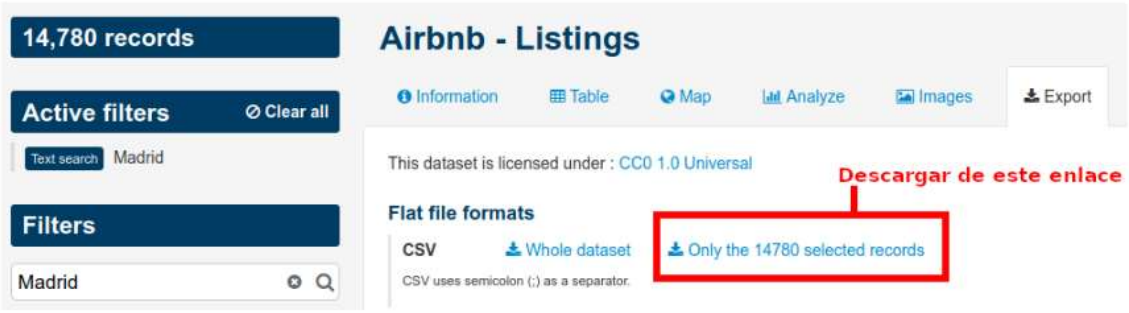
```
---
format: html
editor: visual
  markdown:
    wrap: 72
---
```

Vasmos a cargar el dataset de AirBnB descargado de aquí




```{r}
install.packages("ggplot2")
install.packages("dplyr")
```

```{r}
library(ggplot2)
library(dplyr, warn.conflicts = FALSE)
```

```{r}
airbnb<-read.csv('airbnb-listings.csv',sep = ';')
options(repr.plot.height=4,repr.plot.width=6,repr.plot.res = 300)
```

1. Vamos a quedarnos con las columnas de mayor interés:
   'City','Room.Type','Neighbourhood','Accommodates','Bathrooms','Bedrooms','Beds','Price','Square.Feet','Guests.Included',
   'Extra.People','Review.Scores.Rating','Latitude', 'Longitude'

```{r}
airbnb <- airbnb [, c('City','Room.Type','Neighbourhood','Accommodates','Bathrooms'
,'Bedrooms','Beds','Price','Square.Feet','Guests.Included','Extra.People','Review
.Scores.Rating','Latitude', 'Longitude')]

print(airbnb)
```

Description: df [14,780 × 14]

City <chr>	Room.Type <chr>	Neighbourhood <chr>	Accommod... <int>
Madrid	Entire home/apt		2
Madrid	Entire home/apt		4
Madrid	Entire home/apt		4
Madrid	Entire home/apt	Embajadores	2
Madrid	Entire home/apt	Embajadores	5
Madrid	Entire home/apt	La Latina	4
Madrid	Entire home/apt		2
Madrid	Private room	La Latina	2
Madrid	Entire home/apt	Embajadores	4
Madrid	Entire home/apt	La Latina	2

1-10 of 14,780 rows | 1-4 of 14 columns Previous 1 2 3 4 5 6 ... 100 Next

Nos quedarmos solo con las entradas de Madrid para Room.Type=="Entire home/apt" y cuyo barrio (Neighbourhood) no está vacio ''

```{r}
df_madrid <- airbnb %>% filter(City == "Madrid" & Room.Type=="Entire home/apt" & Neighbourhood != "")

print(df_madrid)
```

Description: df [5,601 × 14]

City <chr>	Room.Type <chr>	Neighbourhood <chr>	Accommod... <int>	Bathroo... <dbl>
Mad...	Entire home/apt	Embajadores	2	1.0
Mad...	Entire home/apt	Embajadores	5	1.0
Mad...	Entire home/apt	La Latina	4	1.0
Mad...	Entire home/apt	Embajadores	4	1.0
Mad...	Entire home/apt	La Latina	2	1.0
Mad...	Entire home/apt	La Latina	14	2.0
Mad...	Entire home/apt	La Latina	2	1.0
Mad...	Entire home/apt	Palacio	5	3.0
Mad...	Entire home/apt	La Latina	4	1.0
Mad...	Entire home/apt	Palacio	5	2.0

1-10 of 5,601 rows | 1-5 of 14 columns Previous 1 2 3 4 5 6 ... 100 Next

Podemos eliminar las siguientes columnas que ya no son necesarias: "Room.Type",'City' Llama a nuevo dataframe df_madrid.

```{r}
df_madrid <- df_madrid [, c('Neighbourhood','Accommodates','Bathrooms','Bedrooms'
,'Beds','Price','Square.Feet','Guests.Included','Extra.People','Review.Scores
.Rating','Latitude', 'Longitude')]

print(df_madrid)
```

Description: df [5,601 × 12]

Neighbourhood <chr>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
Embajadores	2	1.0	1	2	50
Embajadores	5	1.0	2	4	95
La Latina	4	1.0	1	2	69
Embajadores	4	1.0	1	2	57
La Latina	2	1.0	1	1	59
La Latina	14	2.0	3	14	120
La Latina	2	1.0	1	1	89
Palacio	5	3.0	3	3	192
La Latina	4	1.0	1	1	100
Palacio	5	2.0	2	2	100

1-10 of 5,601 rows | 1-6 of 12 columns Previous 1 2 3 4 5 6 ... 100 Next

2. Crea una nueva columna llamada Square.Meters a partir de Square.Feet. Recuerda que un pie cuadrado son 0.092903 metros cuadrados.

```{r}
df_madrid <- mutate(df_madrid, Square.Meters =  0.092903 * Square.Feet )

print(df_madrid)
```

Description: df [5,601 × 13]

Neighbourhood <chr>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
Embajadores	2	1.0	1	2	50
Embajadores	5	1.0	2	4	95
La Latina	4	1.0	1	2	69
Embajadores	4	1.0	1	2	57
La Latina	2	1.0	1	1	59
La Latina	14	2.0	3	14	120
La Latina	2	1.0	1	1	89
Palacio	5	3.0	3	3	192
La Latina	4	1.0	1	1	100
Palacio	5	2.0	2	2	100

1-10 of 5,601 rows | 1-6 of 13 columns Previous 1 2 3 4 5 6 ... 100 Next

3. ¿Que porcentaje de los apartamentos no muestran los metros cuadrados? Es decir, ¿cuantos tienen NA en Square.Meters?

```{r}
num_na <- sum(is.na(df_madrid$Square.Meters))
num_row <- nrow(df_madrid)
porc_NA_SQM <- round(num_na / num_row * 100,2)
cat("Numero total de NA en la columna metros cuadrados: ",num_na, "\n")
cat("Numero total de filas en el dataset df_madrid: ",num_row, "\n")
cat("Porcentaje de los apartamentos no muestran los metros cuadrados: ",porc_NA_SQM
, "%")
```

```
Numero total de NA en la columna metros cuadrados:  5254
Numero total de filas en el dataset df_madrid:  5601
Porcentaje de los apartamentos no muestran los metros cuadrados:  93.8 %
```

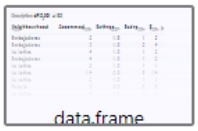
4. De todos los apartamentos que tienen un valor de metros cuadrados diferente de NA ¿Que porcentaje de los apartamentos tienen 0 metros cuadrados?

```{r}
n_zeros <- nrow(df_madrid[!is.na(df_madrid$Square.Meters) & df_madrid$Square.Meters
== 0,])

nrow_no_na <- nrow(df_madrid[!is.na(df_madrid$Square.Meters),])

porc_apt_SqrZero <- round(n_zeros / nrow_no_na * 100,2)


cat("El numero total de apartamentos con cero metros cuadrados es: ",n_zeros,"\n")
cat("El numero total de apartamentos excluyendo los de NA metros cuadrados es: "
,nrow_no_na,"\n")
cat("El porcentaje de apartamentos con cero metros cuadrados es: ",porc_apt_SqrZero
,"%")
```

```
El numero total de apartamentos con cero metros cuadrados es:  128
El numero total de apartamentos excluyendo los de NA metros cuadrados es:  347
El porcentaje de apartamentos con cero metros cuadrados es:  36.89 %
```

5. Reemplazar todos los 0m^2 por NA

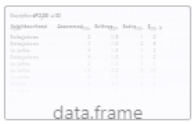
```{r}
df_madrid <- mutate(df_madrid, Square.Meters = ifelse(Square.Meters == 0,  NA, Square.Meters))
num_na_mod <- sum(is.na(df_madrid$Square.Meters))


print(df_madrid)
cat("Numero total de NA en la columna metros cuadrados modificada: ",num_na_mod, "\n")
```

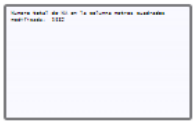


Description: df [5,601 × 13]

Neighbourhood <chr>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
Embajadores	2	1.0	1	2	50
Embajadores	5	1.0	2	4	95
La Latina	4	1.0	1	2	69
Embajadores	4	1.0	1	2	57
La Latina	2	1.0	1	1	59
La Latina	14	2.0	3	14	120
La Latina	2	1.0	1	1	89
Palacio	5	3.0	3	3	192
La Latina	4	1.0	1	1	100
Palacio	5	2.0	2	2	100

1-10 of 5,601 rows | 1-6 of 13 columns Previous 1 2 3 4 5 6 ... 100 Next





```
Numero total de NA en la columna metros cuadrados modificada:  5382
```

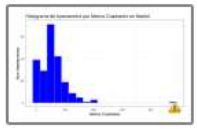
Hay muchos NAs, vamos a intentar crear un modelo que nos prediga cuantos son los metros cuadrados en función del resto de variables para tratar de rellenar esos NA. Pero **antes de crear el modelo** vamos a hacer:

* pintar el histograma de los metros cuadrados y ver si tenemos que filtrar algún elemento más.

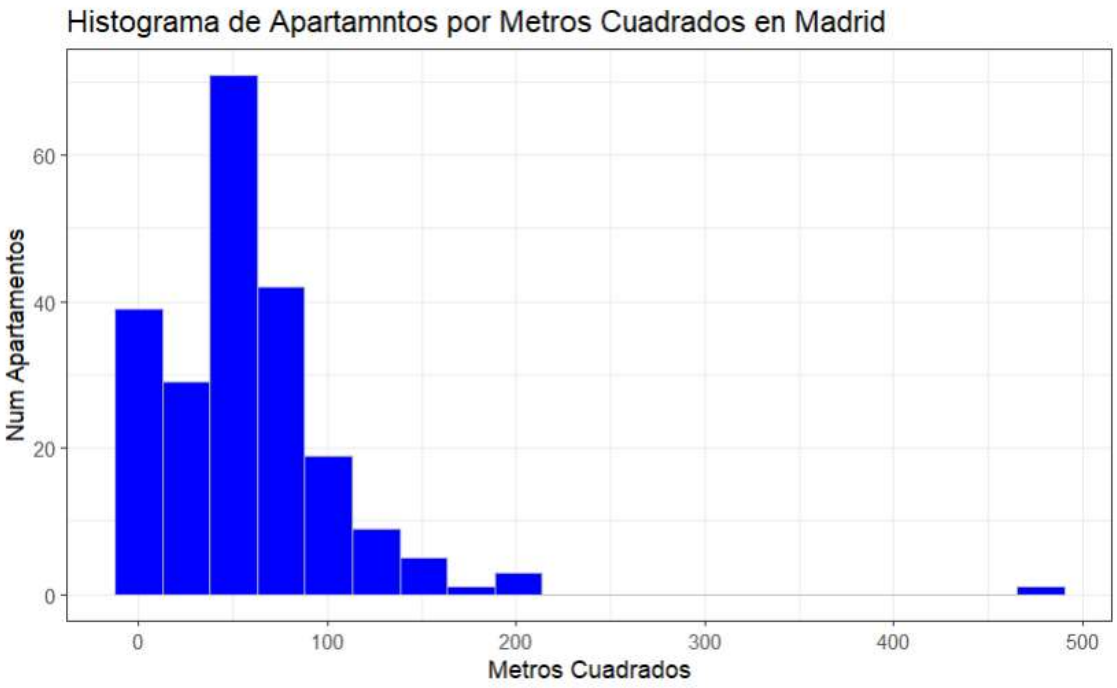
* crear una variable sintética nueva basada en la similitud entre barrios que usaremos en nuestro modelo.

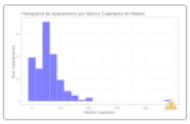
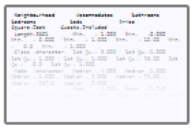
6. Pinta el histograma de los metros cuadrados y ver si tenemos que filtrar algún elemento más

```{r}
ggplot(data = df_madrid, aes(x = df_madrid$Square.Meters)) + geom_histogram(bins=20, fill='blue', color='gray')+theme_bw() + labs(x = "Metros Cuadrados", y = "Num Apartamentos") + ggtitle("Histograma de Apartamntos por Metros Cuadrados en Madrid")

summary(df_madrid)
```



```
Warning: [38;5;232mUse of `df_madrid$Square.Meters` is discouraged.
[36mi[38;5;232m Use `Square.Meters` instead.[39m
Warning: [38;5;232mRemoved 5382 rows containing non-finite values (`stat_bin()`).[39m
```

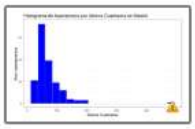


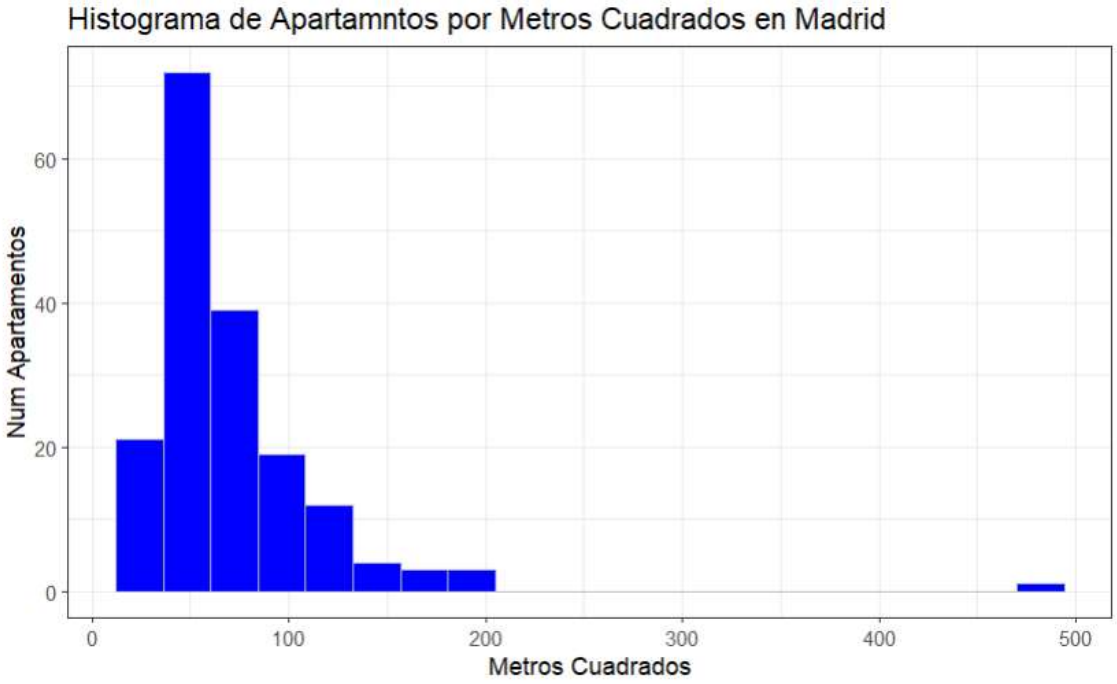
```
 Mode  :character   Median : 4.000   Median :1.000   Median : 1.000   Median :
2.000   Median : 75.00   Median : 237.0   Median : 2.000
                     Mean   : 4.101   Mean   :1.248   Mean   : 1.453   Mean   :
2.393   Mean   : 88.74   Mean   : 392.6   Mean   : 1.952
                     3rd Qu.: 5.000   3rd Qu.:1.000   3rd Qu.: 2.000   3rd Qu.:
3.000   3rd Qu.:100.00   3rd Qu.: 646.0   3rd Qu.: 2.000
                     Max.   :16.000   Max.   :6.500   Max.   :10.000   Max.
:16.000   Max.   :850.00   Max.   :5167.0   Max.   :15.000
                                      NA's   :14      NA's   :7        NA's   :8
NA's   :5        NA's   :5254
  Extra.People    Review.Scores.Rating    Latitude       Longitude
Square.Meters
 Min.   :  0.00   Min.   : 20.00       Min.   :40.33   Min.   :-3.835   Min.   :
2.044
 1st Qu.:  0.00   1st Qu.: 88.00       1st Qu.:40.41   1st Qu.:-3.707   1st Qu.:
30.008
 Median : 10.00   Median : 93.00       Median :40.42   Median :-3.702   Median :
49.982
 Mean   :  8.79   Mean   : 91.29       Mean   :40.42   Mean   :-3.699   Mean   :
57.788
 3rd Qu.: 15.00   3rd Qu.: 97.00       3rd Qu.:40.43   3rd Qu.:-3.696   3rd Qu.:
74.973
 Max.   :276.00   Max.   :100.00       Max.   :40.51   Max.   :-3.576   Max.
:480.030
                  NA's   :833                                           NA's
:5382
```

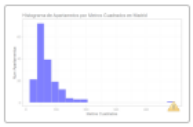
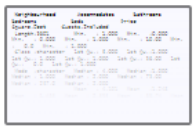
7. Asigna el valor NA a la columna Square.Meters de los apartamentos que tengan menos de 20 m^2

```{r}
df_madrid <- mutate(df_madrid, Square.Meters = ifelse(Square.Meters <= 20,  NA,
Square.Meters))


ggplot(data = df_madrid, aes(x = df_madrid$Square.Meters)) + geom_histogram(bins=20
, fill='blue', color='gray')+theme_bw() + labs(x = "Metros Cuadrados", y = "Num
Apartamentos") + ggtitle("Histograma de Apartamntos por Metros Cuadrados en Madrid"
)

summary(df_madrid)
print(df_madrid)
```

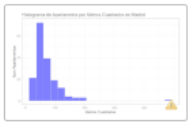


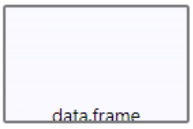

```
⚠ Warning: [38;5;232mUse of `df_madrid$Square.Meters` is discouraged.
[36mi[38;5;232m Use `Square.Meters` instead.[39m
Warning: [38;5;232mRemoved 5427 rows containing non-finite values (`stat_bin()`).[39m
```




```
 Mode  :character   Median : 4.000   Median :1.000   Median : 1.000   Median :
2.000   Median : 75.00   Median : 237.0   Median : 2.000
                     Mean   : 4.101   Mean   :1.248   Mean   : 1.453   Mean   :
2.393   Mean   : 88.74   Mean   : 392.6   Mean   : 1.952
                     3rd Qu.: 5.000   3rd Qu.:1.000   3rd Qu.: 2.000   3rd Qu.:
3.000   3rd Qu.:100.00   3rd Qu.: 646.0   3rd Qu.: 2.000
                     Max.   :16.000   Max.   :6.500   Max.   :10.000   Max.
:16.000   Max.   :850.00   Max.   :5167.0   Max.   :15.000
                                      NA's   :14      NA's   :7        NA's   :8
NA's   :5        NA's   :5254
  Extra.People    Review.Scores.Rating    Latitude       Longitude
Square.Meters
 Min.   :  0.00   Min.   : 20.00       Min.   :40.33   Min.   :-3.835   Min.   :
22.02
 1st Qu.:  0.00   1st Qu.: 88.00       1st Qu.:40.41   1st Qu.:-3.707   1st Qu.:
44.97
 Median : 10.00   Median : 93.00       Median :40.42   Median :-3.702   Median :
60.02
 Mean   :  8.79   Mean   : 91.29       Mean   :40.42   Mean   :-3.699   Mean   :
70.91
 3rd Qu.: 15.00   3rd Qu.: 97.00       3rd Qu.:40.43   3rd Qu.:-3.696   3rd Qu.:
79.99
 Max.   :276.00   Max.   :100.00       Max.   :40.51   Max.   :-3.576   Max.
:480.03
                  NA's   :833                                           NA's
:5427
```





Description: df [5,601 × 13]

Neighbourhood <chr>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
Embajadores	2	1.0	1	2	50
Embajadores	5	1.0	2	4	95
La Latina	4	1.0	1	2	69
Embajadores	4	1.0	1	2	57
La Latina	2	1.0	1	1	59
La Latina	14	2.0	3	14	120
La Latina	2	1.0	1	1	89
Palacio	5	3.0	3	3	192
La Latina	4	1.0	1	1	100
Palacio	5	2.0	2	2	100

1-10 of 5,601 rows | 1-6 of 13 columns

Previous 1 2 3 4 5 6 ... 100 Next

8. Existen varios Barrios que todas sus entradas de Square.Meters son NA, vamos a eliminar del dataset todos los pisos que pertenecen a estos barrios.

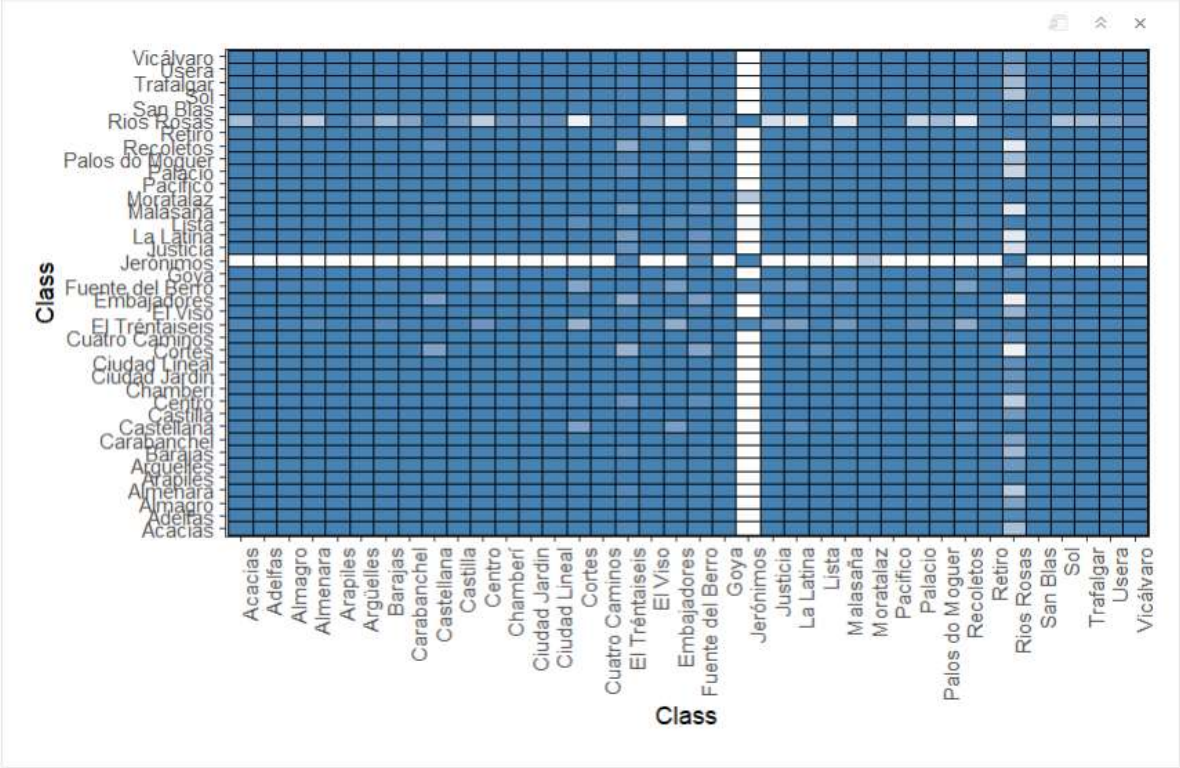
```{r}
library(dplyr)


mod_df_madrid <- df_madrid %>% group_by(Neighbourhood) %>% summarise(sum(!is.na
(Square.Meters)))

barrios <- mod_df_madrid[mod_df_madrid[2] > 0, ][1] %>% pull()
barrios <- barrios[barrios != ""]


cat("Son", length(unique(df_madrid$Neighbourhood)), "barrios con todas sus entradas
de Square.Meters diferentes de NA.\nLos barrios son:\n", paste(barrios, collapse =
", "), "\n")

df_madrid <- df_madrid[df_madrid$Neighbourhood %in% barrios, ]
```

```
Son 65 barrios con todas sus entradas de Square.Meters diferentes de NA.
Los barrios son:
 Acacias, Adelfas, Almagro, Almenara, Arapiles, Argüelles, Barajas, Carabanchel,
Castellana, Castilla, Centro, Chamberí, Ciudad Jardin, Ciudad Lineal, Cortes,
Cuatro Caminos, El Tréntaiseis, El Viso, Embajadores, Fuente del Berro, Goya,
Jerónimos, Justicia, La Latina, Lista, Malasaña, Moratalaz, Pacifico, Palacio,
Palos do Moguer, Recoletos, Retiro, Rios Rosas, San Blas, Sol, Trafalgar, Usera,
Vicálvaro
```

El barrio parece ser un indicador importante para los metros cuadrados de un apartamento.

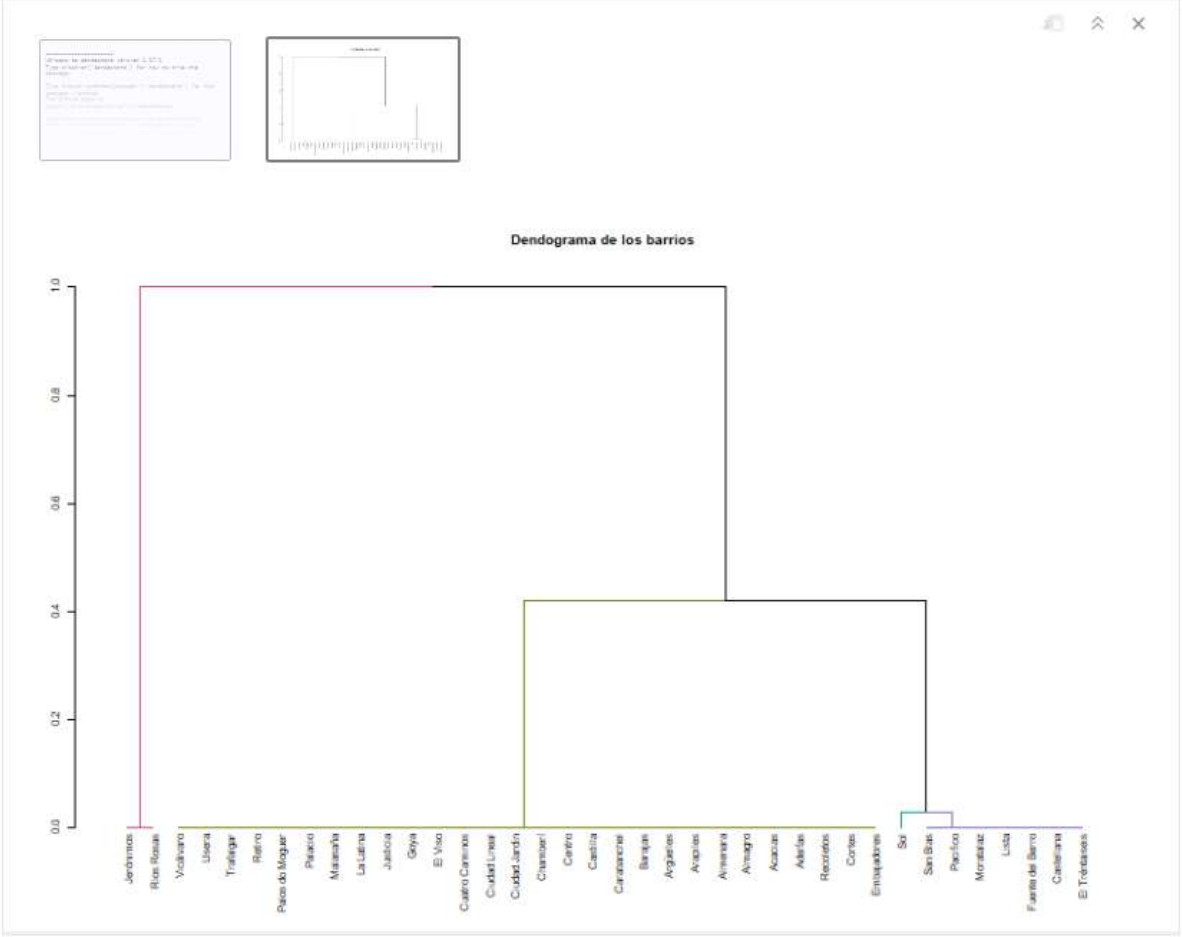
Vamos a agrupar los barrios por metros cuadrados. Podemos usar una matriz de similaridad de Tukey tal y como hicimos en el curso de estadística:

```{r}
tky<-TukeyHSD(aov( formula=Square.Meters~Neighbourhood, data=df_madrid ))
tky.result<-data.frame(tky$Neighbourhood)
cn <-sort(unique(df_madrid$Neighbourhood))
resm <- matrix(NA, length(cn),length(cn))
rownames(resm) <- cn
colnames(resm) <- cn
resm[lower.tri(resm) ] <- round(tky.result$p.adj,4)
resm[upper.tri(resm) ] <- t(resm)[upper.tri(resm)]
diag(resm) <- 1
library(ggplot2)
library(reshape2)
dfResm <- melt(resm)
ggplot(dfResm, aes(x=Var1, y=Var2, fill=value))+
  geom_tile(colour = "black")+
  scale_fill_gradient(low = "white",high = "steelblue")+
  ylab("Class")+xlab("Class")+theme_bw()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1),legend.position="none")
```



9. Usando como variable de distancia: 1-resm Dibuja un dendrograma de los diferentes barrios.

```{r}
library(dendextend)

tky.dist <- as.dist(1 - resm)
tky.tree <- hclust(tky.dist, method = "complete")
tky.dend <- as.dendrogram(tky.tree)
par(cex = 0.4)
tky.dend_colored <- color_branches(tky.dend, k = 4)
plot(tky.dend_colored, main = "Dendograma de los barrios")
```

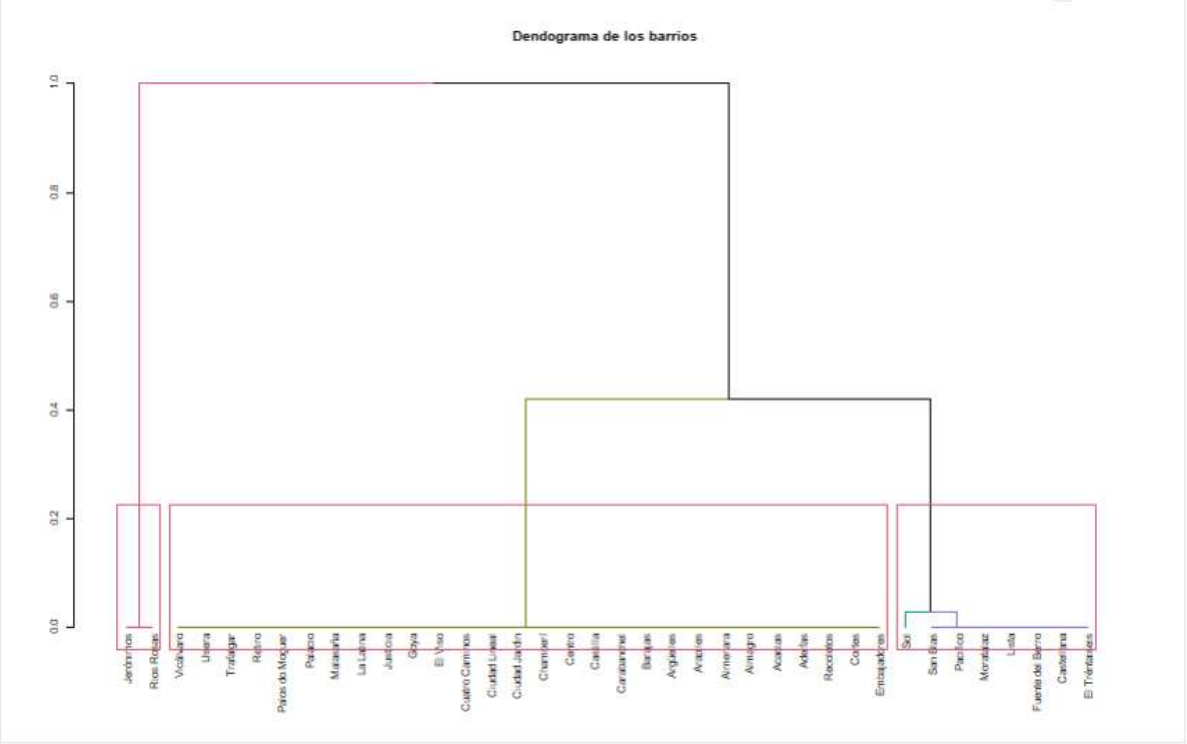



```{r}
cat("Como se puede ver ya no existen metros cuadrados inferiores a 20: \n")
summary(df_madrid[,c(1,13)])
```

```
Como se puede ver ya no existen metros cuadrados inferiores a 20:
 Neighbourhood      Square.Meters
 Length:4901        Min.   : 22.02
 Class :character   1st Qu.: 44.97
 Mode  :character   Median : 60.02
                    Mean   : 70.91
                    3rd Qu.: 79.99
                    Max.   :480.03
                    NA's   :4727
```

10. ¿Que punto de corte sería el aconsejable?

```{r}
tky.dist <- as.dist(1 - resm)
tky.tree <- hclust(tky.dist, method = "complete")
par(cex = 0.4)
tky.dend_colored <- color_branches(tky.dend, k = 4)
plot(tky.dend_colored, main = "Dendograma de los barrios")
tky.treecluster <- rect.hclust(tky.tree, h = 0.4)
```



Se sugiere establecer los cortes en 0.4, lo que resultará en la formación de 3 clusters distintos.

```{r}
tky.dend <- as.dendrogram(tky.tree)
tky.dend
```

```
'dendrogram' with 2 branches and 38 members total, at height 1
```

¿cuantos clusters aparecen?

```{r}
clusters <- cutree(tky.tree, h = 0.4)
cat("Número de clusters:", length(unique(clusters)), "\n")
table(clusters)
```

```
Número de clusters: 3
clusters
 1  2  3
28  8  2
```

11. Vamos a crear una nueva columna en el dataframe df_madrid con un nuevo identificador marcado por los clusters obtenidos. Esta columna la llamaremos neighb_id

```{r}
df_madrid["neighd_id"] <- 0
for (index in 1:3){
  df_madrid[df_madrid$Neighbourhood %in% names(clusters[clusters == index]),
"neighd_id"] <- index
}

print(df_madrid)
```

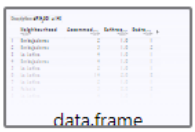
Description: df [4,901 × 14]

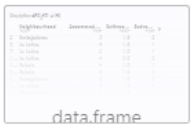
	Neighbourhood <chr>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
1	Embajadores	2	1.0	1	2	50
2	Embajadores	5	1.0	2	4	95
3	La Latina	4	1.0	1	2	69
4	Embajadores	4	1.0	1	2	57
5	La Latina	2	1.0	1	1	59
6	La Latina	14	2.0	3	14	120
7	La Latina	2	1.0	1	1	89
8	Palacio	5	3.0	3	3	192
9	La Latina	4	1.0	1	1	100
1...	Palacio	5	2.0	2	2	100

1-10 of 4,901 rows | 1-7 of 14 columns Previous 1 2 3 4 5 6 ... 100 Next

12. Vamos a crear dos grupos, uno test y otro train.

```{r}
set.seed(12345)
indices_train <- sample(1:nrow(df_madrid), 0.7 * nrow(df_madrid))

df_madrid_train <- df_madrid[indices_train, ]
df_madrid_test <- df_madrid[-indices_train, ]

print(df_madrid)
print(df_madrid_train)
print(df_madrid_test)
```





Description: df [4,901 × 14]

	Neighbourhood <chr>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
1	Embajadores	2	1.0	1	2	50
2	Embajadores	5	1.0	2	4	95
3	La Latina	4	1.0	1	2	69
4	Embajadores	4	1.0	1	2	57
5	La Latina	2	1.0	1	1	59
6	La Latina	14	2.0	3	14	120
7	La Latina	2	1.0	1	1	89
8	Palacio	5	3.0	3	3	192
9	La Latina	4	1.0	1	1	100
1...	Palacio	5	2.0	2	2	100

1-10 of 4,901 rows | 1-7 of 14 columns Previous 1 2 3 4 5 6 ... 100 Next

Description: df [3,430 × 14]

	Neighbourhood <chr>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>
51	Malasaña	2	1.0	0	1
8...	La Latina	3	1.0	1	2
8...	Palacio	3	1.0	0	3
3...	Jerónimos	4	1.0	1	2
6...	Embajadores	2	1.0	1	1
2...	Sol	3	1.0	0	2
2...	Embajadores	2	1.0	1	1
75	Sol	4	1.0	1	2
1...	Palacio	2	0.0	1	1
9...	Embajadores	6	1.0	2	3

1-10 of 3,430 rows | 1-6 of 14 columns Previous 1 2 3 4 5 6 ... 100 Next

Description: df [1,471 × 14]

	Neighbourhood <chr>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
2	Embajadores	5	1.0	2	4	95
3	La Latina	4	1.0	1	2	69
7	La Latina	2	1.0	1	1	89
1...	La Latina	4	2.0	2	3	160
1...	Palacio	4	1.0	1	2	80
1...	Palacio	3	1.0	1	1	68
1...	Embajadores	3	1.0	2	2	50
1...	La Latina	3	1.0	1	2	75
1...	Embajadores	5	1.0	1	3	75
2...	La Latina	3	1.0	1	2	60

1-10 of 1,471 rows | 1-7 of 14 columns Previous 1 2 3 4 5 6 ... 100 Next

```{r}
paste("Número de muestras de train:", nrow(df_madrid_train)," equivalente al  70%")
paste("Número de muestras de test:", nrow(df_madrid_test)," equivalente al 30%")
```

```
[1] "Número de muestras de train: 3430  equivalente al  70%"
[1] "Número de muestras de test: 1471  equivalente al 30%"
```

13. Tratamos de predecir los metros cuadrados en función del resto de columnas del dataframe.

```{r}
cat("Modelo Inicial: \n")

# Modelo inicial
model <- lm(Square.Meters ~  Accommodates + Bathrooms + Bedrooms + Beds + Price
+ Guests.Included + Extra.People + Review.Scores.Rating + Latitude + Longitude +
neighd_id, data = df_madrid_train)
summary(model)

cat("------------------------------------------------------------------------ \n")
cat("Modelo Ajustado: \n")

#Después de validar varias combinaciones, el modelo que presenta un coeficiente
de determinación (R2) más cercano a uno es el siguiente:
model1 <- lm(Square.Meters ~ Accommodates + Bathrooms + Bedrooms + Price +
neighd_id, data = df_madrid_train)
summary(model1)
```

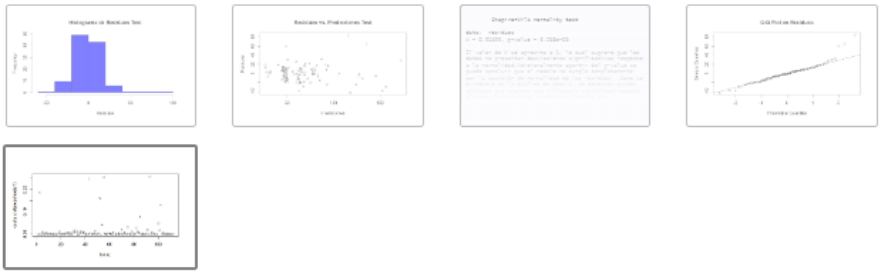
```
Modelo Inicial:

Call:
lm(formula = Square.Meters ~ Accommodates + Bathrooms + Bedrooms +
    Beds + Price + Guests.Included + Extra.People + Review.Scores.Rating +
    Latitude + Longitude + neighd_id, data = df_madrid_train)

Residuals:
    Min      1Q  Median      3Q     Max
-37.434  -9.123  -0.990   8.464  71.889

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)         -1.158e+04  6.025e+03  -1.923  0.05754 .
Accommodates        -5.212e-01  2.231e+00  -0.234  0.81578
Bathrooms            1.139e+01  5.199e+00   2.190  0.03094 *
Bedrooms             2.198e+01  3.406e+00   6.454 4.57e-09 ***
Beds                -2.844e+00  2.023e+00  -1.406  0.16308
Price                8.544e-02  3.147e-02   2.715  0.00788 **
Guests.Included     -8.037e-01  1.841e+00  -0.437  0.66342
Extra.People         4.251e-01  2.064e-01   2.060  0.04215 *
Review.Scores.Rating 2.221e-01  2.067e-01   1.074  0.28546
Latitude             2.740e+02  1.445e+02   1.896  0.06096 .
Longitude           -1.320e+02  1.099e+02  -1.201  0.23264
neighd_id            1.255e+01  4.551e+00   2.757  0.00699 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 18.63 on 95 degrees of freedom
  (3323 observations deleted due to missingness)
Multiple R-squared:  0.7238,    Adjusted R-squared:  0.6918
F-statistic: 22.63 on 11 and 95 DF,  p-value: < 2.2e-16
```

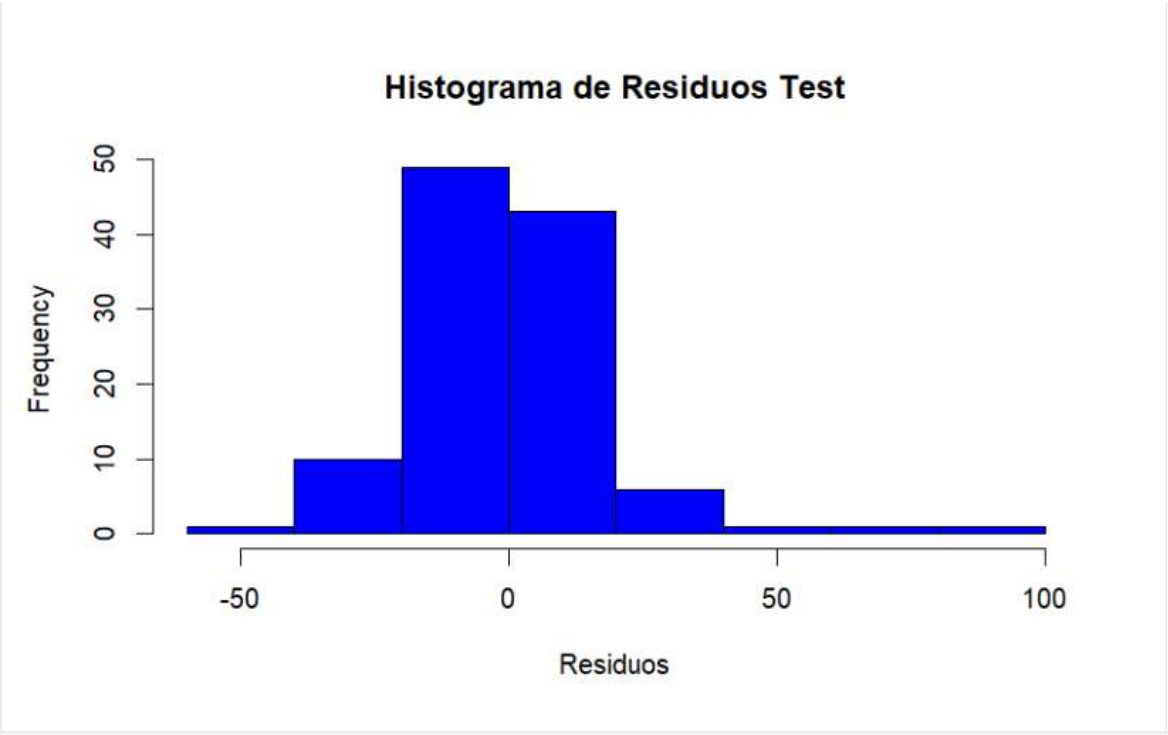
```
-----------------------------------------------------------------------
Modelo Ajustado:

Call:
lm(formula = Square.Meters ~ Accommodates + Bathrooms + Bedrooms +
    Price + neighd_id, data = df_madrid_train)

Residuals:
    Min      1Q  Median      3Q     Max
-42.838 -11.565  -1.244   8.657  84.277

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   4.94317    5.86303   0.843  0.40107
Accommodates -1.70335    1.41537  -1.203  0.23147
Bathrooms    12.31895    4.67308   2.636  0.00964 **
Bedrooms     18.98214    2.98851   6.352 5.42e-09 ***
Price         0.07017    0.02732   2.569  0.01159 *
neighd_id    12.73593    4.49325   2.834  0.00550 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 19.02 on 106 degrees of freedom
  (3318 observations deleted due to missingness)
Multiple R-squared:  0.7049,    Adjusted R-squared:  0.691
F-statistic: 50.64 on 5 and 106 DF,  p-value: < 2.2e-16
```

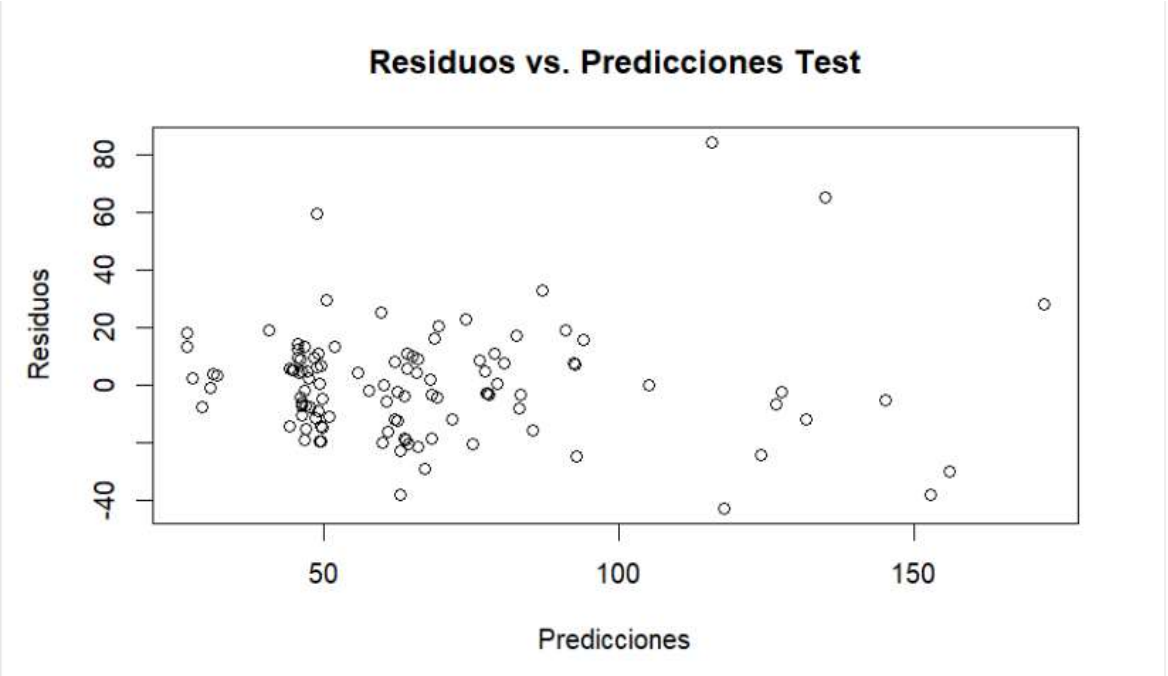
14. Mirad el histograma de los residuos sobre el conjunto de test para evaluar la calidad de vuestro modelo

```{r}
residuos <- residuals(model1)
predicciones <- predict(model1)


# Histograma de los residuos en el conjunto de test
hist(residuos, main = "Histograma de Residuos Test", col = "blue", xlab = "Residuos")

# Gráfico de dispersión de residuos vs. predicciones en el conjunto de test
plot(predicciones, residuos, main = "Residuos vs. Predicciones Test", xlab = "Predicciones", ylab = "Residuos")



# Prueba de normalidad de los residuos (puedes elegir una prueba específica según tus necesidades)
shapiro.test(residuos)
cat("El valor de W se aproxima a 1, lo cual sugiere que los datos no presentan desviaciones significativas respecto a la normalidad.Adicionalmente apartir del p-value se puede concluir que el modelo no cumple completamente con la asunción de normalidad de los residuos. .")


# Q-Q plot para evaluar la normalidad de los residuos
qqnorm(residuos, main = "Q-Q Plot de Residuos")
qqline(residuos)
```

```
# Cooks para evaluar la distribucion de los residuos
plot(cooks.distance(model1))
cat("Como se evidencia en la gráfica de Cook's, se observan puntos atípicos que ejercen una influencia significativamente elevada. Estas mismas tendencias también son perceptibles en el gráfico Q-Q line.")
```

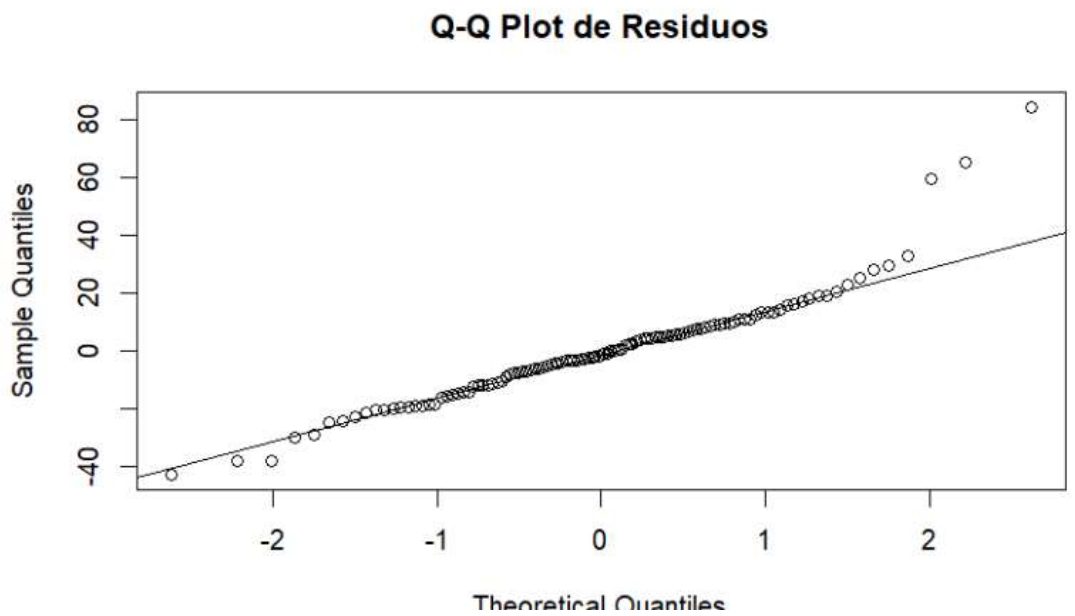


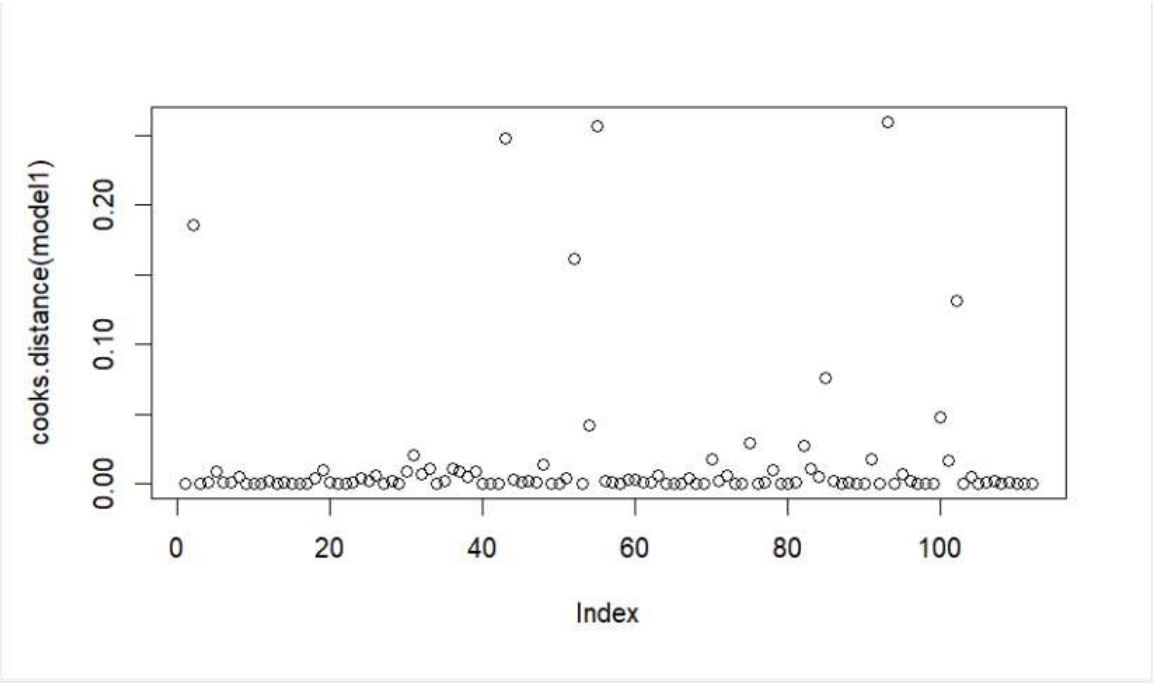

Histograma de Residuos Test
Frequency
0
10
20
30
40
50
-50
0
50
100
Residuos


Residuos vs. Predicciones Test
Residuos
-40
0
20
40
60
80
50
100
150
Predicciones

```
        Shapiro-Wilk normality test

data:  residuos
W = 0.91699, p-value = 3.268e-06
```

El valor de W se aproxima a 1, lo cual sugiere que los datos no presentan desviaciones significativas respecto a la normalidad.Adicionalmente apartir del p-value se puede concluir que el modelo no cumple completamente con la asunción de normalidad de los residuos. .Como se evidencia en la gráfica de Cook's, se observan puntos atípicos que ejercen una influencia significativamente elevada. Estas mismas tendencias también son perceptibles en el gráfico Q-Q line.





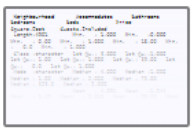
15. Si tuvieramos un anuncio de un apartamento para 6 personas (Accommodates), con 1 baño, con un precio de 80€/noche y 3 habitaciones en el barrio de Sol, con 3 camas y un review de 80. ¿Cuantos metros cuadrados tendría? Si tu modelo necesita algúna variable adicional puedes inventartela dentro del rango de valores del dataset. ¿Como varía sus metros cuadrados con cada habitación adicional?

```{r}
df_nuevo_apartamento <- data.frame(
    "Accommodates" = 6,
    "Bathrooms" = 1,
    "Price" = 80,
    "Bedrooms" = 3,
    "Neighbourhood" = "Sol",
    "Beds" = 3,
    "Review.Scores.Rating" = 80,
    "neighd_id" = as.numeric(as.factor(3)))

predict_meters_apto <- predict(model1, df_nuevo_apartamento, na.action = na.pass)
paste("Los metros cuadrados que se predicen que tendría el apartamento son:", round
(predict_meters_apto, 2))
paste("Por cada habitación adicional se predice que los metros cuadrados aumentan:"
,round(model1$coefficients["Bedrooms"],2))
```

```
[1] "Los metros cuadrados que se predicen que tendría el apartamento son: 82.34"
[1] "Por cada habitación adicional se predice que los metros cuadrados aumentan:
18.98"
```

16. Rellenar los Square.Meters con valor NA con el estimado con el modelo anterior.

```{r}
df_madrid_estimado <- df_madrid
df_madrid_estimado$Square.Meters <- predict(model1,df_madrid_estimado)
print(df_madrid_estimado)
summary(df_madrid_estimado)
```



Description: df [4,901 × 14]

	Neighbourhood <chr>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
1	Embajadores	2	1.0	1	2	50
2	Embajadores	5	1.0	2	4	95
3	La Latina	4	1.0	1	2	69
4	Embajadores	4	1.0	1	2	57
5	La Latina	2	1.0	1	1	59
6	La Latina	14	2.0	3	14	120
7	La Latina	2	1.0	1	1	89
8	Palacio	5	3.0	3	3	192
9	La Latina	4	1.0	1	1	100
1...	Palacio	5	2.0	2	2	100

1-10 of 4,901 rows | 1-7 of 14 columns Previous 1 2 3 4 5 6 ... 100 Next

```
 Neighbourhood       Accommodates      Bathrooms        Bedrooms          Beds
Price          Square.Feet     Guests.Included
 Length:4901        Min.   : 1.000   Min.   :0.000   Min.   : 0.00   Min.   :
1.000   Min.   : 18.00   Min.   :   0.0   Min.   : 1.000
 Class :character   1st Qu.: 3.000   1st Qu.:1.000   1st Qu.: 1.00   1st Qu.:
1.000   1st Qu.: 59.00   1st Qu.:   0.0   1st Qu.: 1.000
 Mode  :character   Median : 4.000   Median :1.000   Median : 1.00   Median :
2.000   Median : 75.00   Median : 323.0   Median : 2.000
                    Mean   : 4.091   Mean   :1.244   Mean   : 1.42   Mean   :
2.366   Mean   : 90.39   Mean   : 406.7   Mean   : 1.954
                    3rd Qu.: 5.000   3rd Qu.:1.000   3rd Qu.: 2.00   3rd Qu.:
3.000   3rd Qu.:100.00   3rd Qu.: 646.0   3rd Qu.: 2.000
                    Max.   :16.000   Max.   :6.000   Max.   :10.00   Max.
:16.000   Max.   :800.00   Max.   :5167.0   Max.   :15.000
                                     NA's   :14      NA's   :7       NA's   :8
NA's   :5        NA's   :4567
  Extra.People     Review.Scores.Rating    Latitude       Longitude
Square.Meters      neighd_id
 Min.   :  0.000   Min.   : 20.00       Min.   :40.36   Min.   :-3.761   Min.
: 22.54   Min.   :1.000
 1st Qu.:  0.000   1st Qu.: 88.00       1st Qu.:40.41   1st Qu.:-3.707   1st
Qu.: 47.08   1st Qu.:1.000
 Median : 10.000   Median : 93.00       Median :40.42   Median :-3.702   Median
: 52.69   Median :1.000
```

```
: 61.65   Mean   :1.184
 3rd Qu.: 15.000   3rd Qu.: 97.00       3rd Qu.:40.42   3rd Qu.:-3.697   3rd
Qu.: 75.32   3rd Qu.:1.000
 Max.   :276.000   Max.   :100.00       Max.   :40.48   Max.   :-3.576   Max.
:212.57   Max.   :3.000
                   NA's   :692                                           NA's
:26
```

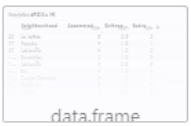
---

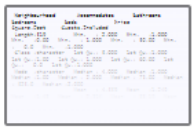
17. Usar PCA para encontrar el apartamento más cercano a uno dado. Este algoritmo nos ayudaría a dado un apartamento que el algoritmo nos devolvería los 5 apartamentos más similares.

Crearemos una función tal que le pasemos un apartamento con los siguientes datos: * Accommodates * Bathrooms * Bedrooms * Beds * Price * Guests.Included * Extra.People * Review.Scores.Rating * Latitude * Longitude * Square.Meters

y nos devuelva los 5 más similares de:

```r
{r}
#Partiendo del dataframe modificado se procede a omitir los NAs
df_madrid_estimado_outna <- na.omit(df_madrid_estimado)
print(df_madrid_estimado_outna)
summary(df_madrid_estimado_outna)
```





```
 Neighbourhood       Accommodates      Bathrooms        Bedrooms          Beds
Price          Square.Feet     Guests.Included
 Length:319         Min.   : 2.000   Min.   :1.000   Min.   :0.00   Min.   :
1.000   Min.   : 30.00   Min.   :   0.0   Min.   :1.000
 Class :character   1st Qu.: 3.000   1st Qu.:1.000   1st Qu.:1.00   1st Qu.:
1.000   1st Qu.: 60.00   1st Qu.:   0.0   1st Qu.:1.000
 Mode  :character   Median : 4.000   Median :1.000   Median :1.00   Median :
2.000   Median : 75.00   Median : 323.0   Median :2.000
                    Mean   : 4.339   Mean   :1.249   Mean   :1.48   Mean   :
2.514   Mean   : 88.19   Mean   : 397.7   Mean   :2.119
                    3rd Qu.: 5.000   3rd Qu.:1.000   3rd Qu.:2.00   3rd Qu.:
3.000   3rd Qu.: 95.00   3rd Qu.: 646.0   3rd Qu.:2.000
                    Max.   :16.000   Max.   :5.000   Max.   :5.00   Max.
:16.000   Max.   :500.00   Max.   :5167.0   Max.   :8.000
  Extra.People   Review.Scores.Rating    Latitude       Longitude
Square.Meters      neighd_id
 Min.   : 0.00   Min.   : 47.00       Min.   :40.37   Min.   :-3.732   Min.   :
26.69   Min.   :1.000
 1st Qu.: 0.00   1st Qu.: 87.00       1st Qu.:40.41   1st Qu.:-3.707   1st Qu.:
47.29   1st Qu.:1.000
 Median :10.00   Median : 92.00       Median :40.42   Median :-3.703   Median :
57.13   Median :1.000
 Mean   :10.42   Mean   : 89.96       Mean   :40.42   Mean   :-3.699   Mean   :
62.83   Mean   :1.226
```

```
75.25   3rd Qu.:1.000
 Max.    :70.00    Max.    :100.00        Max.    :40.48    Max.    :-3.576    Max.
:198.02   Max.    :3.000
```

Description: df [319 × 14]

	Neighbourhood <chr>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
22	La Latina	6	2.0	2	2	72
77	Acacias	4	1.0	1	2	60
97	Malasaña	4	1.0	2	3	105
1...	Recoletos	2	1.0	0	2	61
1...	Malasaña	8	2.0	3	4	120
1...	Sol	4	1.0	1	2	70
1...	Cuatro Caminos	4	1.0	1	3	76
1...	Argüelles	4	1.5	2	2	95
1...	Cortes	6	2.0	2	1	108
1...	Cortes	2	2.0	1	1	60

1-10 of 319 rows | 1-7 of 14 columns Previous 1 2 3 4 5 6 ... 32 Next

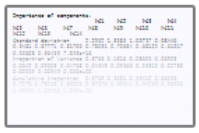
```{r}
# Convertirlos los barrios en factores numéricos.
df_madrid_estimado_outna$Neighbourhood <- as.numeric(factor
(df_madrid_estimado_outna$Neighbourhood, levels = unique
(df_madrid_estimado_outna$Neighbourhood)))

print(df_madrid_estimado_outna)
```

Description: df [319 × 14]

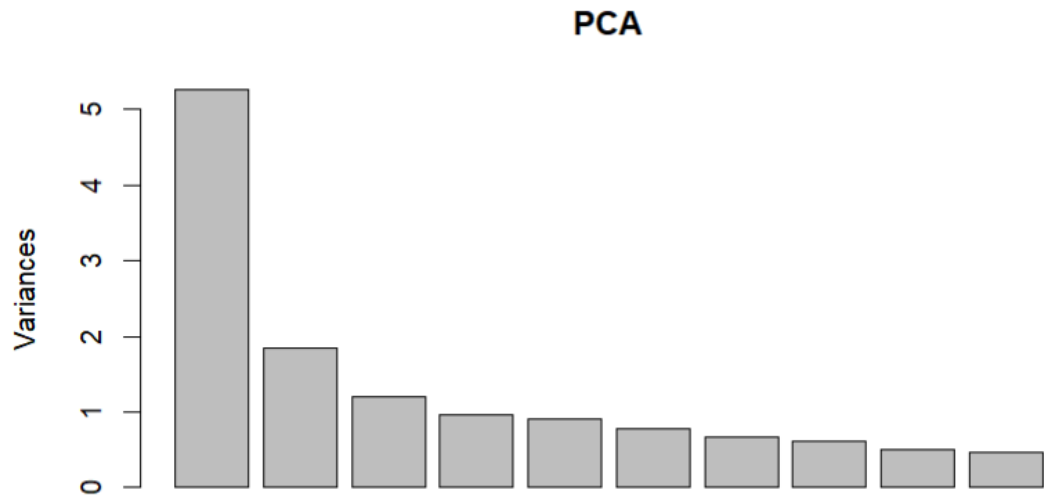
	Neighbourho... <dbl>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
22	1	6	2.0	2	2	72
77	2	4	1.0	1	2	60
97	3	4	1.0	2	3	105
1...	4	2	1.0	0	2	61
1...	3	8	2.0	3	4	120
1...	5	4	1.0	1	2	70
1...	6	4	1.0	1	3	76
1...	7	4	1.5	2	2	95
1...	8	6	2.0	2	1	108
1...	8	2	2.0	1	1	60

1-10 of 319 rows | 1-7 of 14 columns Previous 1 2 3 4 5 6 ... 32 Next

```{r}
# Calculo PCA
PCA <- prcomp(df_madrid_estimado_outna, center = TRUE, scale = TRUE)
# Resumen de los resultados
summary(PCA)
# Gráfico de la proporción de varianza explicada
plot(PCA)
```




```
Importance of components:
                          PC1    PC2     PC3     PC4    PC5     PC6     PC7
PC8     PC9    PC10    PC11    PC12    PC13      PC14
Standard deviation     2.2952 1.3583 1.09757 0.98446 0.9481 0.87771 0.81706
0.78031 0.70684 0.68125 0.61917 0.53608 0.36459 7.833e-16
Proportion of Variance 0.3763 0.1318 0.08605 0.06923 0.0642 0.05503 0.04768
0.04349 0.03569 0.03315 0.02738 0.02053 0.00949 0.000e+00
Cumulative Proportion  0.3763 0.5081 0.59412 0.66335 0.7276 0.78258 0.83026
0.87376 0.90944 0.94259 0.96998 0.99051 1.00000 1.000e+00
```




```{r}
#Elemento a analizar del dataset
df_set_apartamento <- subset(df_madrid_estimado_outna, Neighbourhood == 1 &
Accommodates == 6 & Review.Scores.Rating == 90 & Price == 150)

print(df_set_apartamento)
```

Description: df [1 × 14]

	Neighbourho... <dbl>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
5...	1	6	1	3	5	150

1 row | 1-7 of 14 columns

```r
df_set_apartamento_PCA <- predict(PCA, newdata = df_set_apartamento)

# Calcular la distancia euclidiana sin normalizar
mas_cercanos <- dist(rbind(df_set_apartamento_PCA, predict(PCA, newdata =
df_madrid_estimado_outna)))

# Asignar la distancia a la columna 'mas_cercano'
df_madrid_estimado_outna$mas_cercano <- mas_cercanos[1:nrow
(df_madrid_estimado_outna)]

# Mostrar el DataFrame actualizado
print(df_madrid_estimado_outna)
```

Description: df [319 × 15]

	Neighbourho... <dbl>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
22	1	6	2.0	2	2	72
77	2	4	1.0	1	2	60
97	3	4	1.0	2	3	105
1...	4	2	1.0	0	2	61
1...	3	8	2.0	3	4	120
1...	5	4	1.0	1	2	70
1...	6	4	1.0	1	3	76
1...	7	4	1.5	2	2	95
1...	8	6	2.0	2	1	108
1...	8	2	2.0	1	1	60

1-10 of 319 rows | 1-7 of 15 columns Previous 1 2 3 4 5 6 ... 32 Next

```r
df_madrid_estimado_outna <- df_madrid_estimado_outna[order
(df_madrid_estimado_outna$mas_cercano),]
print(df_madrid_estimado_outna)
```

Description: df [319 × 15]

	Neighbourho... <dbl>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
5...	1	6	1.0	3	5	150
5...	1	6	1.0	2	4	116
2...	3	5	1.0	3	3	120
4...	3	6	1.0	2	3	68
2...	1	5	1.0	2	3	90
1...	3	5	2.0	2	4	110
1...	1	8	1.0	3	5	80
1...	1	4	1.0	2	3	85
1...	1	4	1.0	2	4	70
2...	2	4	1.0	2	4	70

1-10 of 319 rows | 1-7 of 15 columns Previous 1 2 3 4 5 6 ... 32 Next

```r
top_5_mas_cerca <- df_madrid_estimado_outna[1:5,]
print(top_5_mas_cerca)
```

Description: df [5 × 15]

Neighbourho... <dbl>	Accommod... <int>	Bathroo... <dbl>	Bedro... <int>	B... <int>	Pri... <int>
5...	1	6	1	3	5	150
5...	1	6	1	2	4	116
2...	3	5	1	3	3	120
4...	3	6	1	2	3	68
2...	1	5	1	2	3	90

5 rows | 1-7 of 15 columns

Description: df [5 × 15]

Square.Feet <int>	Guests.Included <int>	Extra.People <int>	Review.Scores.Rating <int>
969	4	20	90
646	4	20	96
0	4	15	96
646	4	15	94
753	2	20	88

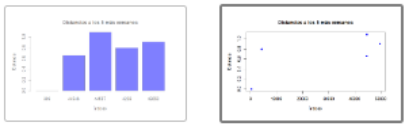
5 rows | 8-11 of 15 columns

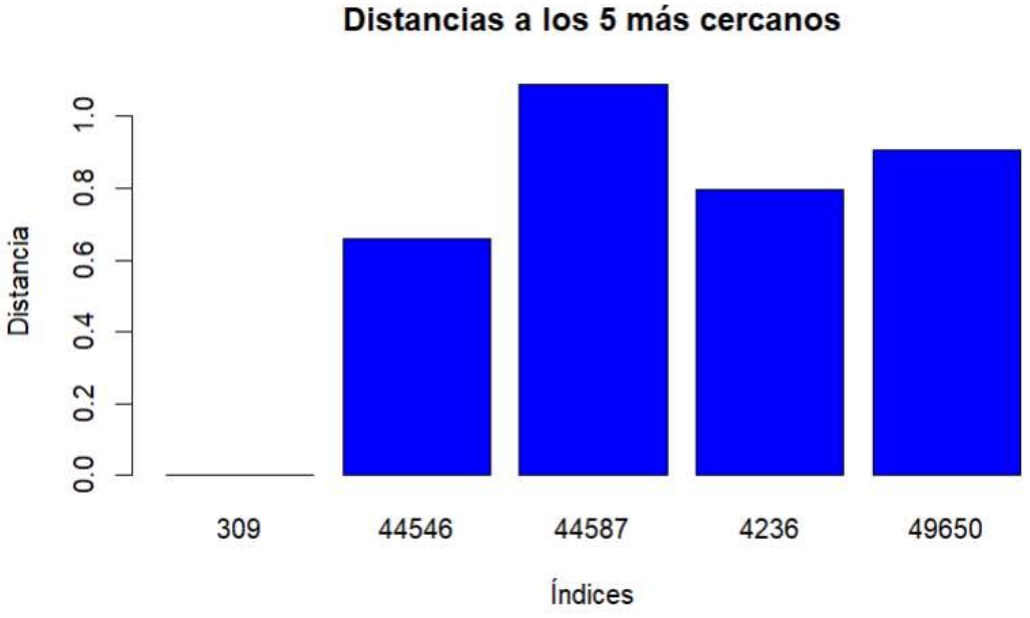
Description: df [5 × 15]

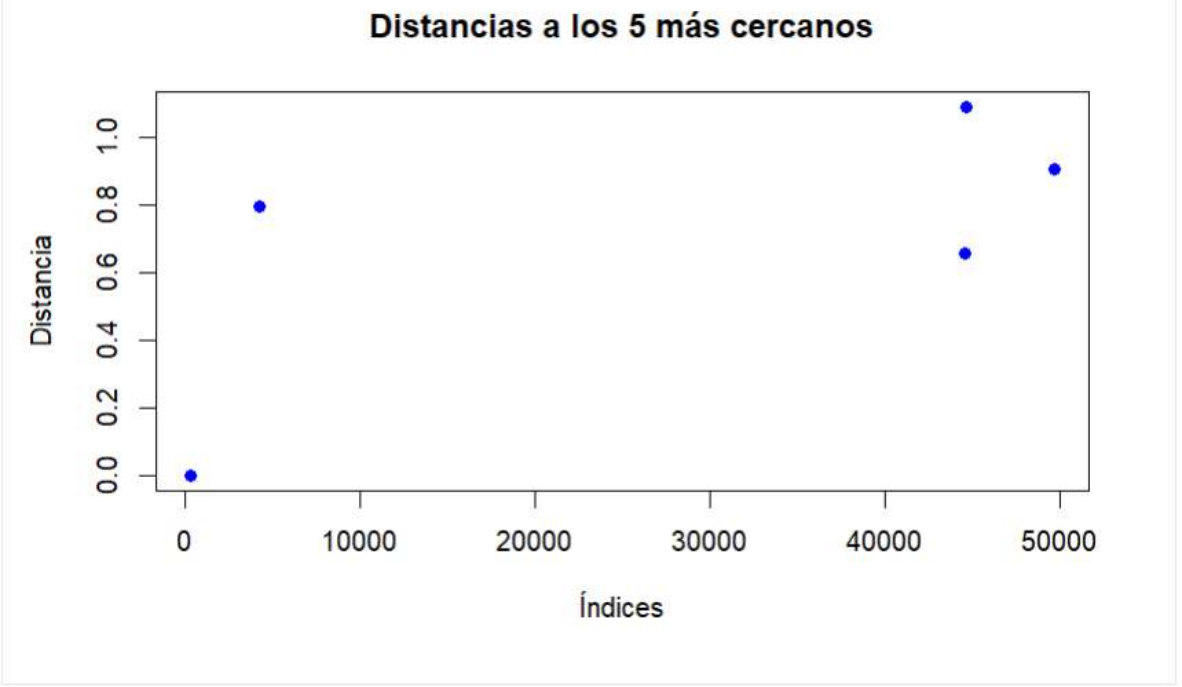
Latitude <dbl>	Longitude <dbl>	Square.Meters <dbl>	neighd_id <dbl>	mas_cerca... <dbl>
40.412...	-3.707814	87.25025	1	0.000000
40.412...	-3.713691	65.88224	1	1.901999
40.421...	-3.702710	86.84842	1	2.563910
40.423...	-3.702581	62.51395	1	2.649937
40.413...	-3.706431	65.76110	1	2.711657

5 rows | 12-16 of 15 columns

```r
# Extraer las distancias de los 5 más cercanos
distancias_5_mas_cercanos <- as.vector(mas_cercanos[indices_5_mas_cercanos])

# Gráfico de barras para visualizar las distancias
barplot(distancias_5_mas_cercanos, names.arg = indices_5_mas_cercanos,
        col = "blue", main = "Distancias a los 5 más cercanos",
        xlab = "Índices", ylab = "Distancia")


# Gráfico de dispersión para visualizar las distancias
plot(indices_5_mas_cercanos, distancias_5_mas_cercanos,
     col = "blue", pch = 16, main = "Distancias a los 5 más cercanos",
     xlab = "Índices", ylab = "Distancia")
```






Distancias a los 5 más cercanos
Distancia
0.0
0.2
0.4
0.6
0.8
1.0
0
10000
20000
30000
40000
50000
Índices