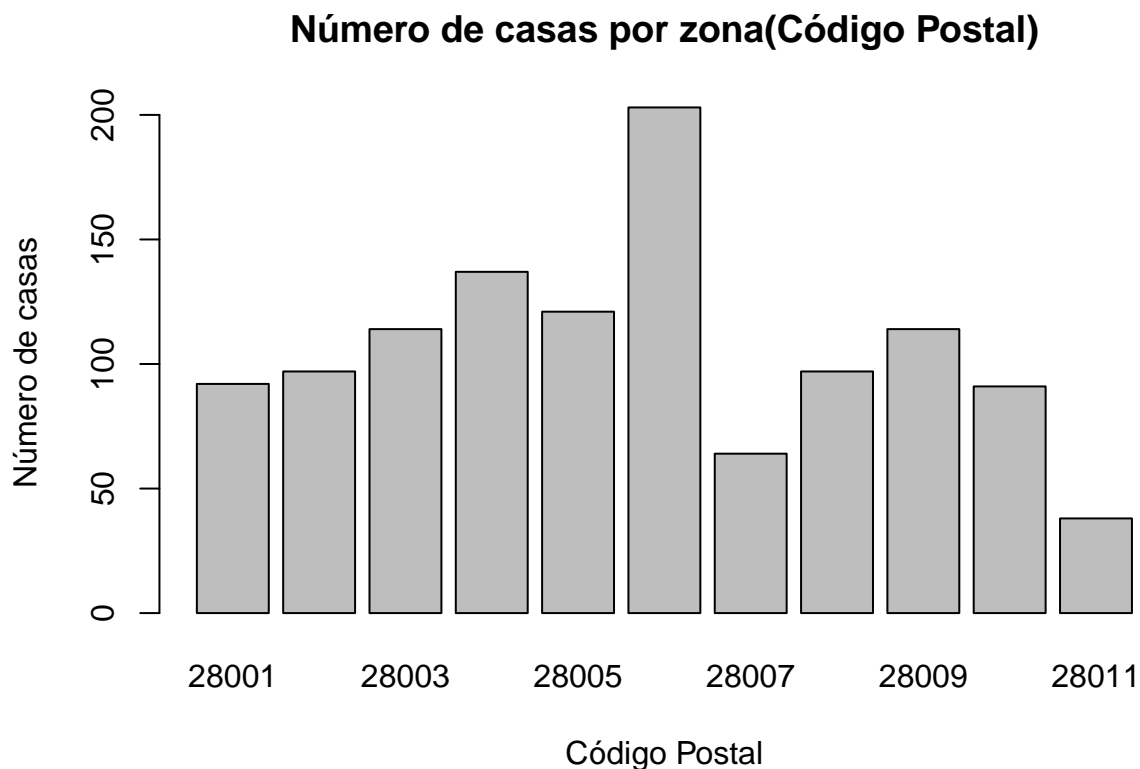
Finalmente despues del preprocesamiento de datos, obtenemos el dataset final

```
data<-predata
```

Una vez con los datos bien definido, pasamos a la visualización.

Primero vamos a ver el número de casas por código postal.

```
barplot(table(data$postalcode),
main="Número de casas por zona(Código Postal)",
xlab="Código Postal",
ylab="Número de casas",)
```



Vamos a etiquetar y categorizar según los metros construidos, con el objetivo de ver los tipos de viviendas

```
data_metros <- data
```

```
head(data_metros[order(data_metros$floor_built),])
```

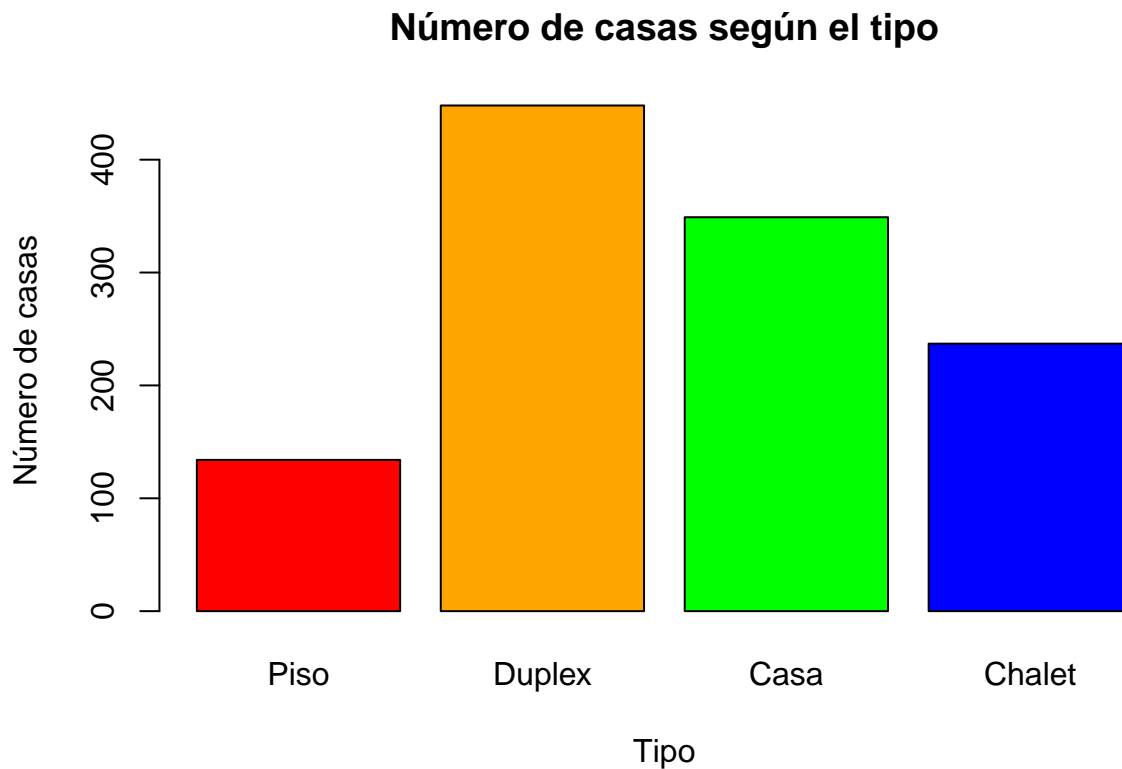
```
##      price floor_built bathrooms terrace bedrooms postalcode garage_included
## 7018   1750         30         1      0         2      28009          0
## 2497    800         35         1      0         2      28005          0
## 1376   1990         40         1      0         2      28009          0
## 1537    850         40         1      0         2      28009          0
## 1550    750         40         1      0         2      28009          0
## 1627    950         40         1      0         2      28010          0
```

```
rangos <- c(0,65,100,150,Inf)
```

```
values <- c('Piso','Duplex','Casa','Chalet')
```

```
data_metros$tipo <- cut(data_metros$floor_built, breaks = rangos, labels = values)
```

```
barplot(table(data_metros$tipo),
main="Número de casas según el tipo",
xlab="Tipo",
ylab="Número de casas",
col=c("red","orange","green","blue"),)
```



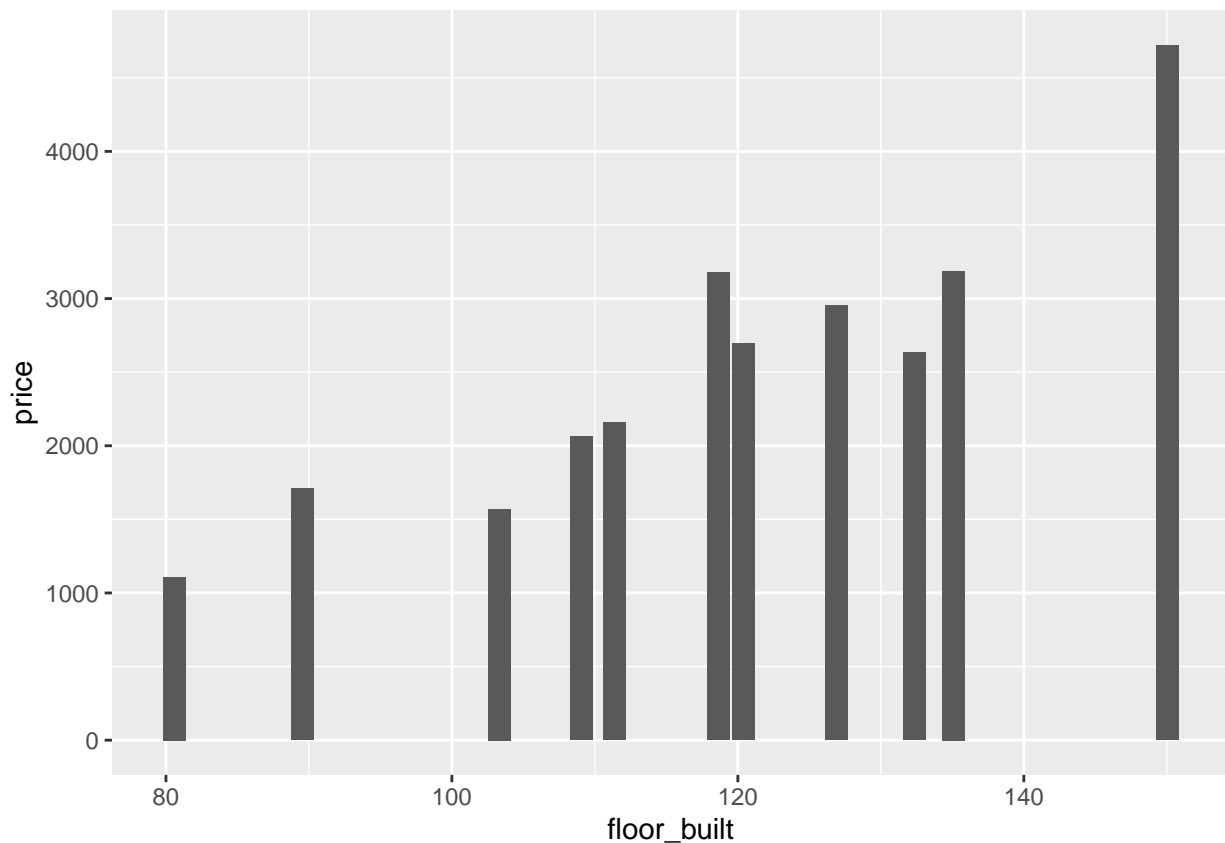
Vamos a calcular el precio medio, metros construidos y los tipos de casas dependiendo de la zona (Código postal). Para el tipo de casa pasamos a numeric con la asignación por defecto el 1 a pisos, 2 a duplex... De esta manera obtendremos una media del tipo de casa por zona

```
data_metros$tipo <- as.numeric(as.factor(data_metros$tipo))
medial <- aggregate(data_metros[, c(1,2,8)], list(data_metros$postalcode), mean)
medial
```

```
##      Group.1    price floor_built    tipo
## 1      28001 4720.315   150.02174 3.000000
## 2      28002 2155.979   111.34021 2.402062
## 3      28003 2632.711   132.33333 2.859649
## 4      28004 3177.555   118.61314 2.671533
## 5      28005 1710.264    89.57025 2.066116
## 6      28006 3186.567   135.08867 2.807882
## 7      28007 1570.141   103.35938 2.468750
## 8      28008 2063.856   109.04124 2.402062
## 9      28009 2694.211   120.39474 2.657895
## 10     28010 2953.846   126.87912 2.637363
## 11     28011 1108.816    80.57895 1.842105
```

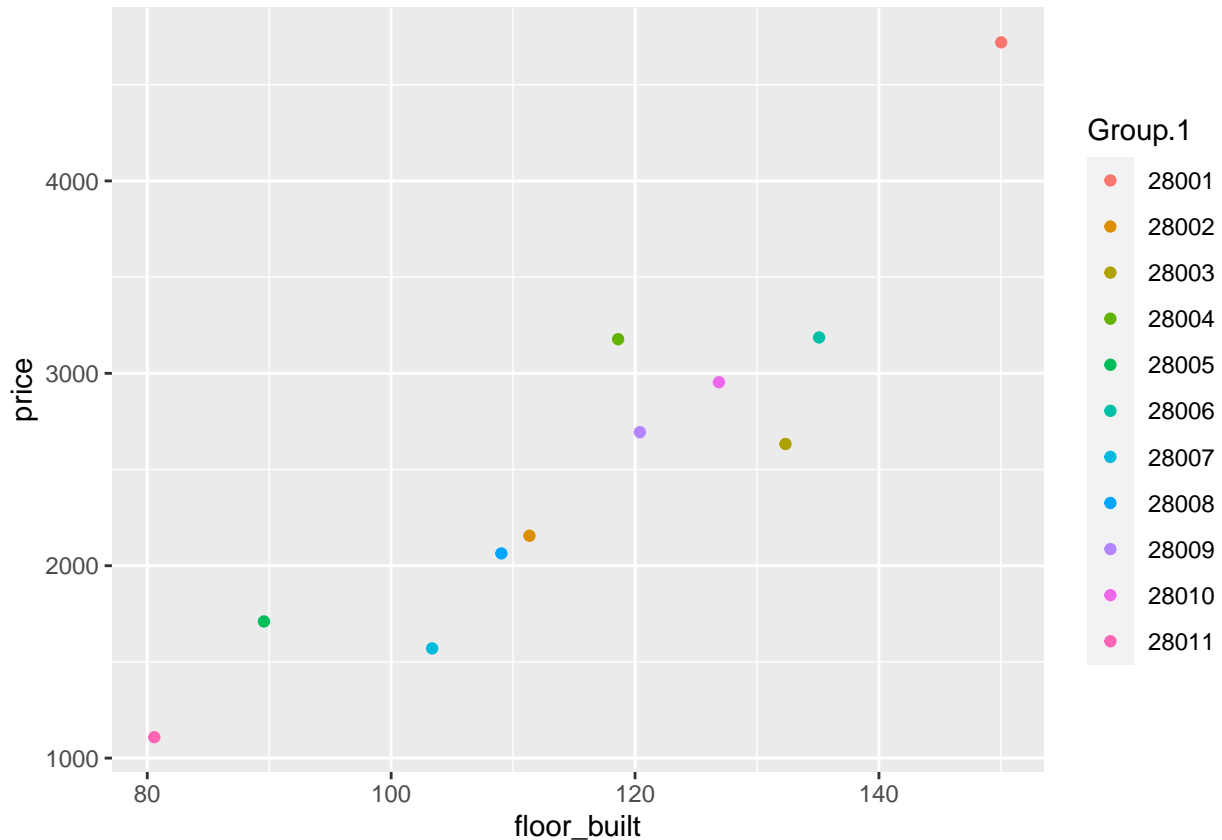
Vamos a graficar la media de precio según los metros construidos de media en barras, para ver si vemos algo interesante

```
ggplot(medial, aes(x = floor_built, y = price)) +
  geom_col()
```



Nos percatamos de que en algunos casos en los que no se cumple que a mayor metros mayor precio. Vamos a graficar la media de precio según los metros construidos de media por zonas en puntos

```
#Pasamos a factor el codigo postal para ver mejor el codigo de colores
media1$Group.1 <- as.factor(as.numeric(media1$Group.1))
ggplot(media1, aes(x= floor_built, y=price, colour=Group.1)) + geom_point()
```

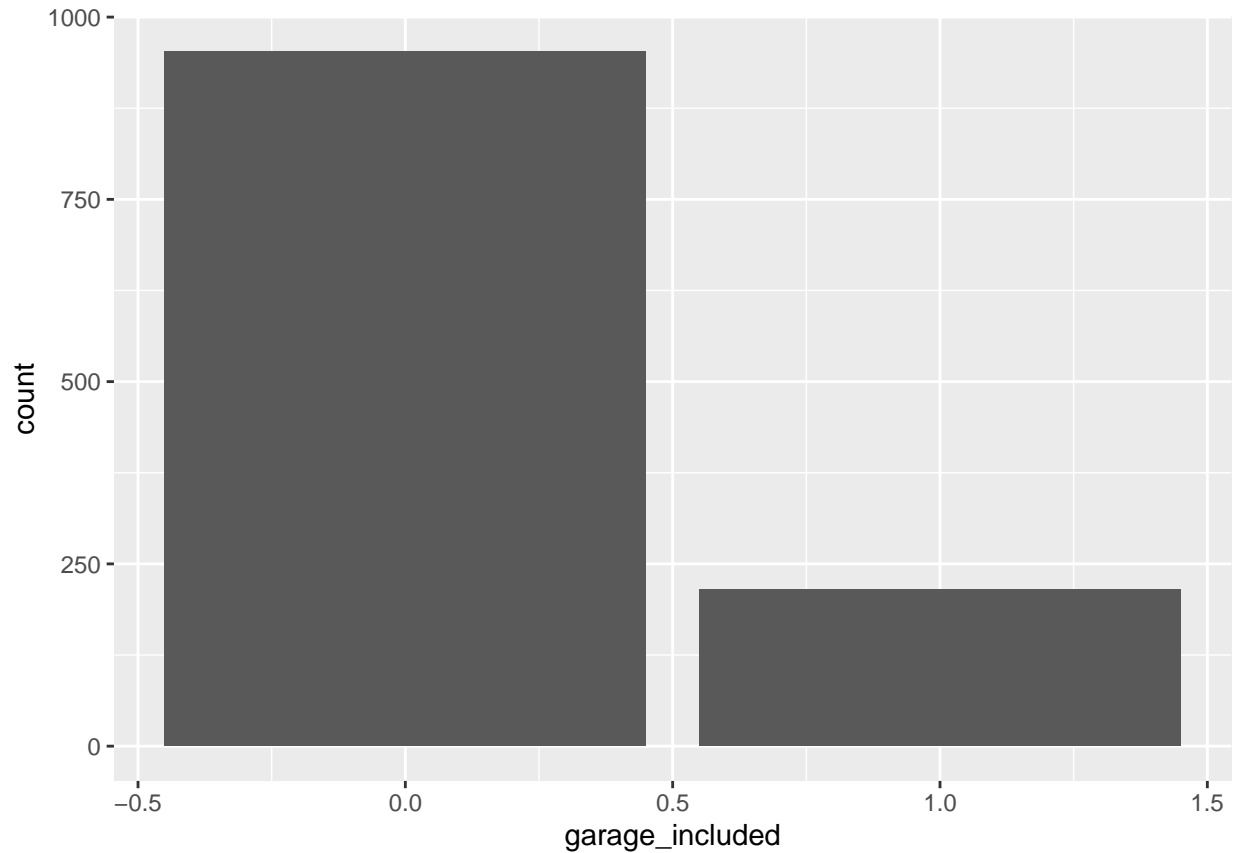


También vamos a hacer algunos analisis para ver como influyen otras variables, como el garaje, terraza. .

Gráfica para saber dependiendo del tipo de casa si lleva garaje o no. Podriamos hacer esto con las diferentes variables, solo habría que cambiar el valor del aes.

```
a<- ggplot(data_metros, aes(garage_included))
a + geom_bar(aes(fill = tipo))
```

```
## Warning: The following aesthetics were dropped during statistical transformation: fill
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a 'group' aesthetic or to convert a numerical
##   variable into a factor?
```

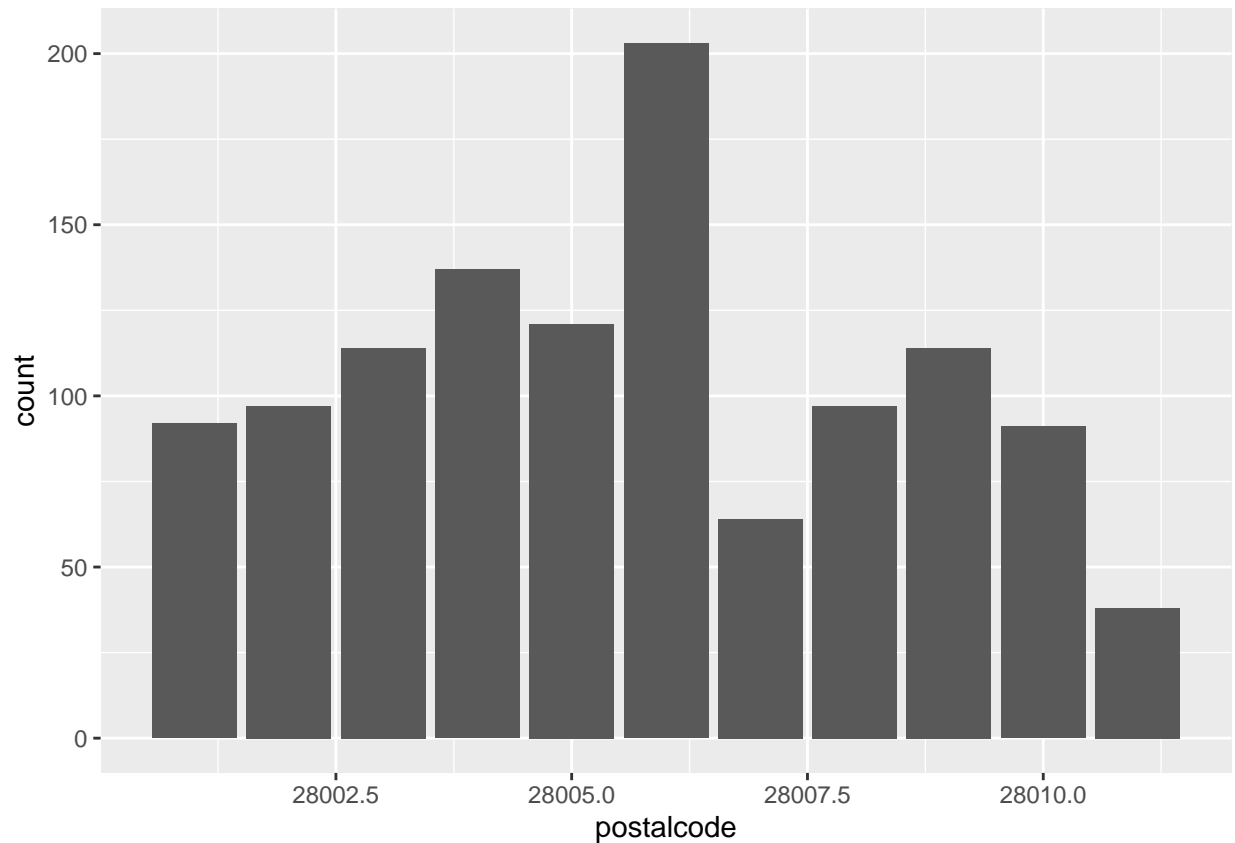


Gráfica según el número de tipos de casas por zona

```
a<- ggplot(data_metros, aes(postalcode))
```

```
a + geom_bar(aes(fill = tipo))
```

```
## Warning: The following aesthetics were dropped during statistical transformation: fill
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a 'group' aesthetic or to convert a numerical
##   variable into a factor?
```

Análisis supervisado:

Ahora vamos a realizar un árbol de decisión sobre la variable precio.

Primero partimos el dataset para tener datos de entrenamiento y datos de validación.

```
train <- createDataPartition(data$price, p = 0.7, list=FALSE)
data_train <- data[train,]
data_val <- data[-train,]
nrow(data_train)
```

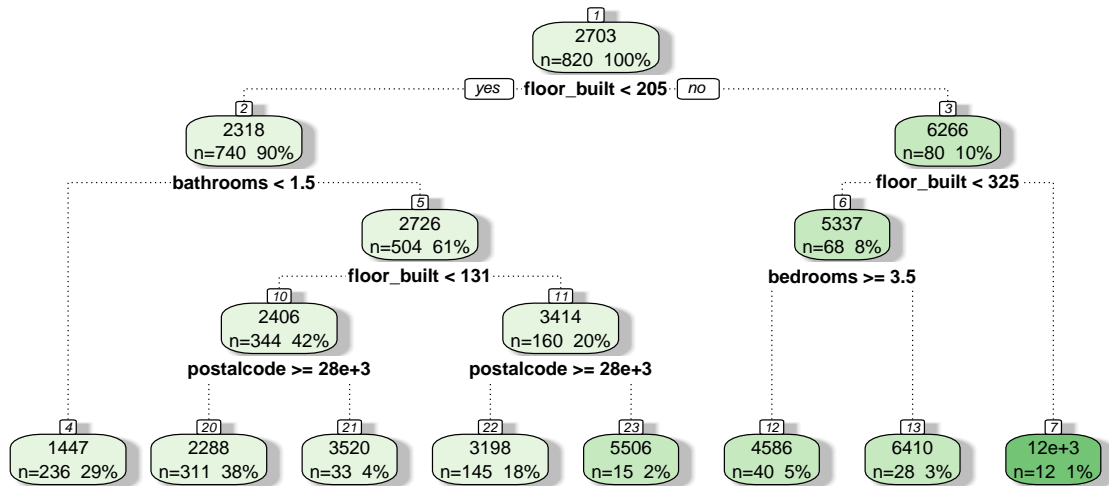
```
## [1] 820
```

```
nrow(data_val)
```

```
## [1] 348
```

Realizando el árbol podemos ver las diferentes variables que afectan a su precio y en que nos podemos basar para aproximar el precio medio de una vivienda.

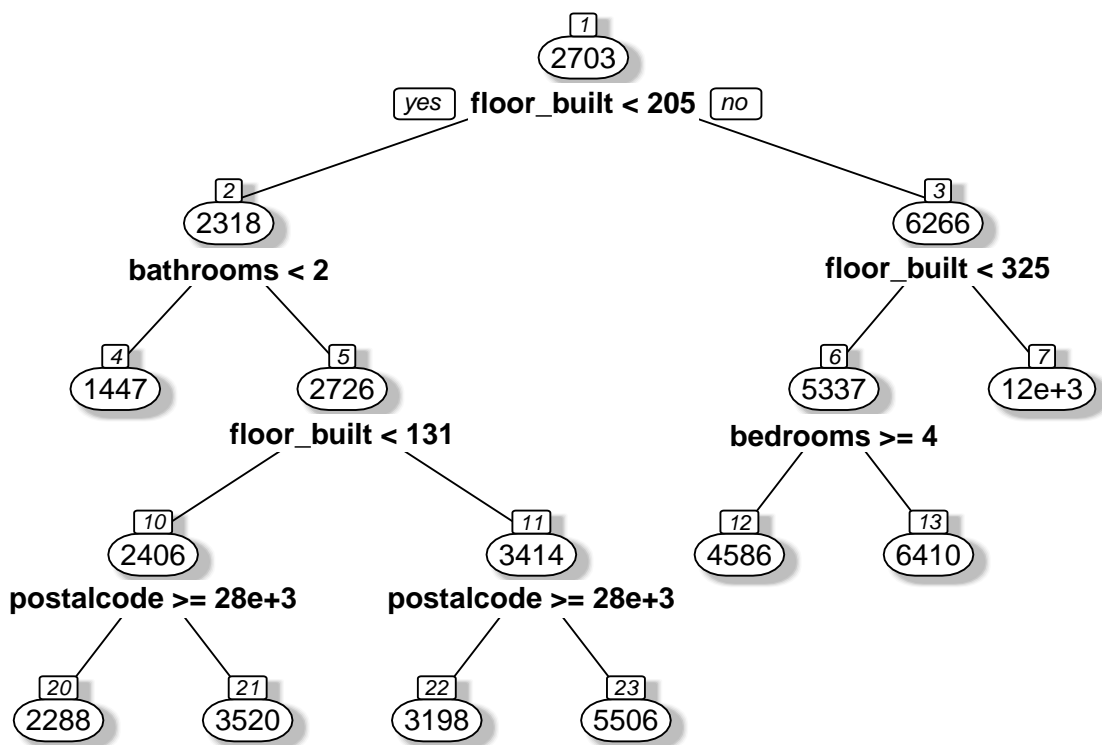
```
arbol <- rpart(formula = price ~ ., data = data_train)
fancyRpartPlot(arbol)
```



Rattle 2023-ene.-08 20:05:04 josed

Hacemos un par de cambios en la visualizacion para verlo mejor.

```
prp(arbol, type = 2, nn = TRUE,
    fallen.leaves = FALSE,
    varlen = 0, shadow.col = "gray")
```



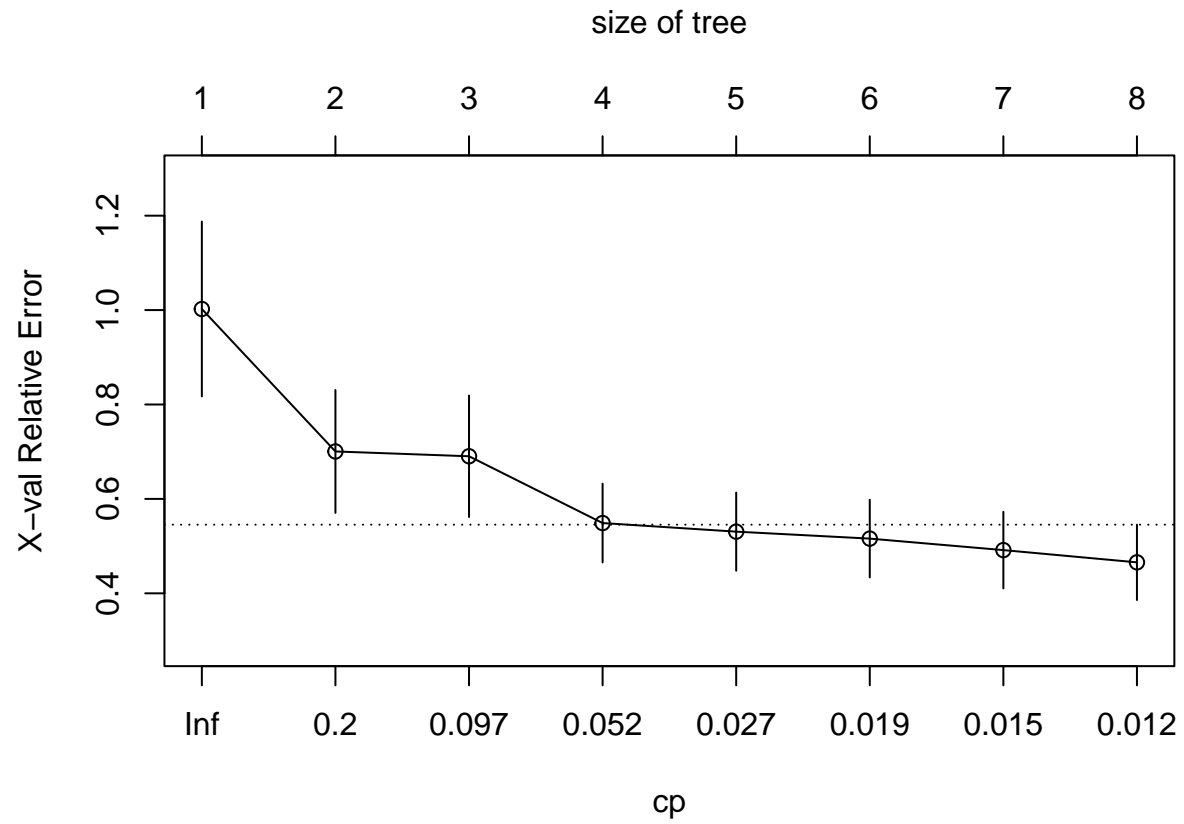
Observamos que los metros construidos y los cuartos de baños son las variables que mas afectan.

Vamos a ver el error relativo y el cp

```
arbol$cptable
```

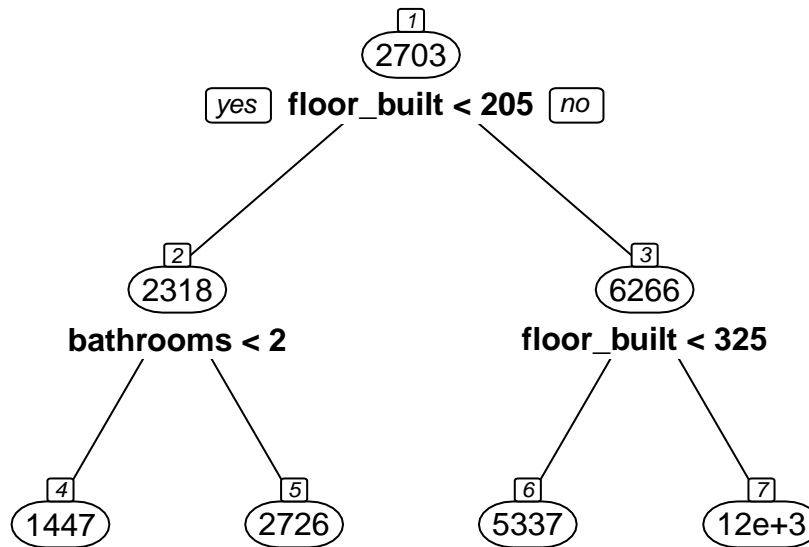
##	CP	nsplit	rel error	xerror	xstd
## 1	0.33989326	0	1.0000000	1.0022770	0.18525854
## 2	0.11816270	1	0.6601067	0.7004840	0.13010449
## 3	0.07949750	2	0.5419440	0.6903374	0.12867676
## 4	0.03355712	3	0.4624465	0.5488270	0.08345104
## 5	0.02186820	4	0.4288894	0.5306323	0.08282386
## 6	0.01656338	5	0.4070212	0.5159217	0.08234627
## 7	0.01369404	6	0.3904578	0.4915559	0.08114578
## 8	0.01000000	7	0.3767638	0.4655977	0.07982682

```
plotcp(arbol)
```



Podamos el árbol para reducirlo, usando el cp obtenido anteriormente

```
arbol_podado <- prune(arbol, cp = 0.052)
prp(arbol_podado, type = 2, nn = TRUE,
    fallen.leaves = FALSE,
    varlen = 0)
```



Predecimos en el data de validacion

```
precio_pred <- predict(arbol, newdata = data_val)
```

```
precio_pred[1]
```

```
##      17
```

```
## 1446.576
```

```
data_val[1,1]
```

```
## [1] 1290
```

El valor predecido no difiere mucho del valor real como podemos ver.

Vamos a hacer predicciones con datos nuevo del precio según el arbol, para ello vamos a crear valores de pruebas.

```
price <- c(0,0,0,0,0)
floor_built<- c(134,134,234,123,100)
bathrooms<- c(1,3,1,2,3)
terrace<- c(0,0,1,0,0)
bedrooms<- c(2,2,4,2,2)
postalcode<- c(28002,28002,28003,28003,28008)
```

```
garage_included<- c(1,1,0,0,1)

data_test_nuevo <- data.frame(price,floor_built,bathrooms,terrace,bedrooms,postalcode,garage_included)

#Predecimos el precio segun el arbol
nuevo_precio_pred <- predict(arbol, newdata = data_test_nuevo)

nuevo_precio_pred
```

```
##          1          2          3          4          5
## 1446.576 3198.097 4585.525 2287.662 2287.662
```

Ahora pasamos a aplicar un metodo de regresion multiple para calcular el precio en funcion de las demas variables

```
regresion_mul <- lm(formula = price ~ ., data = data_train)

summary(regresion_mul)
```

```
##
## Call:
## lm(formula = price ~ ., data = data_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4689.6  -688.1  -155.4   487.6 12892.0
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.375e+06  4.490e+05   3.063  0.002265 **
## floor_built   2.689e+01  1.109e+00  24.252 < 2e-16 ***
## bathrooms     2.044e+02  5.417e+01   3.773  0.000173 ***
## terrace     -3.090e+02  1.034e+02  -2.988  0.002893 **
## bedrooms     -5.946e+02  7.358e+01  -8.082  2.31e-15 ***
## postalcode   -4.908e+01  1.603e+01  -3.061  0.002277 **
## garage_included -2.242e+02  1.187e+02  -1.889  0.059215 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1279 on 813 degrees of freedom
## Multiple R-squared:  0.5982, Adjusted R-squared:  0.5952
## F-statistic: 201.7 on 6 and 813 DF,  p-value: < 2.2e-16
```

Segun el R^2 el modelo puede explicar en un 59% la variabilidad del precio, ya que $R^2=0,5982$. Tambien vemos que alguna variable predictor está relacionada con el precio ya que el p-value es bastante infimo.. No encontramos ninguna variable con un p-value alto por lo que nos indican que todas contribuyen en parte al modelo.

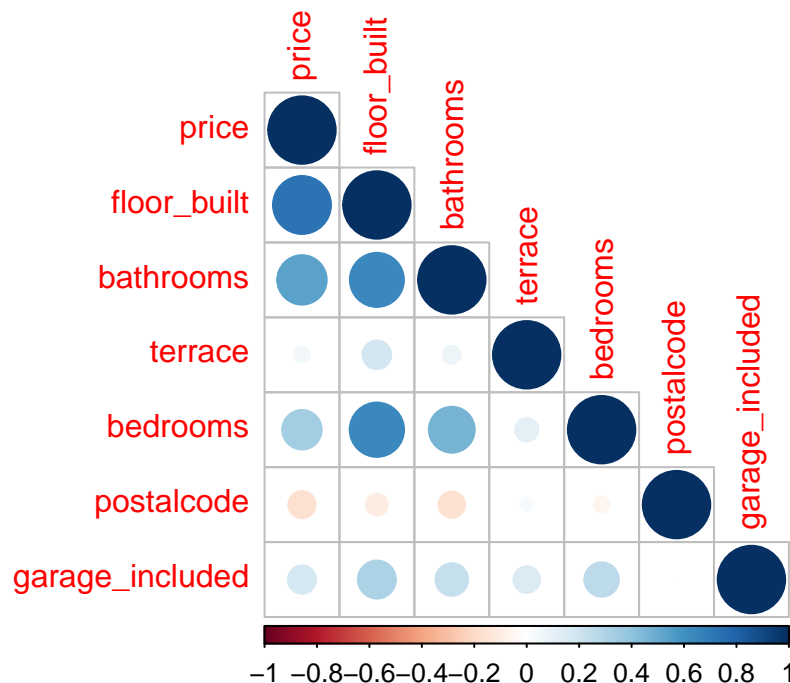
El metodo step, nos arroja que los metros construidos y los cuarto de baños son las variables que mas correlacion tienen. Los

```
step(regresion_mul, direction = "both", trace = 0)
```

```
##
## Call:
## lm(formula = price ~ floor_built + bathrooms + terrace + bedrooms +
##     postcode + garage_included, data = data_train)
##
## Coefficients:
## (Intercept)      floor_built      bathrooms      terrace
## 1375126.42         26.89         204.39        -308.97
## bedrooms      postcode  garage_included
## -594.64        -49.08        -224.19
```

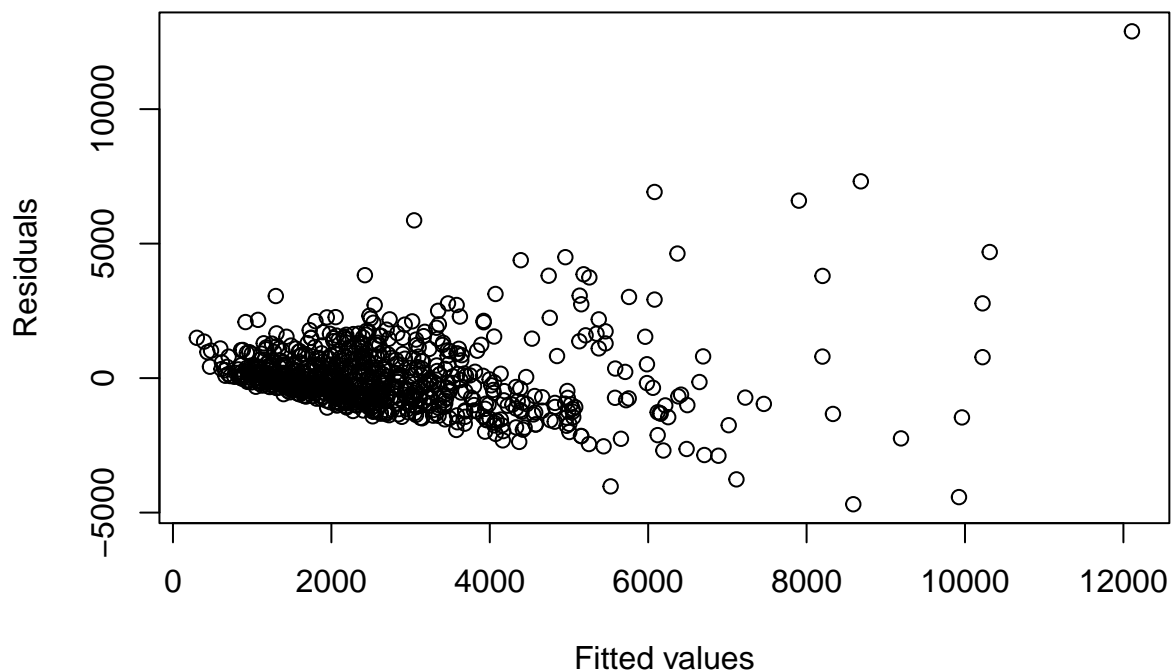
Hacemos un grafico corrplot para ver la correlacion con las variables, donde sacamos que las variables que mas influyen son los metros construidos, los cuarto de baños y las camas. Vemos según diferentes métodos que esas son las variables con mayor correlacion.

```
corrplot(cor(data_train) ,type = "lower",)
```



Ahora pasamos a graficar los residuos en funcion de los valores ajustados, es decir distancias entre los estimados y reales.

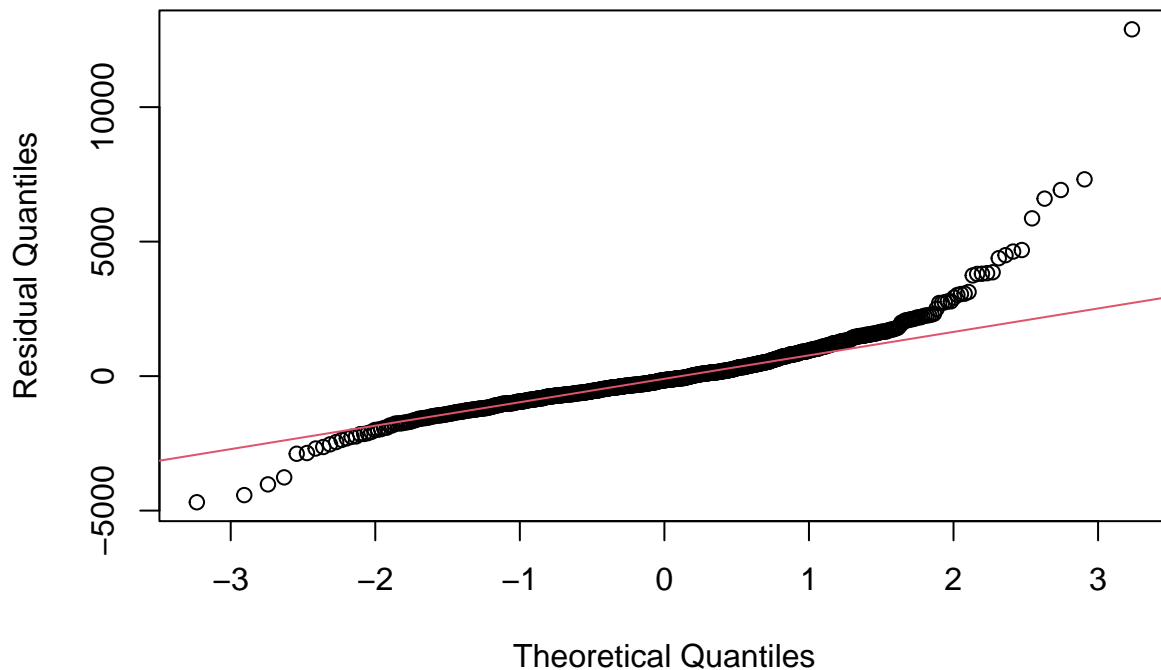
```
plot(regresion_mul$fitted.values, regresion_mul$residuals,
     xlab = "Fitted values", ylab = "Residuals")
```



Sacamos el grafico Q-Q para comprar los residuos de dos distribuciones de probabilidad cuando trazamos los cuantiles entre ellos. No apreciamos ningun patron y podemos intuir que esta formanod una linea por lo que concluimos con que el modelo es bueno.

```
qqnorm(regresion_mul$residuals, ylab = "Residual Quantiles")  
qqline(regresion_mul$residuals, col = 2)
```


Normal Q-Q Plot



Y vamos a predecir según la regresion multiple, luego compararemos con el arbol de decision.

```
data_val_regresion <-predict(regresion_mul, data_val)
predict(regresion_mul,data_test_nuevo)
```

```
##          1          2          3          4          5
## 3309.420 3718.206 4675.591 3393.100 2509.388
```

```
data_val_regresion[1]
```

```
##          17
## 1543.52
```

```
data_val[1,1]
```

```
## [1] 1290
```

Primero vemos los datos arrojados por la prediccion en los valores de pruebas creados anteriormente, no distan muchos del de los aboles. Luego, al igual que con los arboles, comparamos el valor del precio real con el calculado con la regresion, vemos que está mas lejano que los arboles y por tanto ha hecho peor prediccion

No supervisado: Clustering con k-means

Primero pasamos a ver que las variables sean numericas

```
lapply(data,class)
```

```
## $price
## [1] "numeric"
##
## $floor_built
## [1] "numeric"
##
## $bathrooms
## [1] "numeric"
##
## $terrace
## [1] "numeric"
##
## $bedrooms
## [1] "numeric"
##
## $postalcode
## [1] "numeric"
##
## $garage_included
## [1] "numeric"
```

Comprobamos si necesitamos escalarlas.

```
summary(data)
```

```
##      price      floor_built      bathrooms      terrace
## Min.   : 725   Min.   : 30.0   Min.   : 1.000   Min.   :0.0000
## 1st Qu.:1450   1st Qu.: 80.0   1st Qu.: 1.000   1st Qu.:0.0000
## Median :2200   Median :101.0   Median : 2.000   Median :0.0000
## Mean   :2698   Mean   :119.6   Mean   : 2.016   Mean   :0.2765
## 3rd Qu.:3350   3rd Qu.:140.0   3rd Qu.: 2.000   3rd Qu.:1.0000
## Max.   :25000   Max.   :512.0   Max.   :20.000   Max.   :1.0000
## bedrooms      postalcode      garage_included
## Min.   :2.000   Min.   :28001   Min.   :0.0000
## 1st Qu.:2.000   1st Qu.:28003   1st Qu.:0.0000
## Median :2.000   Median :28006   Median :0.0000
## Mean   :2.643   Mean   :28006   Mean   :0.1841
## 3rd Qu.:3.000   3rd Qu.:28008   3rd Qu.:0.0000
## Max.   :6.000   Max.   :28011   Max.   :1.0000
```

Vemos que hay mucha diferencia entre el maximo y el minimo por lo cual sería necesario escalarlo, para tener una mejor vision global

```
data_scaled = scale(data)
summary(data_scaled)
```

```
##      price      floor_built      bathrooms      terrace
## Min.   :-1.0237   Min.   :-1.4362   Min.   :-0.97777   Min.   :-0.618
## 1st Qu.: -0.6476   1st Qu.: -0.6350   1st Qu.: -0.97777   1st Qu.: -0.618
```

## Median :-0.2584	Median :-0.2985	Median :-0.01565	Median :-0.618
## Mean : 0.0000	Mean : 0.0000	Mean : 0.00000	Mean : 0.000
## 3rd Qu.: 0.3382	3rd Qu.: 0.3265	3rd Qu.: -0.01565	3rd Qu.: 1.617
## Max. :11.5707	Max. : 6.2877	Max. :17.30258	Max. : 1.617
## bedrooms	postalcode	garage_included	
## Min. :-0.7999	Min. :-1.6429	Min. :-0.4748	
## 1st Qu.: -0.7999	1st Qu.: -0.9334	1st Qu.: -0.4748	
## Median :-0.7999	Median : 0.1309	Median :-0.4748	
## Mean : 0.0000	Mean : 0.0000	Mean : 0.0000	
## 3rd Qu.: 0.4441	3rd Qu.: 0.8405	3rd Qu.: -0.4748	
## Max. : 4.1761	Max. : 1.9048	Max. : 2.1045	

Una vez los datos bien estructurados, pasamos a calcular el mejor valor de k. Para ello iremos iterando el valor de 1 a 20 para guardar el valor de compactación en cada iteración, luego lo graficaremos y veremos a qué k que produce un cambio considerable. Inicializamos la variable nstart a 10 para controlar la aleatoriedad.

```
v_compac <-0

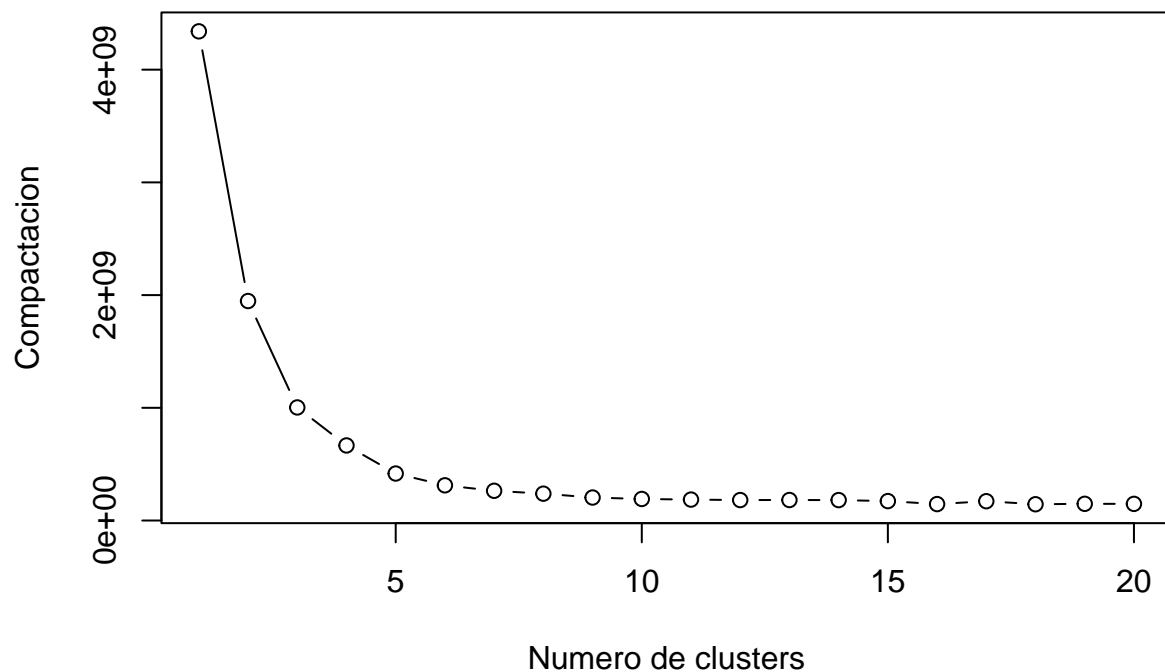
for(i in 1:20){

  km1<-kmeans(data,center=i,nstart=10)
  v_compac[i] <- km1$tot.withinss

}

par(mfrow = c(1,1))

plot(1:20, v_compac, type = "b",
     xlab = "Numero de clusters",
     ylab = "Compactacion")
```



Observamos que en k=3 y k=4 empieza a ver un cambio significativo, por lo cual escogemos estos valores para K. Pasamos a ver los resultados y comparamos

K=3

```
kmeans3 <- kmeans(data, center =3,nstart= 10)
kmeans3
```

```
## K-means clustering with 3 clusters of sizes 45, 757, 366
```

```
##
```

```
## Cluster means:
```

```
##      price floor_built bathrooms  terrace bedrooms postalcode garage_included
## 1 9491.111   265.88889   3.755556 0.4000000 3.488889   28003.42      0.3555556
## 2 1714.639    95.07662   1.634082 0.2721268 2.503303   28005.96      0.1677675
## 3 3897.115   152.42077   2.592896 0.2704918 2.827869   28005.23      0.1967213
##
```

```
## Clustering vector:
```

```
##      7  15  16  17  18  24  28  29  39  42  45  47  50  58  60  66
##      2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
##     71  73  76  82  83  95  96  98 107 108 120 122 129 133 136 137
##      2   2   2   2   2   2   2   2   2   2   2   2   2   3   2   2
##    140 144 160 162 164 172 173 189 193 204 207 209 210 222 224 225
##      2   2   2   2   2   3   2   2   2   2   2   3   2   2   1   1
##    226 227 237 240 246 248 250 252 257 266 268 270 284 286 288 315
##      1   3   2   3   2   2   2   2   2   2   2   2   3   1   1   2
##    320 321 330 331 332 343 347 356 366 367 368 373 377 378 382 401
##      2   2   2   2   1   2   2   2   3   2   2   2   2   2   2   2
```

##	403	412	418	428	429	432	441	442	443	448	449	485	489	499	501	502
##	2	3	2	3	2	1	2	3	2	2	2	2	2	2	2	2
##	509	517	524	533	535	545	550	552	555	566	568	569	572	575	579	589
##	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
##	590	599	600	603	606	613	628	636	642	643	664	668	673	682	690	693
##	2	2	2	2	2	2	2	2	2	2	2	2	2	3	2	2
##	702	704	705	709	710	713	718	721	725	733	742	1098	1173	1175	1176	1177
##	2	2	2	2	2	2	2	3	2	2	2	2	3	2	2	2
##	1179	1181	1182	1184	1188	1193	1199	1200	1203	1204	1205	1212	1214	1218	1220	1223
##	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
##	1228	1230	1231	1236	1239	1244	1245	1252	1264	1268	1269	1271	1272	1273	1277	1279
##	2	2	3	2	2	2	2	2	3	2	2	2	3	3	2	2
##	1282	1285	1296	1301	1305	1307	1317	1318	1319	1321	1325	1327	1328	1333	1335	1340
##	2	2	2	2	3	2	3	2	2	2	2	3	3	2	2	2
##	1345	1348	1349	1356	1362	1366	1367	1374	1376	1378	1379	1384	1385	1391	1392	1397
##	2	2	2	2	3	3	2	2	2	2	2	2	3	2	2	2
##	1400	1401	1402	1403	1409	1414	1415	1418	1420	1421	1422	1427	1428	1429	1432	1436
##	2	2	2	3	2	2	2	2	2	2	2	2	3	2	2	2
##	1437	1438	1441	1450	1453	1454	1456	1461	1463	1468	1469	1473	1474	1476	1478	1481
##	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
##	1495	1496	1507	1508	1512	1515	1529	1530	1537	1540	1541	1545	1546	1547	1548	1550
##	3	2	2	2	2	2	2	2	2	2	2	2	3	2	2	2
##	1554	1560	1564	1566	1571	1575	1584	1586	1592	1594	1596	1598	1599	1600	1602	1603
##	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
##	1605	1610	1621	1622	1623	1625	1627	1629	1630	1631	1639	1643	1656	1665	1670	1671
##	2	2	2	2	2	3	2	2	2	2	2	2	2	2	2	2
##	1672	1674	1687	1692	1693	1695	1700	1710	1713	1716	1722	1724	1725	1727	1729	1733
##	2	2	2	2	2	2	3	2	2	2	3	2	2	2	2	3
##	1737	1741	1748	1749	1754	1756	1761	1763	1767	1768	1771	1773	1777	1780	1787	1793
##	2	2	2	3	3	3	3	2	3	2	2	3	2	2	3	3
##	1797	1801	1806	1807	1808	1812	1814	1816	1821	1822	1823	1829	1833	1834	1835	1836
##	2	2	2	3	3	2	3	3	3	3	3	3	3	2	3	3
##	1838	1840	1841	1853	1855	1857	1858	1861	1864	1867	1871	1874	1879	1882	1883	1885
##	2	2	2	1	3	2	3	3	2	3	2	3	3	3	2	3
##	1888	1889	1891	1892	1904	1905	1906	1908	1911	1913	1915	1918	1920	1923	1924	1930
##	3	2	2	3	3	3	3	3	3	3	3	3	3	2	2	3
##	1934	1938	1939	1942	1945	1946	1948	1949	1953	1954	1956	1957	1960	1962	1965	1968
##	2	2	2	3	2	3	3	3	2	3	2	3	2	2	2	3
##	1969	1970	1971	1972	1975	1976	1977	1978	1983	1990	1991	1993	2003	2007	2008	2012
##	3	1	3	2	3	2	1	2	2	2	2	2	3	2	3	3
##	2013	2016	2020	2021	2041	2045	2049	2056	2057	2063	2069	2070	2079	2082	2084	2086
##	2	3	3	2	3	2	2	2	2	3	2	3	2	2	3	2
##	2088	2090	2091	2093	2094	2099	2101	2103	2107	2108	2115	2116	2144	2147	2151	2164
##	2	2	2	3	3	3	2	2	2	3	3	3	2	3	2	2
##	2165	2167	2172	2175	2178	2184	2187	2190	2197	2204	2205	2208	2209	2213	2228	2230
##	2	3	2	3	3	2	3	3	3	2	2	3	2	2	3	2
##	2232	2241	2251	2269	2270	2272	2273	2283	2286	2287	2289	2300	2301	2305	2307	2317
##	2	3	2	2	2	2	3	2	2	3	2	2	2	3	2	2
##	2335	2337	2341	2346	2350	2351	2354	2358	2364	2368	2473	2482	2483	2485	2495	2496
##	3	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
##	2497	2499	2503	2507	2509	2511	2513	2519	2521	2530	2538	2549	2552	2557	2558	2563
##	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
##	2568	2570	2572	2579	2589	2599	2602	2605	2610	2612	2614	2620	2621	2623	2641	2645
##	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2

##	2661	2662	2670	2682	2689	2710	2712	2727	2739	2744	2746	2747	2751	2764	2766	2768
##	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
##	2771	3046	3595	4239	4646	4649	4652	4653	4655	4659	4661	4663	4664	4665	4670	4672
##	2	2	2	3	2	2	2	2	2	2	2	2	2	2	3	2
##	4678	4679	4680	4682	4697	4699	4700	4704	4706	4724	4731	4743	4745	4763	4768	4769
##	2	2	2	2	2	2	2	2	2	3	2	2	2	2	3	3
##	4770	4771	4772	4779	4783	4784	4792	4793	4794	4796	4797	4804	4805	4806	4809	4810
##	3	3	3	3	2	2	2	3	3	2	2	2	2	3	2	2
##	4826	4831	4833	4838	4843	4845	4850	4853	4859	4872	4873	4874	4875	4878	4879	4890
##	2	2	2	2	3	3	3	2	2	3	2	2	2	3	2	2
##	4898	4900	4905	4923	4930	4931	4935	4937	4938	4945	4952	4958	4959	4961	4965	4968
##	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
##	4977	4982	4983	4987	4994	5000	5007	5014	5016	5026	5031	5032	5033	5042	5043	5046
##	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1
##	5059	5069	5070	5071	5074	5075	5606	5609	5610	5613	5615	5619	5622	5624	5630	5635
##	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
##	5637	5642	5643	5644	5669	5677	5680	5684	5687	5690	5693	5694	5695	5700	5701	5706
##	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
##	5712	5715	5717	5719	5722	5724	5726	5736	5737	5742	5743	5749	6229	6235	6240	6246
##	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
##	6273	6277	6278	6279	6282	6288	6289	6291	6299	6305	6308	6316	6317	6319	6327	6329
##	3	3	1	2	2	3	2	3	2	2	2	3	2	2	3	3
##	6330	6331	6332	6336	6349	6350	6354	6363	6367	6376	6377	6378	6379	6384	6385	6387
##	2	3	3	1	2	2	2	3	3	3	3	3	1	2	3	3
##	6389	6397	6398	6400	6402	6408	6413	6418	6419	6426	6430	6432	6434	6436	6442	6448
##	2	2	2	2	2	2	2	2	3	3	2	3	3	1	2	2
##	6449	6452	6454	6458	6461	6463	6465	6468	6471	6475	6481	6482	6484	6486	6488	6496
##	3	2	2	3	3	2	3	3	1	3	2	2	2	3	3	3
##	6497	6500	6502	6503	6505	6506	6510	6511	6512	6513	6514	6515	6516	6517	6518	6519
##	3	3	3	3	3	3	3	1	1	3	1	2	3	3	2	3
##	6520	6522	6524	6525	6526	6527	6530	6532	6533	6534	6536	6537	6538	6542	6543	6546
##	3	3	3	2	2	3	3	3	1	3	2	3	3	3	3	2
##	6549	6551	6552	6553	6554	6556	6557	6558	6566	6568	6569	6570	6573	6575	6576	6578
##	3	3	3	3	3	3	3	2	3	3	3	3	1	3	3	3
##	6580	6581	6582	6587	6589	6590	6591	6595	6596	6597	6598	6599	6600	6602	6603	6607
##	3	3	3	3	3	3	3	3	3	3	3	3	3	2	2	3
##	6612	6613	6617	6619	6620	6624	6629	6630	6631	6633	6635	6636	6645	6646	6648	6656
##	2	2	2	3	3	2	3	3	3	3	3	3	3	3	3	3
##	6657	6658	6660	6661	6662	6663	6665	6668	6670	6671	6672	6673	6675	6677	6678	6680
##	3	3	3	3	1	1	3	3	3	3	1	3	3	1	3	2
##	6681	6682	6683	6684	6685	6686	6687	6688	6693	6694	6697	6699	6700	6701	6703	6704
##	3	3	3	3	2	3	3	3	2	3	3	3	2	3	2	2
##	6708	6713	6714	6717	6718	6720	6740	6742	6743	6748	6749	6750	6752	6764	6773	6774
##	2	2	1	3	3	1	2	3	3	3	1	3	2	3	3	3
##	6775	6779	6780	6781	6786	6788	6789	6794	6795	6800	6801	6806	6817	6818	6819	6820
##	2	2	3	3	1	2	2	2	3	2	3	3	3	3	3	3
##	6823	6826	6831	6833	6839	6840	6841	6847	6859	6870	6873	6875	6877	6879	6880	6882
##	3	3	3	2	2	1	3	3	2	2	2	1	2	3	2	3
##	6884	6896	6900	6902	6903	6913	6916	6921	6938	6943	6960	6965	6966	6974	6975	6976
##	2	2	3	3	3	2	3	2	3	1	1	3	3	2	3	1
##	6981	6993	7004	7014	7015	7018	7043	7045	7049	7051	7064	7075	7076	7081	7086	7087
##	2	1	2	2	2	2	2	3	2	2	2	2	3	2	3	3
##	7088	7091	7099	7110	7116	7119	7125	7134	7145	7156	7158	7164	7172	7173	7174	7183
##	3	2	2	2	2	3	1	3	2	2	2	1	3	3	2	3

```

## 7184 7213 7218 7228 7233 7242 7250 7251 7260 7264 7267 7268 7283 7287 7289 7297
##      2      2      2      2      2      2      2      2      2      2      3      2      2      2      2      3
## 7300 7305 7306 7310 7325 7327 7331 7334 7880 7882 7883 7888 7895 7905 7916 7918
##      2      3      3      2      2      2      2      2      2      2      2      2      2      2      3      2
## 7920 7923 7929 7946 7949 7957 7964 7978 7999 8000 8016 8017 8022 8025 8028 8050
##      2      2      2      3      2      1      2      2      2      2      2      2      2      1      3      2
## 8051 8055 8059 8061 8069 8072 8082 8083 8084 8103 8111 8114 8115 8118 8120 8126
##      2      2      3      2      3      2      2      3      2      3      2      2      2      3      2      3
## 8142 8145 8148 8152 8173 8175 8176 8181 8184 8200 8201 8205 8206 8214 8228 8230
##      2      2      3      2      3      3      2      3      2      2      3      3      3      3      3      3
## 8232 8236 8242 8243 8260 8261 8262 8272 8280 8284 8290 8292 8293 8301 8303 8311
##      3      3      3      3      3      3      3      2      3      3      3      3      3      3      1      2
## 8312 8313 8314 8324 8332 8334 8339 8342 8345 8346 8352 8360 8365 8382 8383 8395
##      2      3      2      3      1      3      3      3      1      2      2      2      3      3      3      3
## 8397 8400 8405 8406 8407 8411 8412 8413 8420 8422 8423 8428 8437 8441 8442 8448
##      3      3      3      3      3      2      2      3      3      2      3      2      3      3      3      3
## 8453 8455 8457 8458 8468 8489 8500 8503 8509 8522 8535 8536 8540 8565 8566 8577
##      2      2      2      2      2      2      2      2      2      2      3      3      3      2      2      3
## 8578 8583 8590 8604 8621 8622 8645 8647 8649 8650 8653 8660 8672 8673 8680 8691
##      2      2      2      2      2      3      2      3      2      1      2      2      2      3      3      2
## 8692 8693 8699 8702 8715 8735 8743 8745 8747 8750 8755 8766 8768 8774 8776 8783
##      2      2      2      2      3      3      2      2      3      2      3      2      3      2      2      2
## 8789 8791 8796 8805 8814 8815 8817 8819 8831 8838 8840 8842 8843 8856 8870 8879
##      3      2      3      2      2      2      2      3      2      2      2      2      3      2      2      3
## 8882 8886 8891 8893 8901 8907 8911 8915 8918 8921 8924 8934 8938 8965 8968 8972
##      2      3      2      1      3      2      2      2      2      2      3      2      3      2      3      2
## 8989 8991 8998 9002 9004 9008 9030 9044 9057 9076 9095 9112 9121 9125 9132 9133
##      2      2      2      2      3      3      2      3      2      2      2      2      2      2      2      2
##
## Within cluster sum of squares by cluster:
## [1] 487898485 211998685 303430117
## (between_SS / total_SS = 76.9 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"

```

Ahora pasamos a graficarlo para ver los resultados

```

fviz_cluster(object = kmeans3, data = data, show.clust.cent = TRUE,
              ellipse.type = "euclid", star.plot = TRUE, repel = TRUE) +
  labs(title = "Resultados clustering K-means") +
  theme_bw() +
  theme(legend.position = "none")

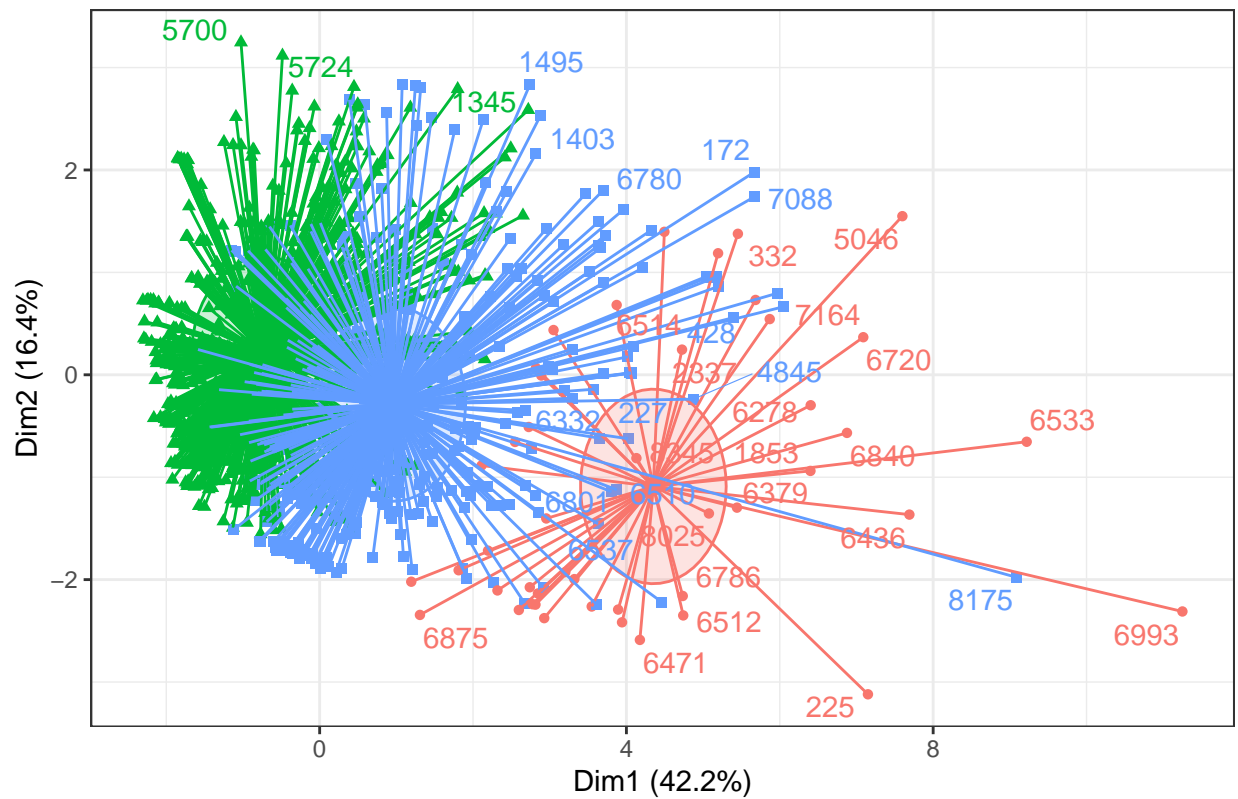
```

```

## Warning: ggrepel: 1132 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps

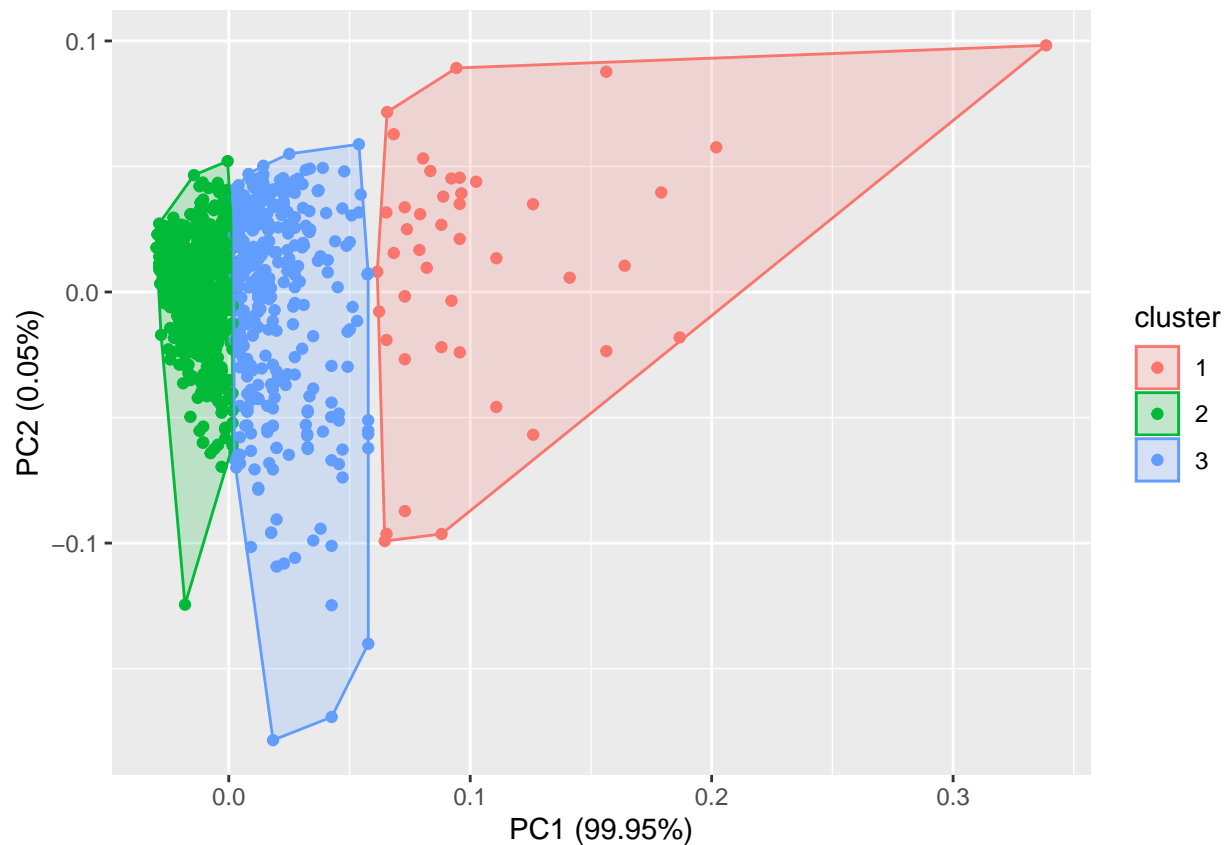
```

Resultados clustering K-means



Tenemos muchos valores, por lo que vamos a realizar otro grafico para verlo mas claro.

```
autoplot(kmeans3, data, frame=TRUE)
```

K=4

```
kmeans4 <- kmeans(data, center =4,nstart= 10)
kmeans4
```

K-means clustering with 4 clusters of sizes 630, 25, 112, 401

##

Cluster means:

	price	floor_built	bathrooms	terrace	bedrooms	postalcode	garage_included
## 1	1542.273	89.35397	1.546032	0.2619048	2.450794	28006.00	0.1619048
## 2	11160.000	296.68000	4.000000	0.5200000	3.680000	28003.68	0.5200000
## 3	5609.545	200.62500	3.089286	0.2232143	3.133929	28005.04	0.2321429
## 4	3173.379	133.52618	2.331671	0.2992519	2.743142	28005.34	0.1845387

##

Clustering vector:

	7	15	16	17	18	24	28	29	39	42	45	47	50	58	60	66
##	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
##	71	73	76	82	83	95	96	98	107	108	120	122	129	133	136	137
##	4	1	1	1	1	1	1	4	1	1	1	1	1	4	1	1
##	140	144	160	162	164	172	173	189	193	204	207	209	210	222	224	225
##	1	4	1	1	4	3	4	1	1	1	1	4	4	1	3	2
##	226	227	237	240	246	248	250	252	257	266	268	270	284	286	288	315
##	2	3	1	4	1	1	4	1	1	1	1	4	4	2	3	1
##	320	321	330	331	332	343	347	356	366	367	368	373	377	378	382	401
##	1	1	1	1	3	4	1	1	4	1	1	1	4	4	4	1
##	403	412	418	428	429	432	441	442	443	448	449	485	489	499	501	502

##	1	4	1	3	1	3	1	3	1	1	1	4	1	1	1	1
##	509	517	524	533	535	545	550	552	555	566	568	569	572	575	579	589
##	1	1	1	1	1	1	1	4	1	4	1	1	1	1	1	1
##	590	599	600	603	606	613	628	636	642	643	664	668	673	682	690	693
##	1	1	1	1	1	1	1	1	1	1	1	1	1	4	4	4
##	702	704	705	709	710	713	718	721	725	733	742	1098	1173	1175	1176	1177
##	4	1	1	1	1	1	1	4	4	1	1	1	4	1	1	1
##	1179	1181	1182	1184	1188	1193	1199	1200	1203	1204	1205	1212	1214	1218	1220	1223
##	1	1	1	1	1	4	1	1	1	1	1	1	1	1	1	1
##	1228	1230	1231	1236	1239	1244	1245	1252	1264	1268	1269	1271	1272	1273	1277	1279
##	1	4	4	4	1	1	1	1	4	1	1	1	4	4	1	1
##	1282	1285	1296	1301	1305	1307	1317	1318	1319	1321	1325	1327	1328	1333	1335	1340
##	4	1	1	1	4	1	4	4	1	4	1	4	4	1	4	4
##	1345	1348	1349	1356	1362	1366	1367	1374	1376	1378	1379	1384	1385	1391	1392	1397
##	1	1	1	1	4	4	1	1	1	1	1	1	4	1	1	1
##	1400	1401	1402	1403	1409	1414	1415	1418	1420	1421	1422	1427	1428	1429	1432	1436
##	1	1	1	4	1	1	1	1	1	4	1	1	4	1	1	1
##	1437	1438	1441	1450	1453	1454	1456	1461	1463	1468	1469	1473	1474	1476	1478	1481
##	1	1	1	1	1	1	1	1	1	4	1	1	1	1	1	1
##	1495	1496	1507	1508	1512	1515	1529	1530	1537	1540	1541	1545	1546	1547	1548	1550
##	4	1	1	1	1	1	4	1	1	1	1	1	4	1	1	1
##	1554	1560	1564	1566	1571	1575	1584	1586	1592	1594	1596	1598	1599	1600	1602	1603
##	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
##	1605	1610	1621	1622	1623	1625	1627	1629	1630	1631	1639	1643	1656	1665	1670	1671
##	1	4	1	1	1	3	1	1	1	1	1	4	1	1	4	1
##	1672	1674	1687	1692	1693	1695	1700	1710	1713	1716	1722	1724	1725	1727	1729	1733
##	1	1	1	1	1	4	4	1	1	1	4	1	1	1	1	4
##	1737	1741	1748	1749	1754	1756	1761	1763	1767	1768	1771	1773	1777	1780	1787	1793
##	4	4	1	4	4	4	4	4	4	4	1	3	1	4	3	4
##	1797	1801	1806	1807	1808	1812	1814	1816	1821	1822	1823	1829	1833	1834	1835	1836
##	1	4	1	4	3	1	4	3	4	4	3	3	3	4	4	3
##	1838	1840	1841	1853	1855	1857	1858	1861	1864	1867	1871	1874	1879	1882	1883	1885
##	4	4	4	2	4	1	4	4	4	4	1	3	4	3	1	4
##	1888	1889	1891	1892	1904	1905	1906	1908	1911	1913	1915	1918	1920	1923	1924	1930
##	4	4	1	4	4	4	4	3	4	4	3	4	4	1	1	3
##	1934	1938	1939	1942	1945	1946	1948	1949	1953	1954	1956	1957	1960	1962	1965	1968
##	1	1	1	4	4	4	4	3	1	4	4	4	1	1	1	4
##	1969	1970	1971	1972	1975	1976	1977	1978	1983	1990	1991	1993	2003	2007	2008	2012
##	4	3	3	1	3	4	3	1	1	1	1	1	4	1	4	4
##	2013	2016	2020	2021	2041	2045	2049	2056	2057	2063	2069	2070	2079	2082	2084	2086
##	1	4	3	1	4	1	4	1	1	4	1	4	1	1	3	1
##	2088	2090	2091	2093	2094	2099	2101	2103	2107	2108	2115	2116	2144	2147	2151	2164
##	1	1	1	3	4	4	1	1	1	4	4	4	1	4	1	1
##	2165	2167	2172	2175	2178	2184	2187	2190	2197	2204	2205	2208	2209	2213	2228	2230
##	1	4	1	4	4	1	4	4	4	1	4	4	1	1	4	1
##	2232	2241	2251	2269	2270	2272	2273	2283	2286	2287	2289	2300	2301	2305	2307	2317
##	1	4	4	4	1	1	4	1	1	4	1	1	1	4	1	1
##	2335	2337	2341	2346	2350	2351	2354	2358	2364	2368	2473	2482	2483	2485	2495	2496
##	4	2	1	1	4	1	1	1	1	1	1	1	1	1	1	1
##	2497	2499	2503	2507	2509	2511	2513	2519	2521	2530	2538	2549	2552	2557	2558	2563
##	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
##	2568	2570	2572	2579	2589	2599	2602	2605	2610	2612	2614	2620	2621	2623	2641	2645
##	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
##	2661	2662	2670	2682	2689	2710	2712	2727	2739	2744	2746	2747	2751	2764	2766	2768

##	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
##	2771	3046	3595	4239	4646	4649	4652	4653	4655	4659	4661	4663	4664	4665	4670	4672
##	1	1	1	4	4	1	1	1	1	1	1	1	1	1	4	4
##	4678	4679	4680	4682	4697	4699	4700	4704	4706	4724	4731	4743	4745	4763	4768	4769
##	4	1	1	1	1	1	1	1	1	4	1	1	1	1	4	4
##	4770	4771	4772	4779	4783	4784	4792	4793	4794	4796	4797	4804	4805	4806	4809	4810
##	4	4	4	4	1	1	1	4	4	1	1	1	1	3	1	1
##	4826	4831	4833	4838	4843	4845	4850	4853	4859	4872	4873	4874	4875	4878	4879	4890
##	1	4	1	1	3	4	4	1	1	4	4	1	1	4	4	1
##	4898	4900	4905	4923	4930	4931	4935	4937	4938	4945	4952	4958	4959	4961	4965	4968
##	1	1	1	1	1	1	1	1	1	1	1	1	4	4	1	1
##	4977	4982	4983	4987	4994	5000	5007	5014	5016	5026	5031	5032	5033	5042	5043	5046
##	1	1	4	1	1	1	1	1	1	1	1	1	1	1	1	2
##	5059	5069	5070	5071	5074	5075	5606	5609	5610	5613	5615	5619	5622	5624	5630	5635
##	1	4	4	1	1	1	1	1	1	1	1	1	1	1	1	1
##	5637	5642	5643	5644	5669	5677	5680	5684	5687	5690	5693	5694	5695	5700	5701	5706
##	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
##	5712	5715	5717	5719	5722	5724	5726	5736	5737	5742	5743	5749	6229	6235	6240	6246
##	1	1	1	1	1	1	1	1	1	1	1	1	4	1	1	1
##	6273	6277	6278	6279	6282	6288	6289	6291	6299	6305	6308	6316	6317	6319	6327	6329
##	4	4	2	4	1	3	1	4	4	4	1	4	4	1	4	4
##	6330	6331	6332	6336	6349	6350	6354	6363	6367	6376	6377	6378	6379	6384	6385	6387
##	4	3	4	2	1	1	1	3	3	4	4	4	2	1	3	3
##	6389	6397	6398	6400	6402	6408	6413	6418	6419	6426	6430	6432	6434	6436	6442	6448
##	4	1	1	1	1	1	1	4	4	4	1	3	4	2	1	4
##	6449	6452	6454	6458	6461	6463	6465	6468	6471	6475	6481	6482	6484	6486	6488	6496
##	4	1	4	4	3	1	4	4	2	4	4	1	4	4	4	4
##	6497	6500	6502	6503	6505	6506	6510	6511	6512	6513	6514	6515	6516	6517	6518	6519
##	4	4	4	4	3	4	3	3	2	4	2	4	4	4	4	4
##	6520	6522	6524	6525	6526	6527	6530	6532	6533	6534	6536	6537	6538	6542	6543	6546
##	4	4	4	1	1	4	4	4	2	4	4	4	4	3	3	1
##	6549	6551	6552	6553	6554	6556	6557	6558	6566	6568	6569	6570	6573	6575	6576	6578
##	4	3	4	4	3	4	3	1	3	3	4	4	3	4	3	4
##	6580	6581	6582	6587	6589	6590	6591	6595	6596	6597	6598	6599	6600	6602	6603	6607
##	3	4	4	4	4	3	3	4	3	3	3	3	3	1	1	3
##	6612	6613	6617	6619	6620	6624	6629	6630	6631	6633	6635	6636	6645	6646	6648	6656
##	4	1	1	4	3	1	3	4	4	4	4	4	4	4	3	4
##	6657	6658	6660	6661	6662	6663	6665	6668	6670	6671	6672	6673	6675	6677	6678	6680
##	3	3	3	4	3	3	4	4	3	3	2	3	4	2	4	4
##	6681	6682	6683	6684	6685	6686	6687	6688	6693	6694	6697	6699	6700	6701	6703	6704
##	4	4	4	4	4	4	3	4	4	4	4	4	1	4	4	1
##	6708	6713	6714	6717	6718	6720	6740	6742	6743	6748	6749	6750	6752	6764	6773	6774
##	1	4	3	4	3	2	4	3	4	4	3	4	1	4	4	3
##	6775	6779	6780	6781	6786	6788	6789	6794	6795	6800	6801	6806	6817	6818	6819	6820
##	1	4	4	4	3	4	4	4	4	4	3	4	3	4	4	4
##	6823	6826	6831	6833	6839	6840	6841	6847	6859	6870	6873	6875	6877	6879	6880	6882
##	3	4	4	1	1	2	4	3	4	4	4	3	1	4	1	3
##	6884	6896	6900	6902	6903	6913	6916	6921	6938	6943	6960	6965	6966	6974	6975	6976
##	4	4	4	3	4	4	4	1	4	3	3	4	4	1	4	3
##	6981	6993	7004	7014	7015	7018	7043	7045	7049	7051	7064	7075	7076	7081	7086	7087
##	1	2	1	1	1	1	1	4	1	1	1	1	4	1	4	4
##	7088	7091	7099	7110	7116	7119	7125	7134	7145	7156	7158	7164	7172	7173	7174	7183
##	3	1	1	4	1	4	2	3	1	4	1	2	4	4	4	4
##	7184	7213	7218	7228	7233	7242	7250	7251	7260	7264	7267	7268	7283	7287	7289	7297

```

##      4      4      4      1      1      4      1      1      1      4      3      1      1      4      4      4
## 7300 7305 7306 7310 7325 7327 7331 7334 7880 7882 7883 7888 7895 7905 7916 7918
##      1      4      4      4      1      1      1      1      4      1      1      1      1      1      4      1
## 7920 7923 7929 7946 7949 7957 7964 7978 7999 8000 8016 8017 8022 8025 8028 8050
##      1      1      1      4      1      3      1      1      1      1      1      1      1      2      4      4
## 8051 8055 8059 8061 8069 8072 8082 8083 8084 8103 8111 8114 8115 8118 8120 8126
##      1      1      4      1      4      1      1      3      1      4      1      1      4      3      4      4
## 8142 8145 8148 8152 8173 8175 8176 8181 8184 8200 8201 8205 8206 8214 8228 8230
##      1      1      3      4      4      4      1      3      4      1      4      4      4      4      4      3
## 8232 8236 8242 8243 8260 8261 8262 8272 8280 8284 8290 8292 8293 8301 8303 8311
##      4      3      4      4      3      4      4      1      4      4      4      3      4      4      3      1
## 8312 8313 8314 8324 8332 8334 8339 8342 8345 8346 8352 8360 8365 8382 8383 8395
##      1      4      4      4      2      3      3      4      2      1      1      1      4      3      3      4
## 8397 8400 8405 8406 8407 8411 8412 8413 8420 8422 8423 8428 8437 8441 8442 8448
##      3      4      3      4      4      4      1      4      4      4      4      4      4      4      3      4
## 8453 8455 8457 8458 8468 8489 8500 8503 8509 8522 8535 8536 8540 8565 8566 8577
##      1      1      1      4      1      1      4      4      1      1      4      4      4      1      1      4
## 8578 8583 8590 8604 8621 8622 8645 8647 8649 8650 8653 8660 8672 8673 8680 8691
##      1      1      1      1      1      4      1      4      1      3      1      4      1      4      4      1
## 8692 8693 8699 8702 8715 8735 8743 8745 8747 8750 8755 8766 8768 8774 8776 8783
##      1      1      1      1      3      4      1      4      4      1      4      1      4      1      1      1
## 8789 8791 8796 8805 8814 8815 8817 8819 8831 8838 8840 8842 8843 8856 8870 8879
##      4      1      4      1      4      1      1      4      1      1      1      1      4      1      1      4
## 8882 8886 8891 8893 8901 8907 8911 8915 8918 8921 8924 8934 8938 8965 8968 8972
##      1      4      1      2      4      1      4      1      1      1      3      1      4      1      4      1
## 8989 8991 8998 9002 9004 9008 9030 9044 9057 9076 9095 9112 9121 9125 9132 9133
##      1      1      1      1      4      4      1      4      4      1      1      1      1      1      1      1
##
## Within cluster sum of squares by cluster:
## [1] 97929284 327314347 120966510 119938239
## (between_SS / total_SS = 84.7 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"

```

Ahora pasamos a graficarlo para ver los resultados, como hemos hecho anteriormente

```

fviz_cluster(object = kmeans4, data = data, show.clust.cent = TRUE,
              ellipse.type = "euclid", star.plot = TRUE, repel = TRUE) +
  labs(title = "Resultados clustering K-means") +
  theme_bw() +
  theme(legend.position = "none")

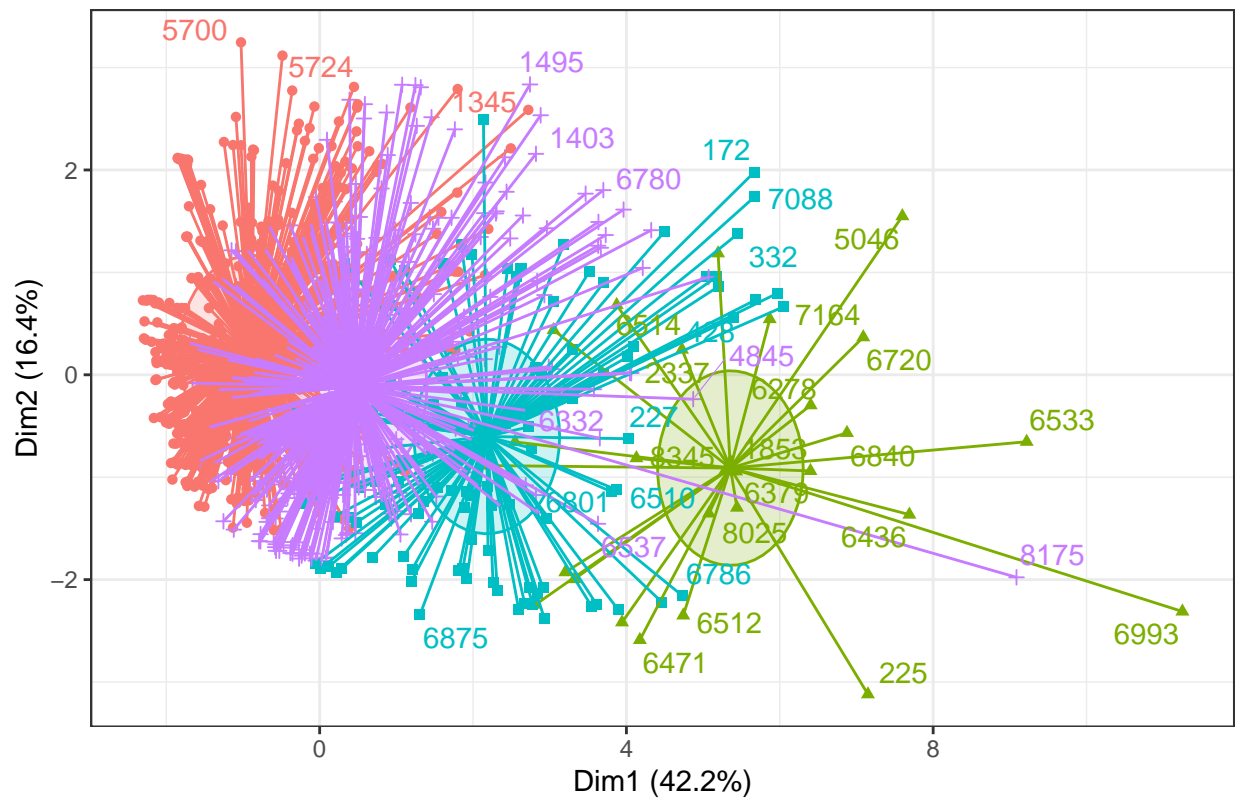
```

```

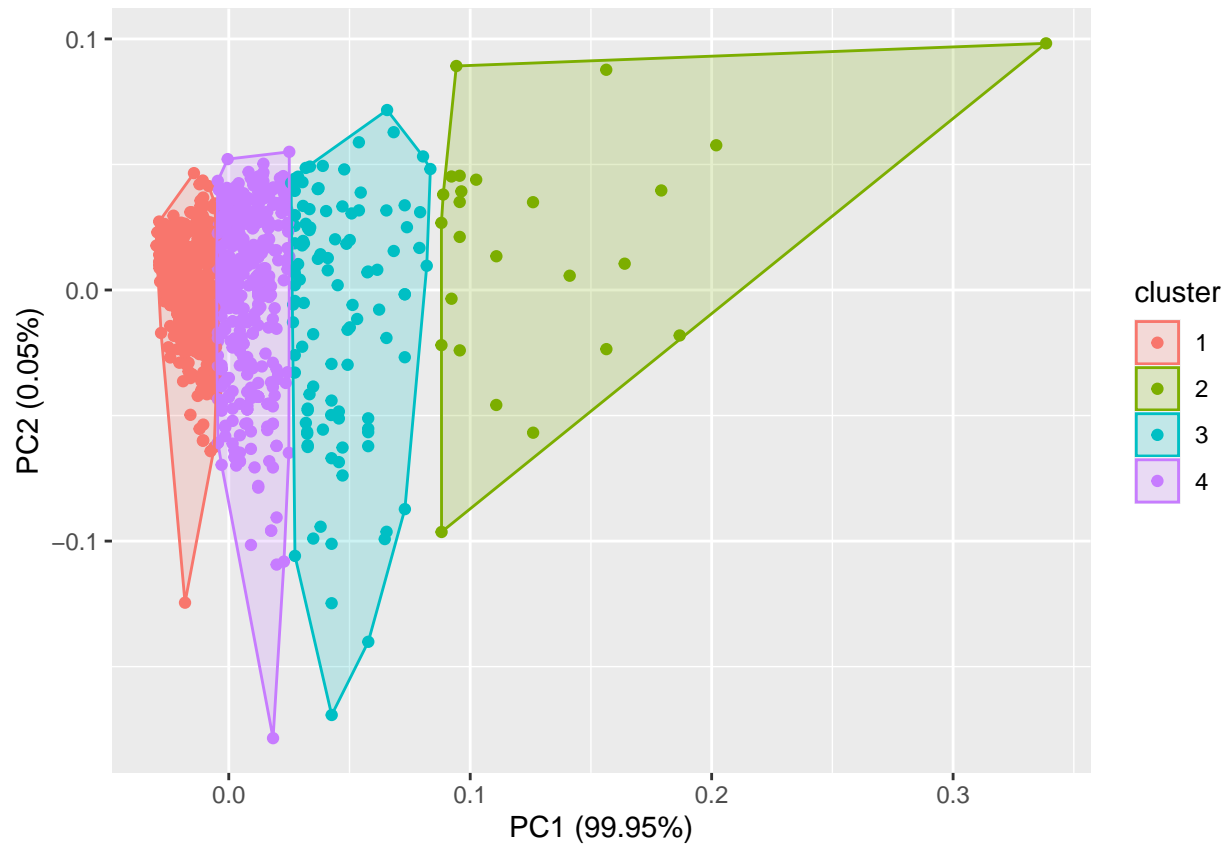
## Warning: ggrepel: 1132 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps

```

Resultados clustering K-means



```
autoplot(kmeans4, data, frame=TRUE)
```



Obsevamos que con $K=4$ tenemos todos los cluster con un gran numero de valores aunque al graficarlos vemos que están muy juntos por la zona media.