Ciencia de datos, práctica 1

Juan Casado Ballesteros, Samuel García Gonzalez, Iván Anaya Martín

October 14, 2019

Abstract

En esta práctica vamos a realizar tres análisis estadísticos, en los dos primeros utilizaremos las funciones propias de R sobre los datos proporcionados por el profesor en los formatos .txt para el primero y .sav para el segundo. En el tercer análisis hemos programado nosotros mismos nuestras propias funciones. Como datos hemos elegido un .csv que contiene información sobre el alquiler en la Ciudad de Nueva York. Intentaremos para este tercer análisis realizar un estudio crítico que nos permita llegar a conocer los datos con los que estamos trabajando.

Las dos últimas secciones del documento hacen referencia a dos listados de funciones, el primero de las funciones de R que deberíamos conocer y el segundo de las creadas por nosotros junto a su implementación.

Contents

1 Primer análisis satelites menores de urano.txt

Comenzamos leyendo los datos del archivo .txt ya que dicho archivo lo hemos escrito con la sintaxis que read.table espera no tendremos que utilizar ningún parámetro adicional para configurar la lectura de los datos.

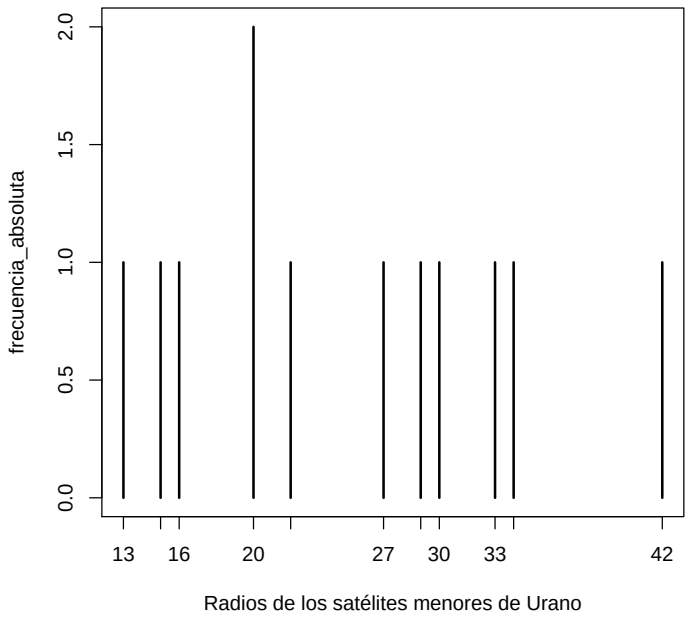
```
> satelites <- read.table("./satelites.txt")
```

1.1 Frecuencias

1.1.1 Frecuencia Absoluta

Primero calcularemos la frecuencia absoluta de los datos, que es el número de apariciones de cada uno de ellos. Vemos que todos los valores aparecen solo una vez excepto 20 que está dos veces.

```
> frecuencia_absoluta<-table(satelites$radio)
```



La moda es el dato cuya frecuencia absolutaes mayor.

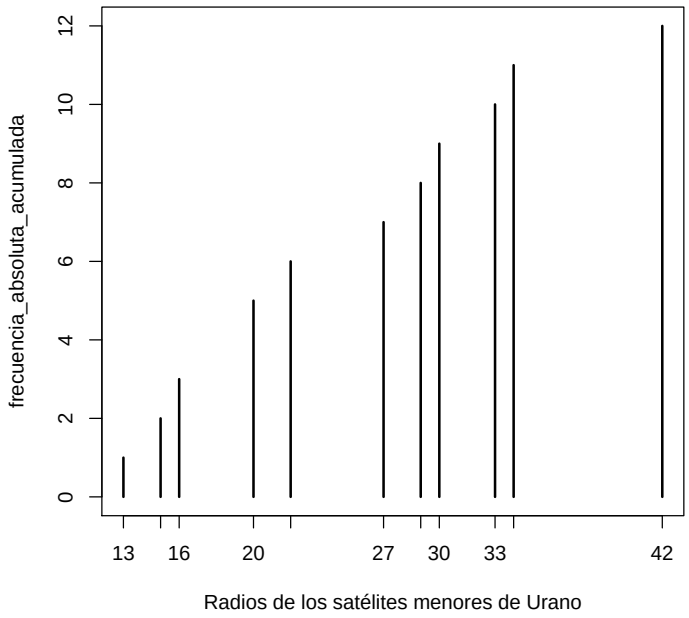
```
> datos_ordenados=sort(frecuencia_absoluta,TRUE)
> moda_ <- as.numeric(rownames(as.matrix(datos_ordenados))[1])
> moda_

[1] 20
```

1.1.2 Frecuencia Absoluta Acumulada

Ahora calcularemos la frecuencia absoluta acumulada que es la crecuencia absoluta de cada dato sumada con la de todos los menores a él. La frecuencia absoluta se va incrementadndo de forma uniforme, de uno en uno, hasta llegar a 20 que aumenta en dos tal y comno se esperaba.

Cuando utilizamos la función textbfcumsum, cuya función es acumular las frecuencias, es recomendable aplicar as.table para que los datos sigan en el formato de tabla. Esto nos permitirá luego mostrarlos en la gráfica sin problemas.

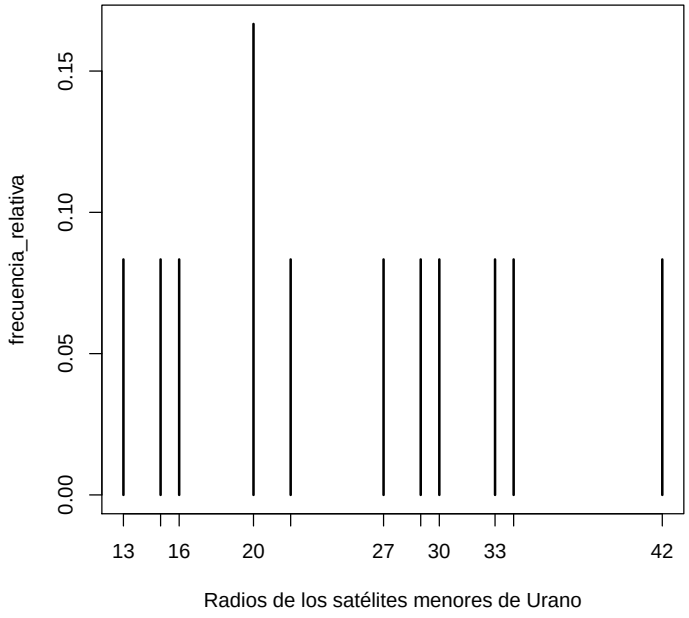
```
> frecuencia_absoluta_acumulada<-as.table(cumsum(frecuencia_absoluta))
```



1.1.3 Frecuencia Relativa

La frecuencia relativa que es la frecuencia absoluta dividida por la cantidad de datos que hay. La suma de las frecuencias relativas debe dar 1 como resultado lo cual comprobamos. La frecuncia relativa por si sola aporta más información que la frecuencia absoluta pues está normalizada entre dos valores conocidos 0 y 1. Para que la frecuencia absoluta nos aporte la misma información necesitamos conocer la cantidad de valores que tenemos para poder enmarcar el dato en su contexto.

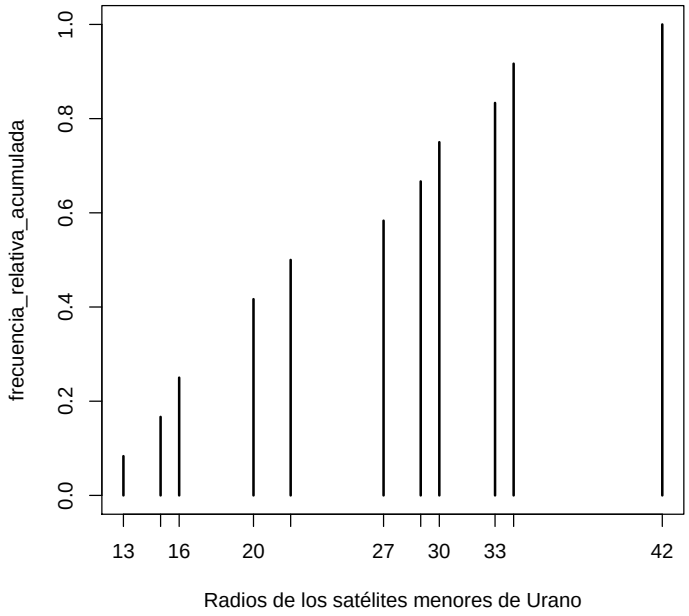
```
> frecuencia_relativa <- (function(data) table(data)/length(data))(satelites$radio)
> sum(frecuencia_relativa)

[1] 1
```



1.1.4 Frecuencia Relativa Acumulada

La frecuencia relativa acumulada que es la suma de la frecuencia relativa de cada valor con la de los menores a él.

```
> frecuencia_relativa_acumulada <- as.table(cumsum(frecuencia_relativa))
```



1.2 Valores representativos

Calcularemos a continuación estadísticos cuya función es resumir los datos de los que disponemos.

1.2.1 Media aritmética

La media aritméticaconsite en una suma de los datos ponderada por la cantidad de estos. Como podemos ver la media está ligeramente desplazada del centro del rango (29) siendo más próxima a radios de menor tamaño.

```
> media_ <- mean(satelites$radio)
> media_

[1] 25.08333
```

1.2.2 Desviación típica

La desviacion típica es pequeña lo que hace que la media sea una buen valor para representar a los datos. Sabemos que la media es buena a partir de la desviacion típica obtenida utilizando el teorema de tchebychev vemos que los valores de todos los radios se los satélites quedan dentro del intervalo $[media-2*desviacintpica, media+2*desviacintpica]$ cuando como mínimo solo tendrían que estar el 75%.

```
> desviacion_tipica_ <- sd(satelites$radio)
> desviacion_tipica_

[1] 8.857029
```

Tchebychev para k=2, al menos el 75% de los datos estarán dentro del rango $[media-2*desviacintpica, media+2*desviacintpica]$, para nuestro caso todos lo están.

```
> c(media_-2*desviacion_tipica_, media_+2*desviacion_tipica_)

[1]  7.369275 42.797392

> c(min(satelites$radio), max(satelites$radio))

[1] 13 42
```

1.2.3 Varianza

La varianza es el cuadrado de la desviación típica.

```
> varianza_ <- var(satelites$radio)
> varianza_

[1] 78.44697
```

1.3 Medidas de ordenanción

1.3.1 Cuartiles

Los cuartiles son aquellos valores de modo que si los ordenáramos sobre un vector estarían en los índices situados en el 25, 50 y 75%. Hemos podico comprobar que R para calcular los cuartiles no lo hace exáctamente así si no que utiliza una distribución de probabilidad. Cuando calculamos los cuartiles a mano con las fómulas de clase obtendremos valores distintos.

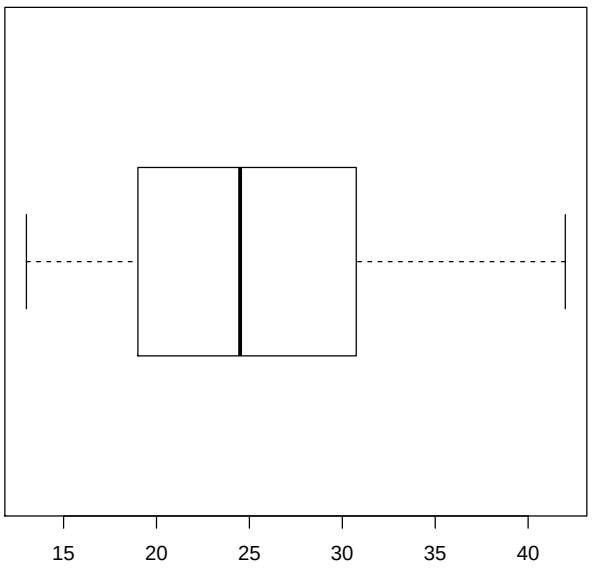
La mediana es el valor que está justo en el centro de los valores ordenados, segundo cuartil.

```
> cuartiles_ <- quantile(satelites$radio, prob=c(0, .25, .5, .75, 1))
> cuartiles_
```

```
   0%   25%   50%   75%  100%
13.00 19.00 24.50 30.75 42.00
```

```
> mediana_ <- median(satelites$radio) #cuartiles_[2]
> mediana_
```

```
[1] 24.5
```



Radios de los satélites menores de Urano

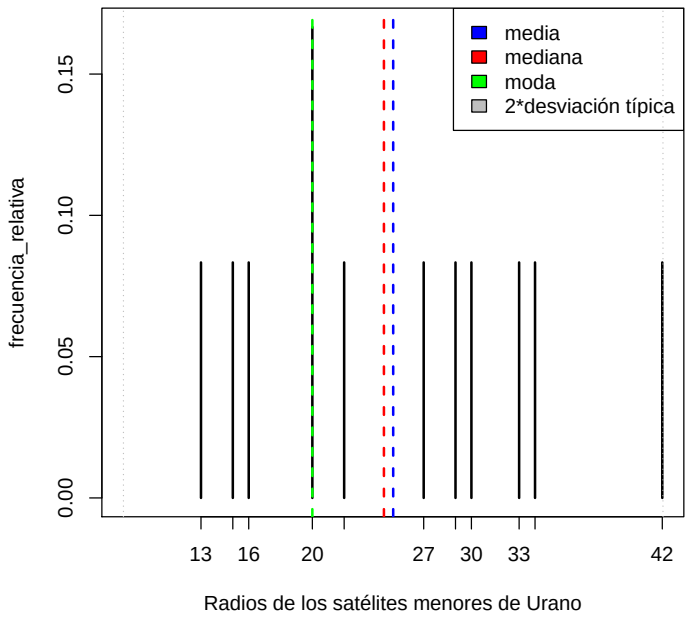
1.3.2 Cuartil 54

```
> cuantil54_ <- quantile(satelites$radio, prob=(.54))
> cuantil54_
```

```
 54%
26.7
```

1.4 Visualización

Hemos encontrado de gran utilidad representar sobre una misma gráfica la frecuencia relativa, la media, la moda y la mediana. Esta representación proporciona la mayoría de la información estadística que estamos calculando de un solo vistazo y ayuda a entender como se distrubuyen los datos que manejamos y cómo se relacionan con los valores que pretenden representarlos.



2 Segundo análisis cardata .sav

Para el segundo análisis estudiaremos los datos de la variable mpg que representa el consumo en millas por galón de un conjunto de automóbiles. Comenzamos cargado los datos que deben ser preparados. Primero eliminamos el warning de Duplicated levels in factor y posteriormente eliminamos los valores nulos para poder realizar las operaciones correctamente.

```
> cardata <- read.spss("./cardata.sav", use.value.labels = FALSE)$mpg
> cardata <- cardata[!is.na(cardata)]
```

2.1 Valores representativos

2.1.1 Media aritmética

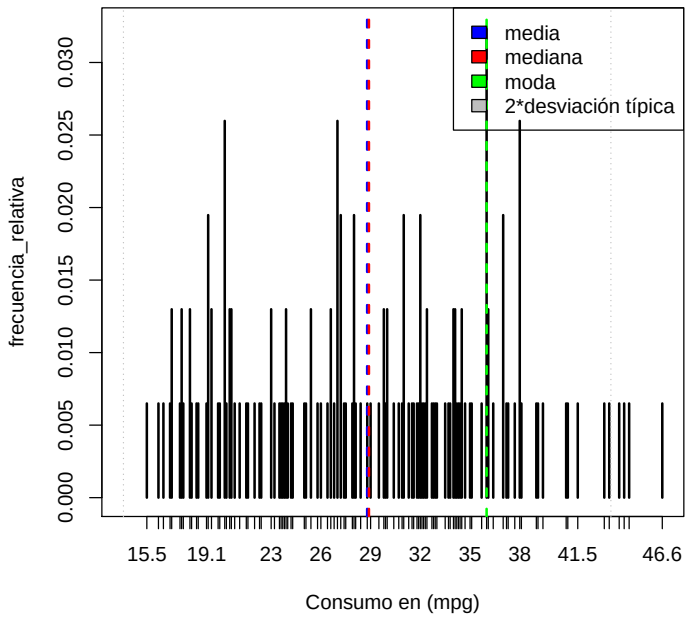
La media está muy próxima a la mediana (28.9) que a su ver está muy próxima al punto medio del rango de valores (31). Podemos ver que está ligeramente desplazada hacia los valores menores sobre todo teniendo en cuenta que la moda es bastante elevada aunque esta muy poco representativa pues en general los valores se distribuyen de forma uniforme sobre el rango.

```
> media_ <- mean(cardata)
> media_

[1] 28.79351
```

2.1.2 Desviación típica y varianza

La desviación típica es baja, el radio de 2desviación típica del teorema de tchebychev cubre a gran parte de los valores, desde luego a más del 75%. Podemos decir que la media representa bien a los valores a pesar de que los que están en el extremo superior del rango queden un poco alejados de ella.

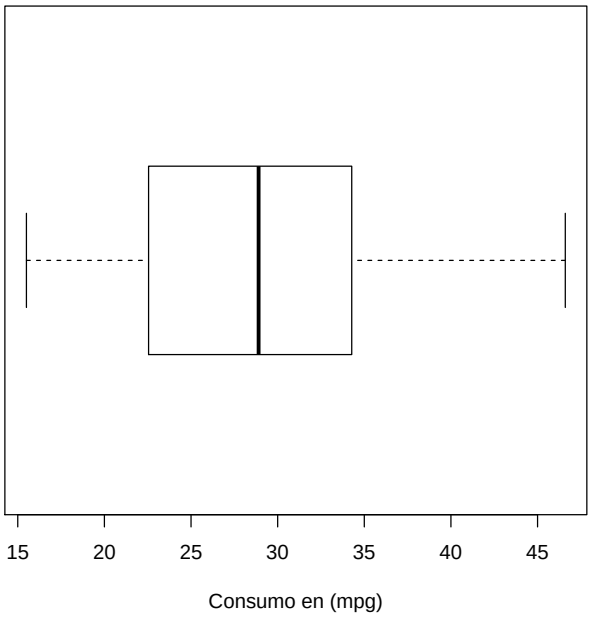
```
> desviacion_tipica_ <- sd(cardata)
> desviacion_tipica_

[1] 7.37721

> varianza_ <- var(cardata)
> varianza_

[1] 54.42323
```


media
mediana
moda
2*desviación típica
frecuencia_relativa
0.000
0.005
0.010
0.015
0.020
0.025
0.030
15.5
19.1
23
26
29
32
35
38
41.5
46.6
Consumo en (mpg)

2.2 Medidas de ordenanción

Podemos ver como los datos están repartidos de una forma bastante homogénea a lo largo de su rango de valores.

```
> cuartiles_ <- quantile(cardata, prob=c(0, .25, .5, .75, 1))
> cuartiles_

    0%    25%    50%    75%   100%
15.500 22.550 28.900 34.275 46.600

> mediana_ <- median(cardata) #cuartiles_[2]
> mediana_

[1] 28.9
```



3 Tercer análisis del alquiler en Nueva York con AirBNB durante 2019 .csv

Haremos ahora un análisis de los datos del alquiler en la ciudad de Nueva York durante el año 2019 con la compañía AirBNB. Nuestro interés y por lo que hemos elegido estos datos es porque deseamos conocer cuales son los precios del alquiler en esa ciudad. Comenzamos por cargar los datos desde un archivo en formato csv.

```
> #https://www.kaggle.com/dgomonov/new-york-city-airbnb-open-data#AB_NYC_2019.csv
> data <- read.csv("AB_NYC_2019.csv")
```

Ya que estos son unos dato por ahora desconocidos para nosotros comenzamos por visualizar las varibales que contienen y su tipo. Visualizamos también la cantidad de datos sobre los que haremos el análisis.

```
> getInfo(data)
```

```
                                unlist.res.
id                                  integer
name                                 factor
host_id                             integer
host_name                            factor
neighbourhood_group                  factor
neighbourhood                        factor
latitude                            numeric
longitude                           numeric
room_type                            factor
price                               integer
minimum_nights                      integer
number_of_reviews                   integer
last_review                          factor
reviews_per_month                   numeric
calculated_host_listings_count      integer
availability_365                    integer
res_frame
 factor integer numeric
      6       7       3
```

```
> length(data$price)
```

```
[1] 48895
```

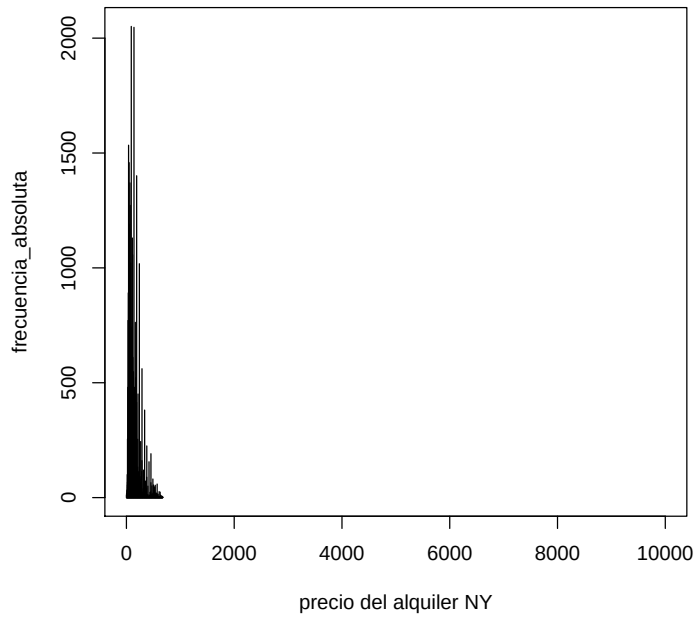
3.1 La ciudad al completo

Primero calcularemos el rango de los valores de los precios para saber cual es el másximo y el mínimo valor que tenemos. Observamos que el rango de valores sobre el que se distribuyen los precios es considerablemente elevado, sobre todo teniendo en cuenta que el precio hace referencia al precio por noche.

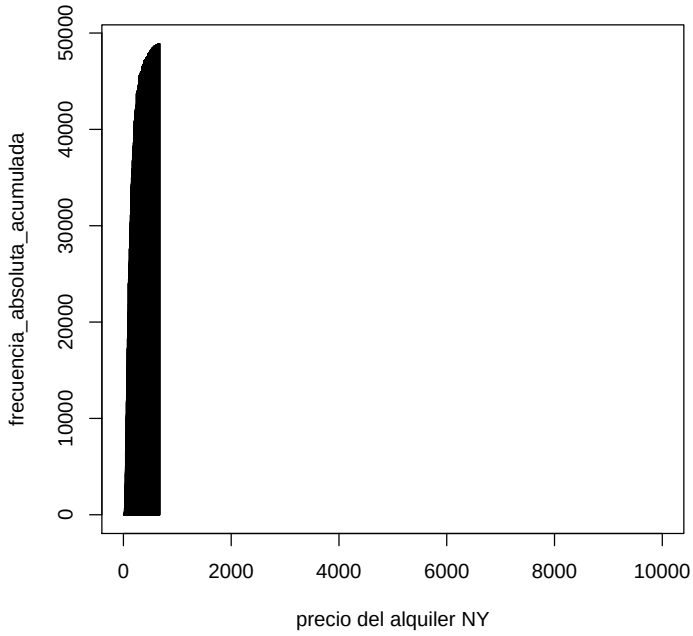
```
> rango_<-rango(data$price)
> rango_

[1]     0 10000
```

Primero calcularemos las frecuencias de los precios en la ciudad. Podemos observar que las mayores frecuencias se concentran en la parte baja de la gráfica, es decir, la mayoría de precios son bajos. No obstante hay unos pocos alquileres que tienen valores muy elevados. Vemos como la moda está muy alejada hacia la derecha de el centro del rango.

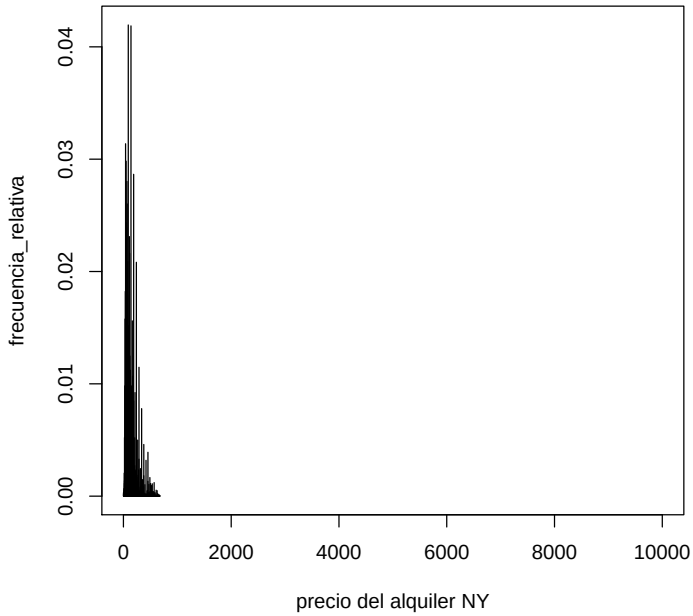
```
> frecuencia_absoluta<-frecuenciaAbsoluta(data$price)
> moda_ <- moda(data$price)
> moda_

[1] 100
```



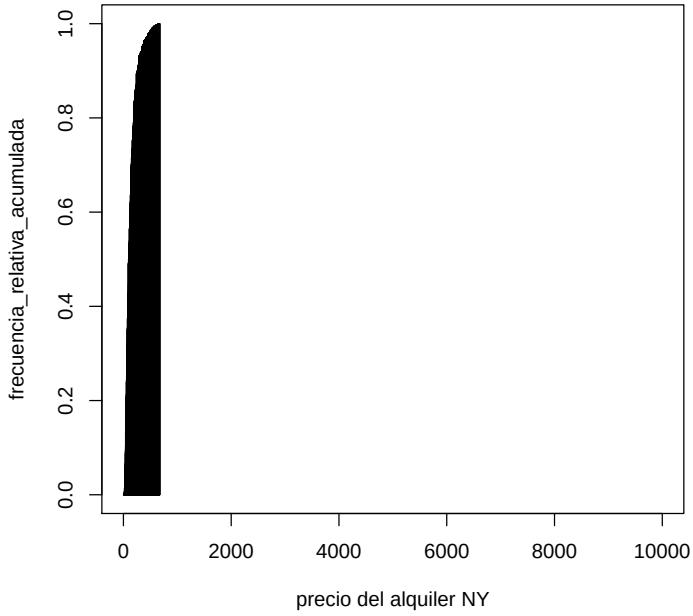

```
> frecuencia_absoluta_acumulada<-frecuenciaAbsolutaAcumulada(data$price)
```




```
> frecuencia_relativa<-frecuenciaRelativa(data$price)
```

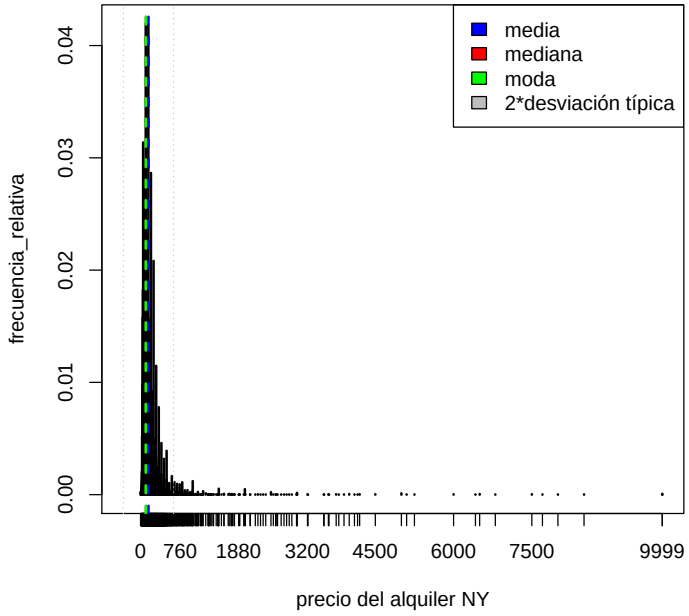



```
> frecuencia_relativa_acumulada<-frecuenciaRelativaAcumulada(data$price)
```



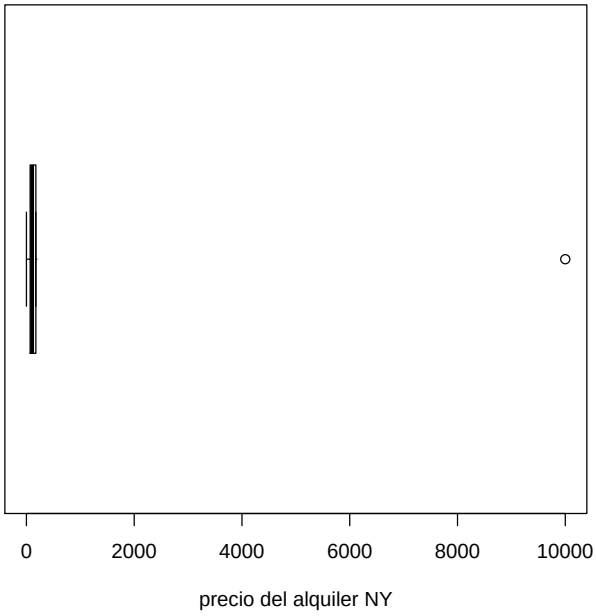
Calculamos ahora la miadia con la desviación típica y la varianza. La media está muy alejada del centro del rango y la desviación típica es muy elevada. Podemos ver como el lado izquierdo del intervalo de tchebychev para k=2 está en su mayor parte vacío y que una gran cantidad de valores quedan fuera de él por la derecha. Concluímos que la media no representa de forma correcta a los datos.

```
> mediaAritmetica_ <- mediaAritmetica(data$price)
> mediaAritmetica_

[1] 152.7207

> desviacion_tipica_ <- desviacionTipica(data$price)
> desviacion_tipica_

[1] 240.1517

> varianza_ <- varianza(data$price)
> varianza_

[1] 57672.85
```



Por último calculamos y representamos los cuartiles observando que tal y como habíamos podido ver anteriormente las frecuencias se agolpan en los valores bajos y se distribuyen unifórmemente pero con muy baja densisdad hasta alcanzar valores altos.

```
> cuartiles_ <- cuartiles(data$price)
> cuartiles_

[1]     0    69   106   175 10000

> mediana_ <- mediana(data$price)
> mediana_

[1] 106
```

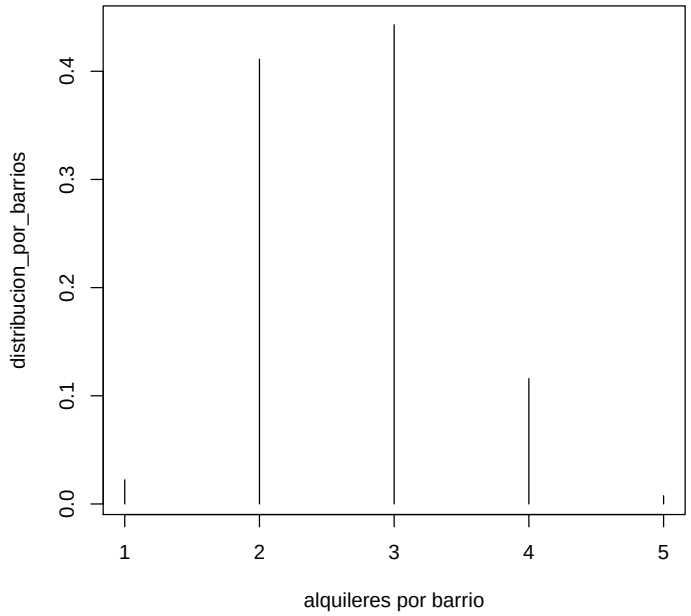

0
2000
4000
6000
8000
10000
precio del alquiler NY

3.2 La ciudad por barrios

Primero calcularemos la distribución de los apartementos en cada barrio y posteriormente la distribución de los precios en esos barrios. Podemos observar que la mayoría de apartementos se encuentran en Manhattan y Brooklyn.

```
> distribucion_por_barrios = frecuenciaRelativa(data$neighbourhood_group)
> distribucion_por_barrios
```

```
       Bronx     Brooklyn    Manhattan       Queens Staten Island
 0.022313120  0.411166786  0.443010533  0.115880969  0.007628592
```



Vemos que ninguno de los valores que podrían resumir a los datos media, moda o mediana lo hacen realmente. Los valores bajos de los datos se ven bien representados pero los muy elevados que son cerca del 25% no. Decidimos ahora calcular el precio medio por cada barrio en vez de hacerlo sobre toda la ciudad a la vez.

Calculamos los valores estadísticos de resumen para cada barrio por separado y los representamos luego superpuestos sobre la gráfica de su frecuencia relativa. Representamos también los cuatiles en las gráficas de bigores para observar la densidad de la distribución de los datos.

```
        Group.1   x.media x.desviacion_tipica x.mediana    x.moda
1         Bronx  87.49679           106.66043  65.00000  60.00000
2      Brooklyn 124.38321           186.86889  90.00000 100.00000
3     Manhattan 196.87581           291.37646 150.00000 150.00000
4        Queens  99.51765           167.08741  75.00000  50.00000
5 Staten Island 114.81233           277.24801  75.00000  75.00000
```

Valores sobre el total de los datos sin haber separado por barrios

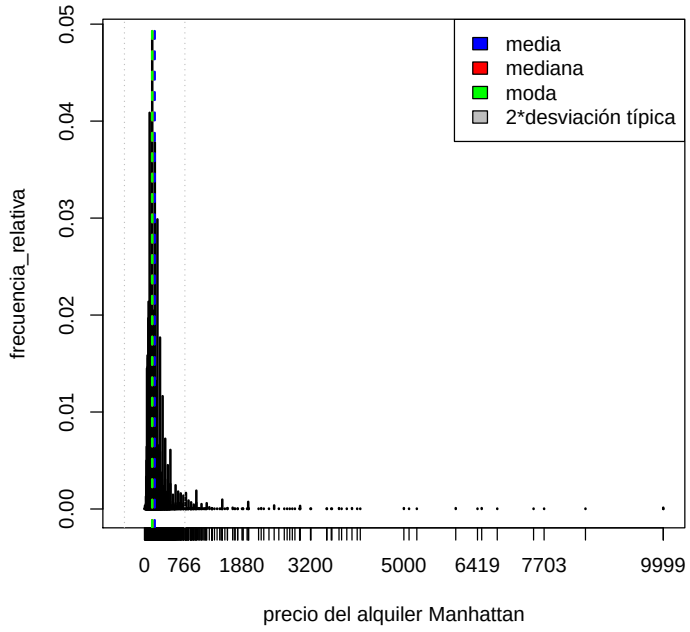
```
          media desviacion_tipica           mediana             moda
       152.7207          240.1517          106.0000          100.0000
```

Podemos ver que la varianza se ha reducido para más barrios de en los que ha aumentado. En Bronx es en el barrio en el que tanto la varianza como la media es menor y Manhattan es en el que ambas son mayores.

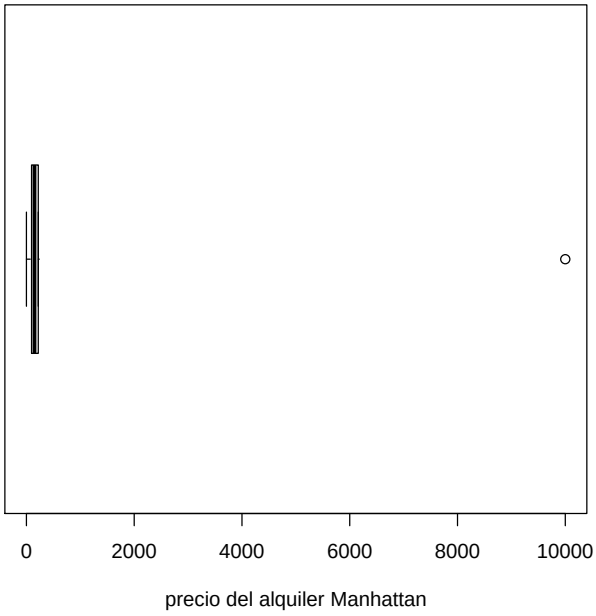
En todos los barrios se observan resultados similares a los observados para toda la ciudad, hay muchos valores bajos pero también una cantidad constante y unifórmemente distribuída que cubre cerca del 25% de los datos de valores mucho más elevados.

Datos para Manhattan

```
> manhattan_data = data$price[data$neighbourhood_group == "Manhattan"]
```

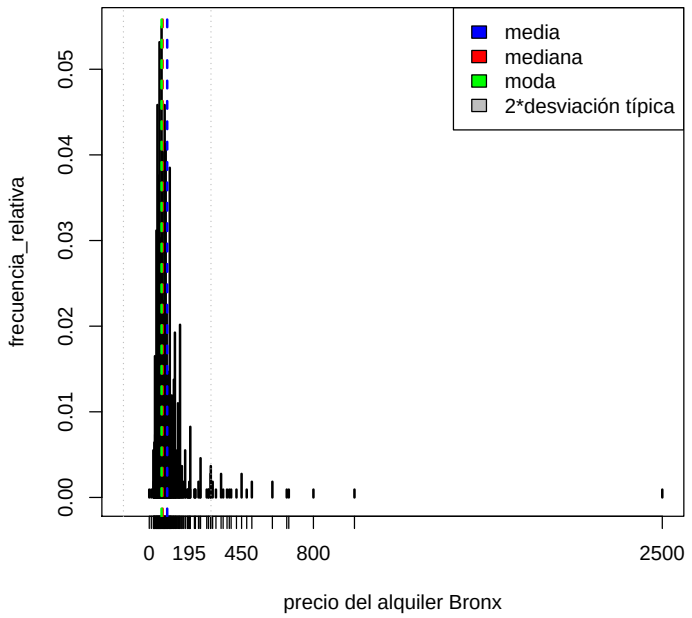



```
> cuartiles(manhattan_data)

[1]     0    95   150   220 10000
```

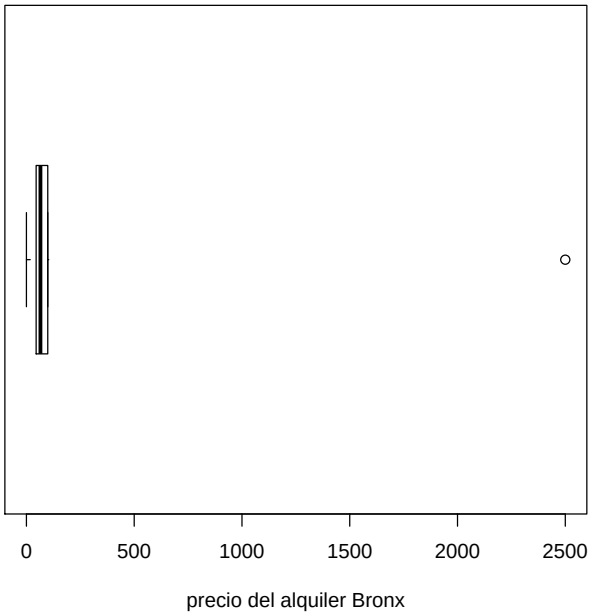


Datos para Bronx

```
> bronx_data <- data$price[data$neighbourhood_group == "Bronx"]
```

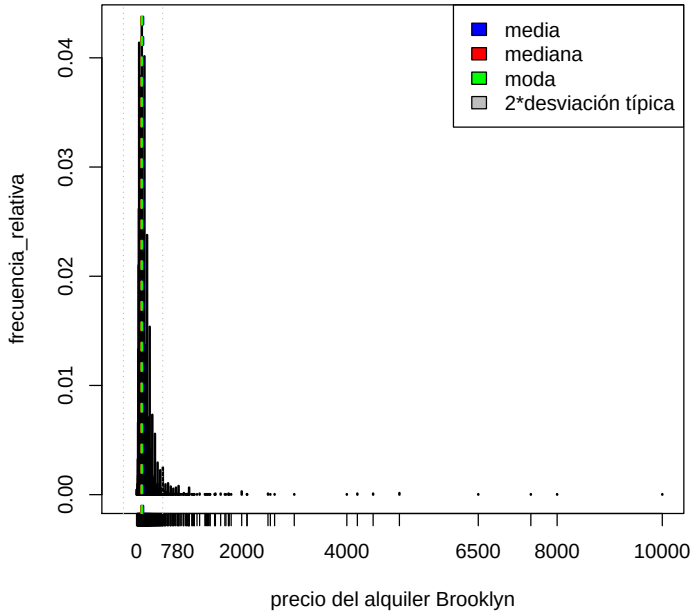


```
> cuartiles(bronx_data)
[1]    0   45   65   99 2500
```

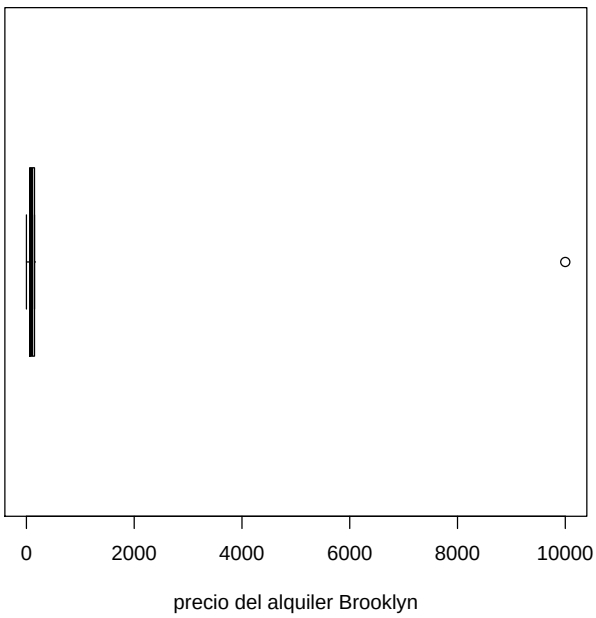


Datos para Brooklyn

```
> brooklyn_data <- data$price[data$neighbourhood_group == "Brooklyn"]
```

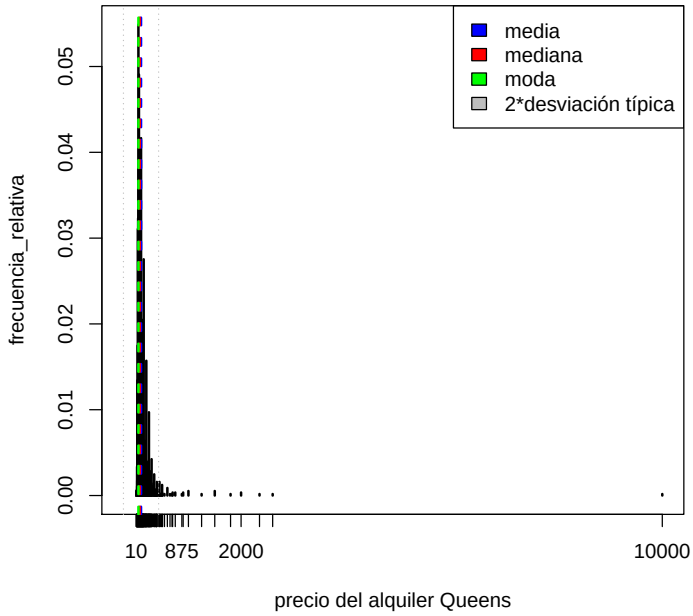



```
> cuartiles(brooklyn_data)
[1]     0    60    90   150 10000
```

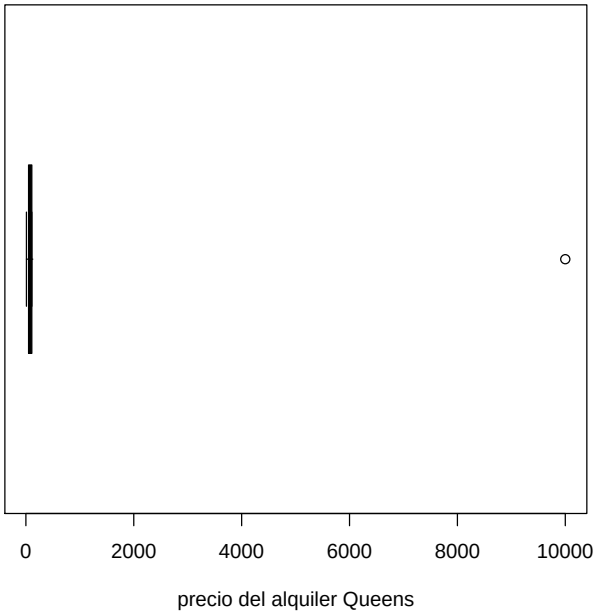


Datos para Queens

```
> queens_data <- data$price[data$neighbourhood_group == "Queens"]
```

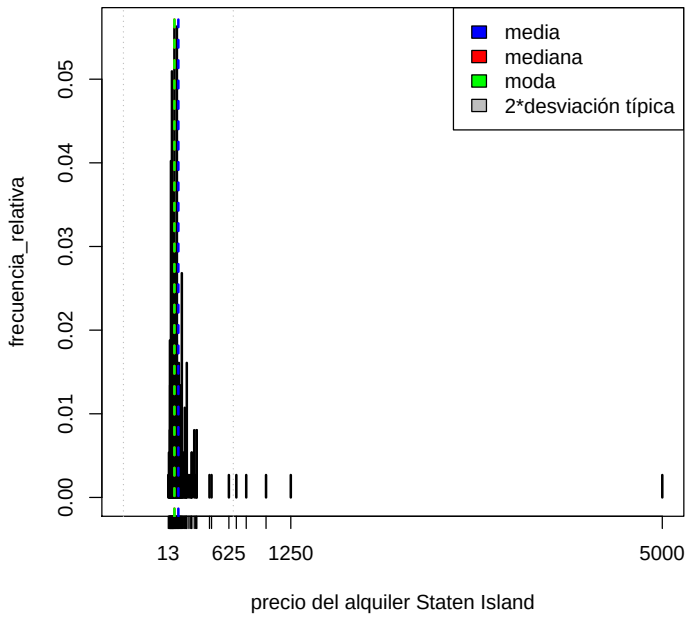



```
> cuartiles(queens_data)

[1]    10    50    75   110 10000
```

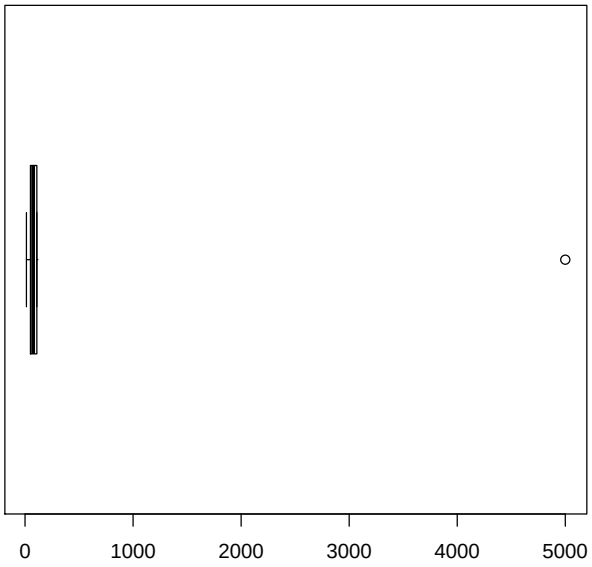


Datos para Staten Island

```
> state_data <- data$price[data$neighbourhood_group == "Staten Island"]
```



```
> cuartiles(state_data)
[1]   13.0   50.0   75.0  109.5 5000.0
```



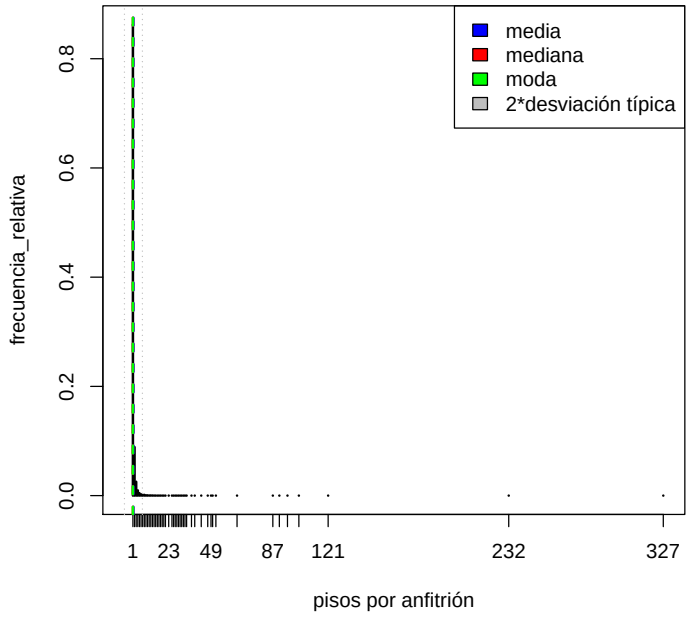
3.3 Apartamentos por anfitrión

Ya que la información que hemos obtenido hasta ahora de los datos ha resultado ser bastante pobre decidimos ahora tomar una aproximación diferente. Evaluaremos por tanto cuantos apartamentos tiene cada anfitrión en alquiler.

Podemos ver que sin ninguna duda el valor significativo en este caso es 1; el 86/

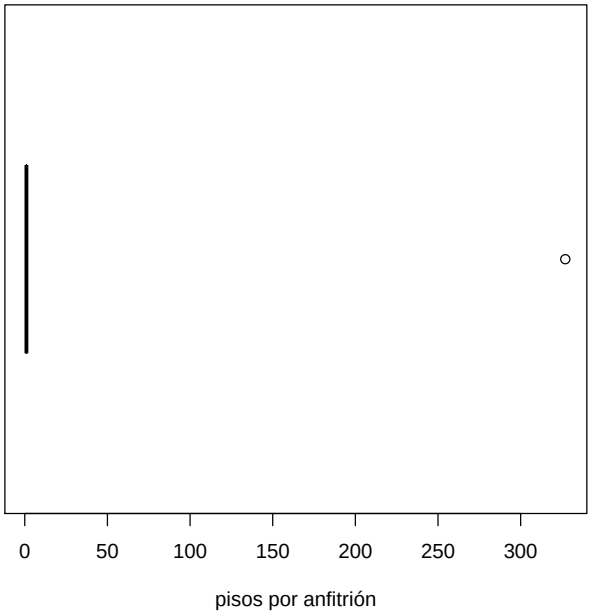
```
> host_ammount <- frecuenciaAbsoluta(data$host_id)
```

```
            media desviacion_tipica  mediana.37565381              moda
         1.305363          2.760710          1.000000          1.000000
```




```
> cuartiles(host_ammount)
```

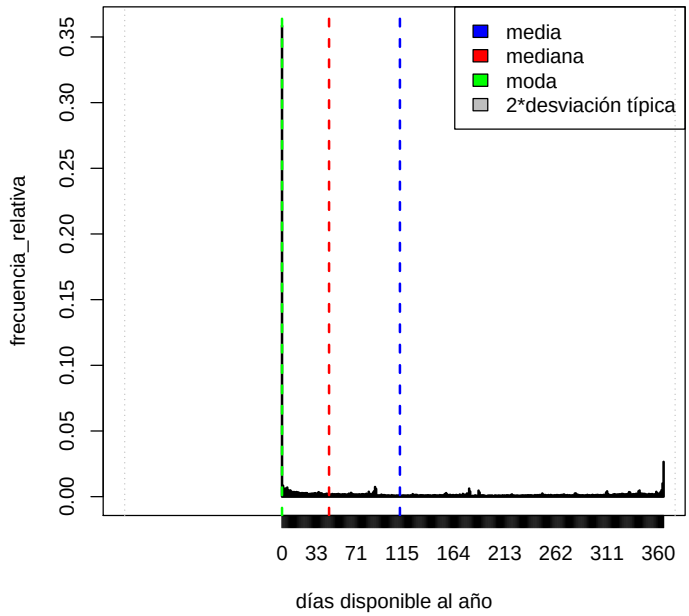
```
     2438   9049657  37565381 160057164 219517861
        1         1         1         1       327
```


0
50
100
150
200
250
300
pisos por anfitrión

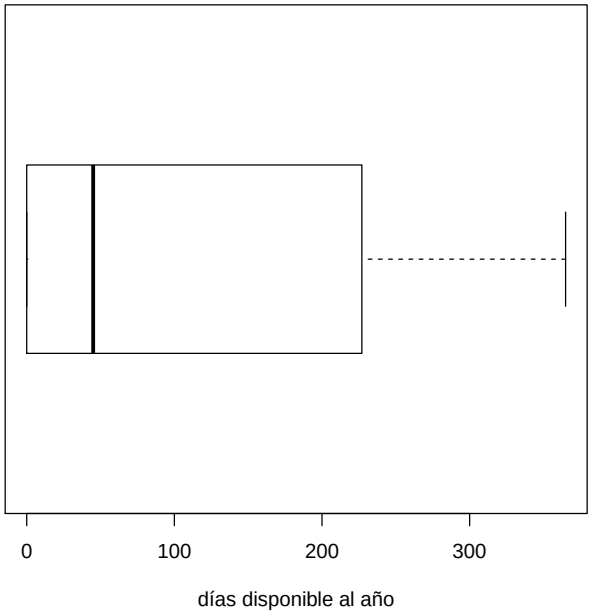
3.4 Disponibilidad

Evaluamos ahora la disponiblidad de los apartamentos, es decir, veremos cuantos días los apartementos en alquiler están libres. Podemos ver que la disponibilidad en la ciudad es baja y que el 50/Entorno al 35/

```
> media_ <- mediaAritmetica(data$availability_365)
> media_

[1] 112.7813

> desviacion_tipica_ <- desviacionTipica(data$availability_365)
> desviacion_tipica_

[1] 131.6209

> varianza_ <- varianza(data$availability_365)
> varianza_

[1] 17324.07

> moda_ <- moda(data$availability_365)
> moda_

[1] 0
```




```
> cuartiles(data$availability_365)

[1]   0   0  45 227 365
```

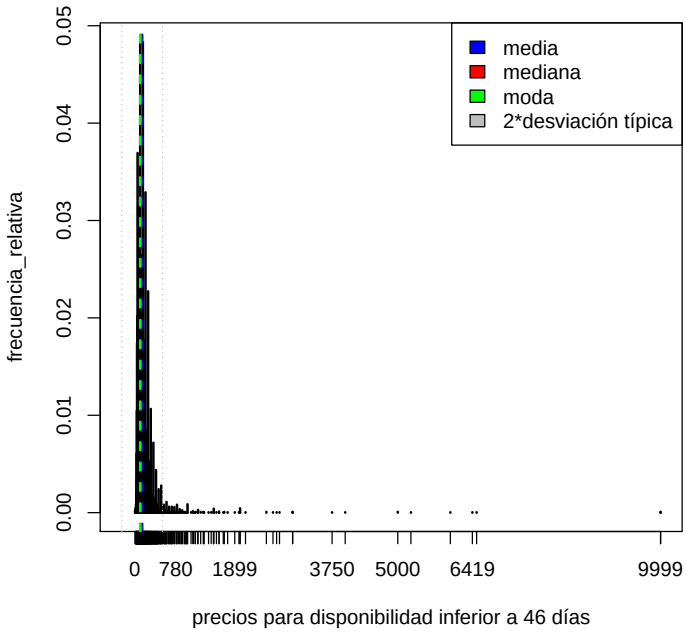


También podemos ver estos mismos datos pero sobre cada barrios, observado que la disponibilidad no afecta a todos los barrios por igual. En el Bronx, Queens y Staten Island los alquileres con menos de 46 días disponibles estám muy lejos del 50/

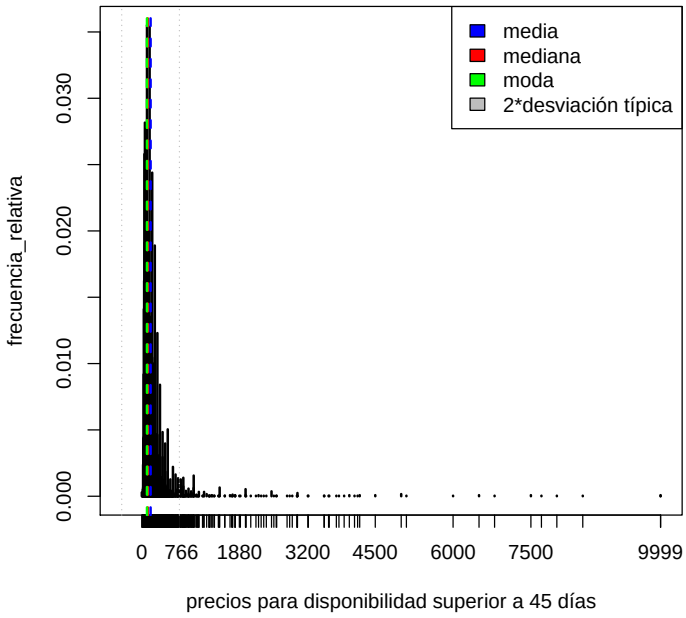
```
> distribucion_por_barrios <- frecuenciaAbsoluta(data$neighbourhood_group)
> disponibilidad_por_barrios <- c()
> poca_disponibilidad <- table(data$neighbourhood_group[data$availability_365 <= 45])
> for (barrio in 1:length(distribucion_por_barrios)){
+   disponibilidad_por_barrios <- c(disponibilidad_por_barrios,
+     poca_disponibilidad[barrio]/distribucion_por_barrios[barrio])
+ }
> disponibilidad_por_barrios

       Bronx      Brooklyn     Manhattan        Queens Staten Island
   0.2676444     0.5413351     0.5179355     0.3595129     0.1930295
```

Adicionalmente parece ser que el precio no es razón para que un apartamento tenga más o menos disponibilidad pues se observan resultado similares en ambos casos.

```
> plotFrecuencyData(data$price[data$availability_365 <= 45], xlabel="precios para disponib
```




```
> plotFrecuencyData(data$price[data$availability_365 > 45], xlabel="precios para disponibi
```



4 Guia en R para el análisis estadístico

4.1 Frecuencias

```
> Frecuencia_absoluta <- table(data_)
> Frecuencia_absoluta_acumulada <- cumsum(frecuencia_absoluta)
> Frecuencia_relativa <- (function(data) table(data)/length(data))(data_)
> Frecuencia_relativa_acumulada <- cumsum(frecuencia_relativa)
```

4.2 Medidas representativas

```
> Media <- mean(data_)
> Desviacion_tipica <- sd(data_)
> Varianza <- var(data_)
```

4.3 Medidas de ordenación

```
> Mediana <- median(data_)
> Cuartiles <- quantile(data_, prob=c(0, .25, .5, .75, 1))
> Cuantil54 <- quantile(data_, prob=(.54))
```

5 Funciones creadas

5.1 Frecuencias

5.1.1 Frecuencia Absoluta

Obtenemos una lista ordenada de los valores únicos que componen nuestros datos. La ferecuencia absoluta de cada valor único es la cantidad de veces que aparece.

```
> frecuenciaAbsoluta
```

```
function(data){
  uniquedata<-unique(data)
  uniquedata<- sort(uniquedata)
  newdata<- vector(mode="numeric", length=0)
  for (value in uniquedata) {
    ammount<-length(data[data==value])
    newdata<-c(newdata, ammount)
  }
  setNames(newdata, uniquedata)
}
<bytecode: 0x7f91947bb528>
```

5.1.2 Frecuencia Absoluta Acumulada

Obtenemos una lista ordenada de los valores únicos que componen nuestros datos. La ferecuencia absoluta acumulada de cada valor único es la cantidad de veces que aparece él sumada a la cantidad de veces que aparecen los valores menores a él.

```
> frecuenciaAbsolutaAcumulada
```

```
function(data){
    frecuencia <- frecuenciaAbsoluta(data)
    uniquedata<-unique(data)
    uniquedata<- sort(uniquedata)
    newdata <- vector(mode="numeric", length=0)
    acc <- 0
    for (value in 1:length(frecuencia)) {
        acc <- frecuencia[value]+acc
        newdata <- c(newdata, acc)
    }
    setNames(newdata, uniquedata)
}
<bytecode: 0x7f91939088d8>
```

5.1.3 Frecuencia Relativa

Obtenemos una lista ordenada de los valores únicos que componen nuestros datos. La ferecuencia relativa de cada valor es la cantidad de veces que aparece dividida por la cantidad de valores que hay. Si sumamos todas las frecuencias relativas debemos obtener 1 como resultado.

```
> frecuenciaRelativa

function(data){
  frecuencia <-frecuenciaAbsoluta(data)
  uniquedata<-unique(data)
  uniquedata<- sort(uniquedata)
  newdata <- vector(mode="numeric", length=0)
  for (value in 1:length(uniquedata)) {
    newdata <- c(newdata, frecuencia[value]/length(data))
  }
  setNames(newdata, uniquedata)
}
<bytecode: 0x7f91948832b8>
```

5.1.4 Frecuencia Relativa Acumulada

Obtenemos una lista ordenada de los valores únicos que componen nuestros datos. La ferecuencia relativa acumulada de cada valor es la frecuencia absoluta de ese valor dividida entre la cantidad de valores que hay.

```
> frecuenciaRelativaAcumulada

function(data){
  frecuencia <- frecuenciaAbsolutaAcumulada(data)
  uniquedata<-unique(data)
  uniquedata<- sort(uniquedata)
  newdata<- vector(mode="numeric", length=0)
  for (index in 1:length(uniquedata)) {
    newdata <- c(newdata, frecuencia[index]/length(data))
  }
  setNames(newdata, uniquedata)
}
<bytecode: 0x7f919393ac78>
```

5.1.5 Moda

Nos indica que dato tiene la mayor frecuencia absoluta.

```
> moda

function(data) {
  frecuencia_absoluta=frecuenciaAbsoluta(data)
  datos_ordenados=sort(frecuencia_absoluta,TRUE)
  as.numeric(rownames(as.matrix(datos_ordenados))[1])
}
<bytecode: 0x7f91946949f8>
```

5.2 Medidas representativas

Los siguentes valores pretenden ser representates de todos los datos que se estén analizando. No tenemos garantís de que realmente lo sean así que para ellos necesitamos de erramientas que nos lo confiermen o desmientan.

5.2.1 Media Aritmetica

Suma ponderada por la cantidad de valores sumados. Se podría haber utilizado la función sum que suma todos los valores de un vector en vez de haber iterado sobre ellos.

```
> mediaAritmetica
```

```
function(data){
  acc <- 0
  for (value in data) {
    acc <- acc + value
  }
  acc / length(data)
}
<bytecode: 0x7f9195986f90>
```

5.2.2 Media Geométrica

Raiz de orden igual a la cantidad de valores de la multiplicación de todos ellos. El logaritmo de la media geométrica también puede expresarse como la media aritmética de los logaritmos de cada dato.

```
> mediaGeometrica
```

```
function(data){
  prod(data)^(1/length(data))
}
```

5.2.3 Media Armónica

Inverso de la media de los inversos.

```
> mediaArmonica
```

```
function(data){
  1/mediaAritmetica(1/data)
}
```

5.2.4 Desviacion Típica

Nos indica como de buena es la media aritmética. Cuanto menor sea la desviación mejor será la media. Raiz cuadrada de la varianza.

```
> desviacionTipica
```

```
function (data) {
  varianza(data)^(1/2)
}
<bytecode: 0x7f91938fc098>
```

5.2.5 Desviacion Media

Nos indica como de buena es la media aritmética. Cuanto menor sea la desviación mejor será la media. Se suele utilizar la desviacion típica en vez de la desviacion media. Suma ponderada por la cantidad de valores del valor absoluto de los errores. Se define error como la resta entre un valor y la media aritmética de todos ellos.

```
> desviacionMedia
```

```
function (data){
  v_media <- media(data)
  acc = 0
  for (value in data){
    acc <- acc + abs(value - v_media)
  }
  acc/length(data)
}
```

5.2.6 Varianza

Nos indica como de buena es la media aritmética. Cuanto menor sea la varianza mejor será la media. Suma ponderada por la cantidad de valores del cuadrado de los errores. Se define error como la resta entre un valor y la media aritmética de todos ellos.

```
> varianza
```

```
function(data){
  v_media <- mediaAritmetica(data)
  acc = 0
  for (value in data){
    acc <- acc + (value - v_media)^2
  }
  acc/length(data)
}
<bytecode: 0x7f9195cc6498>
```

5.2.7 Tchebychev

Decir que la desviación típica es mejor cuanto menor sea no es demasiado precios. El teorema de tchebychev nos proporciona una representación muy visual del significado de la media. Se puede entender la media como un valor entorno al que se ubican los datos y la desviacion típica como una circunferencia con centro en la media y radio el valor de la desviación multiplicado por un factor K. Tchebychev nos proporciona el porcentaje mínimo de valores que estarán dentro de cada valor de radio. Esto nos ayuda a comparar la desviación obtenida con los datos que tratamos de una forma más significativa.

```
> tchebychev
```

```
function(data, limit=10){
  desviacion_tipica = desviacionTipica(data)
```

```
  for (k in 2:limit){
    data_percentage <- (1-(1/k^2))*100
    radious = desviacion_tipica*k
    message(paste("En un radio de",radious,"se encuentran el",
                  data_percentage,"de los datos, k=",k))
  }
}
```

5.3 Medidas de ordenación

5.3.1 Mediana

La mediana la hemos calculado ordenando los datos en orden ascendente y obteniendo el punto que se encuentra en la mitad de ese vector o la media entre los 2 puntos que se encuentran en este lugar.

```
> mediana
```

```
function(data){
  x = sort(data)
  size = length(x)
  if (size  %% 2 == 0){
    (x[size / 2] + x[(size / 2) +1]) / 2.0
  }
  else {
     x[size / 2]
  }
}
<bytecode: 0x7f9193075178>
```

5.3.2 Cuartiles

Los cuartiles los hemos calculado ordenando los datos en orden ascendente y obteniendo los 3 puntos que dividen el vector en 4 partes iguales, es decir, el cuartil 25, el 50 y el 75.

```
> cuartiles
```

```
function(data){
    x = sort(data)
    size = length(x)
    if (size  %% 2 == 0){
        c(
            x[1],
            (x[size / 4] + x[(size / 4) +1]) / 2.0 ,
            (x[size / 2] + x[(size / 2) +1]) / 2.0 ,
            (x[size *0.75] + x[size * 0.75 + 1]) / 2.0,
            x[size]
        )
    }
    else {
        c(
            x[1],
            x[size / 4],
            x[size / 2],
            (x[size *0.75] + x[size * 0.75 + 1]) / 2.0,
            x[size]
        )
  }
}
<bytecode: 0x7f9193e422e8>
```

5.3.3 Cuantil54

El cuantil 54 los hemos calculado ordenando los datos en orden ascendente y obteniendo el punto en el que detrás de él se encuentran el 54

```
> cuantil54

function(x){
  size = length(x)
  x[ceiling(size*0.54)]
}
```

5.3.4 Rango

Nos dice cual es el valor máximo y mínimo de un conjunto de datos.

```
> rango

function(data){
  c(min(data),max(data))
}
```

5.4 Otras funciones

5.4.1 Primer contacto con los datos

Nos proporciona información básica para poder comanzar a trabjar con un conjunto de datos. Muestra el nombre de las variables y su tipo. No se pueden analizar los datos si no se puede acceder a las variables que están disponibles para ser analizadas por no conocer ni su nombre.

```
> getInfo

function (data, graphics=F) {
  res <- lapply(data, class)
  res_frame <- data.frame(unlist(res))
  summary <- table(res_frame)
  if (graphics){
    barplot(summary, main="Data Types", col="steelblue", ylab="Number of Features")
  }
  message("Datos cargados: ")
  print(res_frame)
  message("Resumen: ")
  print(summary)
}
```

5.4.2 Visualización

Hemos creado una función que dibuja una gráfica de la recuencia relativa sobre la que añade líneas verticales representando las principales variables estadísticas que manejamos. Media aritmética, moda, mediana y varianza.

```
> plotFrecuencyData

function(data, xlabel="") {
  uniquedata<-unique(data)
  frecuencia_relativa <- as.table(frecuenciaRelativa(data))
  media_ <- mediaAritmetica(data)
  mediana_ <- mediana(data)
  moda_ <- moda(data)
  desviacion_tipica_ <- desviacionTipica(data)
  tchebychev_min <- media_-2*desviacion_tipica_
  tchebychev_max <- media_+2*desviacion_tipica_
  min_range = min(tchebychev_min, min(uniquedata))
  max_range = max(tchebychev_min, max(uniquedata))
  plot(frecuencia_relativa, type="h", xlab=xlabel, xlim=c(min_range,max_range))
  abline (v=c(media_, mediana_, moda_, tchebychev_min, tchebychev_max),
          col=c("blue","red","green", "gray", "gray"), lty=c(2,2,2,3,3),
          lwd=c(2,2,2,1,1))
  legend ("topright", legend=c("media", "mediana","moda", "2*desviación típica"),
          fill=c("blue","red","green","gray"))
}
<bytecode: 0x7f919496add0>
```

5.4.3 Crear documentos .pdf y .tex

Hemos automatizado este proceso con esta función para hacerlo más sencillo.

```
> saveToPdf
```

```
function(name){
  Sweave(paste(name, ".Rnw", sep=""))
  tools::texi2pdf(paste(name, ".tex", sep=""),clean=T)
}
```