

Análisis Exploratorio de datos

Copyright © M. A. Ibáñez

Este documento está disponible bajo licencia Creative Commons

Reconocimiento - NoComercial - CompartirIgual

<http://creativecommons.org/licenses/by-sa/3.0>

Actualizado el: 20 de septiembre de 2023

Tabla de Contenido

1. Introducción	4
2. Importar datos en Rstudio	5
3. Representación de variables cualitativas	28
3.1. Tabla de frecuencias	28
3.2. Gráfico de rectángulos	41
4. Representación de variables cuantitativas	45
4.1. Gráficos de puntos	45
4.2. Histograma	60
4.3. Diagrama de barras	78

5. Sumario numérico de variables cuantitativas	88
5.1. Características de tendencia central	90
●. Media aritmética	90
●. Mediana	98
5.2. Primer y tercer cuartil	114
●. Gráficos de cajas y bigotes	119
5.3. Características de dispersión	130
●. Varianza y desviación típica	132

1. Introducción

En este tema vamos a estudiar como describir con gráficos y mediante valores numéricos un conjunto de datos.

Lo normal es que los datos estén almacenado en un archivo de datos.

El primer paso es crear en R un marco de datos que almacene estos datos.

Para ello lo que se hace es importar el archivo de datos.

A continuación veremos como se realiza este proceso en RStudio cuando los datos están almacenados en una hoja excel.

2. Importar datos en Rstudio

En este apartado veremos como importar datos en R cuando están almacenados en una hoja de un archivo excel.

Ya hemos indicado anteriormente que la forma habitual de almacenar datos en archivos es mediante tablas rectangulares,

cada fila identifica a un individuo u objeto y

cada columna representa las distintas características medidas en cada individuo.

El archivo con el que vamos a trabajar se llama `Menus.xlsx`.

Es un archivo de datos creado por los alumnos de la asignatura de microeconomía en el curso 2020-21.

Se trata de estudiar como varia el precio de dos menús de comidas distintos (menú tipo 1 y menú tipo 2) según el establecimiento donde se compren los ingredientes de los dos menús.

En total, los alumnos han seleccionado 40 establecimientos.

Cada fila es un establecimiento.

En cada establecimiento tenemos la siguiente información (columnas en el archivo excel):

Persona el nombre del alumno que ha aportado la información.

Fecha la fecha en la que se obtuvo la información.

Google Map la localización en Google Map del establecimiento.

CP el código postal del distrito en el que se encuentra el establecimiento.

Renta la renta media del distrito en el que se encuentra el establecimiento según datos de Hacienda.

Establecimiento el tamaño del establecimiento, codificado de mayor a menor con las categoría 1, 2, 3 y 4.

Empresa El grupo o empresa propietaria del establecimiento.

Menu1 El precio en dicho establecimiento del menú tipo 1

Menu2 El precio en dicho establecimiento del menú tipo 2

Cuando se trabaja en R, es conveniente que el nombre de las columnas no tengan espacios en blanco, ni acentos o símbolos como ?, %, &.

Por comodidad a la hora de escribir código, tampoco se suelen utilizar nombres muy largos.

El archivo debe estar almacenado en la carpeta de trabajo y

en RStudio debemos haber vinculado un proyecto con dicha carpeta de trabajo.

El objetivo es crear un marco de datos que contenga la información almacenada en la hoja excel (importar).

Es con este marco de datos con el que luego vamos a trabajar en R.

Para importar un archivo excel en Rstudio,

vamos al panel **Files** que nos muestra los archivo almacenado en la carpeta de trabajo.

Con el botón izquierdo del ratón pulsamos sobre el archivo excel que deseamos importar,

en este ejemplo el archivo **Menus.xlsx**.

Nos abre una pequeña ventana dando la opción de visualizar los datos **View File** o importar los datos **Import Dataset....**

Seleccionamos esta segunda opción.

Se abre una ventana como la mostrada en la figura 1.

Import Excel Data

File/URL:

Data Preview:

Persona (character)	Fecha (double)	Google Map (character)	CP (double)	Renta (double)	Establecimiento (double)	Empresa (character)	Menu1 (double)
A Garrido	2020-09-25	https://g.page/Labrandero?share	28221	51746.000		2 Labrandero	10.4
A Garrido	2020-10-02	https://goo.gl/maps/LPmhK1WdND1grWBA	28223	72899.000		2 Mercadona	8.3
C Dangelo	2020-09-29	https://www.google.es/maps/place/Carrefour+Express/@40...	28015	41019.000		4 Carrefour Express	11.6
C Dangelo	2020-09-29	https://www.google.es/maps/place/Ahorramas/@40.430934...	28039	48985.000		3 Ahorramas	8.8
C Gugel	2020-10-03	https://goo.gl/maps/6V16mcezykWiXe758	28014	61367.000		2 Dia	7.5
C Gugel	2020-10-09	https://g.page/carrefour-market-tirso-de-molina?share	28012	28823.000		2 Carrefour	7.4

Previewing first 50 entries.

Import Options:

Name: Max Rows: ☒ First Row as Names

Sheet: Skip: ☒ Open Data Viewer

Range: NA:

Code Preview:

```
library(readxl)
Menus <- read_excel("Menus.xlsx")
view(Menus)
```

? Reading Excel files using readxl

Figura 1: Importar un archivo excel en Rstudio

En **File/URL** nos muestra la ruta y el nombre del archivo que queremos importar.

En **Data Preview**: se muestra las primeras 50 filas de la hoja excel que queremos importar. En cada columna nos muestra, el nombre de la columna y la clase del vector que va a crear cuando se importe dicha columna (**double** es lo mismo que **numeric**).

En general, la función que importa el archivo detecta de forma correcta el tipo de datos almacenado en cada columna.

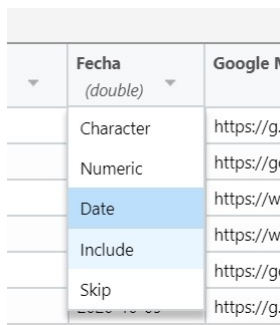
Sin embargo, se puede cambiar pinchando con el botón derecho del ratón sobre el indicador de la clase.

En este ejemplo podemos cambiar la clase de la columna **Fecha** y selec-

cionar la clase `Date` que es una clase especial que tiene R para almacenar fechas.

En la figura 2 se muestra como hacerlo.

También existe la opción `Skip` para indicar que dicha columna no se importe.



The image shows a screenshot of the 'Import Data' dialog box in RStudio. The 'Fecha' column is selected, and its type is currently set to '(double)'. A dropdown menu is open, showing the following options: Character, Numeric, Date (highlighted in blue), Include, Skip, and 2023-10-03. The 'Google M' column is also visible, with its type set to 'double'.

	Fecha	Google M
	(double)	
	Character	https://g.
	Numeric	https://g.
	Date	https://w
	Include	https://w
	Skip	https://g.
	2023-10-03	https://g.

Figura 2: Cambiar el tipo de clase de la columna Fecha

En **Import Options**: hay varias opciones de importar que podemos cambiar.

Name permite asignar un nombre al marco de datos que se va a crear.

Por defecto utiliza el nombre del archivo excel que se va a importar.

Sheet que hoja del archivo excel se va a importar.

Default indica que se va a importar la primera hoja excel del archivo.

Pinchando con el botón derecho del ratón aparece un desplegable con las hojas disponibles (en este ejemplo solo la hoja 1).

Range le indica el rango de filas y columnas de la hoja excel que se desea

importar. Por defecto toda la hoja excel.

NA le indica que símbolo se utiliza en la hoja excel para saber que en dicha celdilla no hay observación.

Por defecto se considera que la celdilla esta vacía cuando no hay datos.

Max Rows para indicar el máximo número de filas se desea importar. Si no se indica nada se importan todas las filas.

Skip para indicarle si se quiere que no importe las primeras filas iniciales de la hoja excel.

El número que se indique corresponde a las filas que no se incluyen. Por defecto no se salta ninguna fila.

La casilla activada **First Row as Names** le está indicando que en la primera fila del archivo excel están los nombres de las columnas.

La casilla activada **Open Data Viewer** le está indicando que después de importar el archivo, abra el visor de datos para visualizar el marco de datos creado.

En **Code:Preview** nos muestra las instrucciones que necesitamos utilizar en R para hacer todo esto.

En la parte superior derecha aparece el icono de un bloc de nota que permite copiar el código y pegar en el editor de instrucciones, tal y como se muestra a continuación.

```
> library(readxl)
> Menu <- read_excel("Menu.xlsx")
> View(Menu)
```

La primera instrucción está cargando el paquete `readxl` que es el que tiene la función `read_excel()` para importar hojas de datos excel.

La segunda instrucción utiliza la función `read_excel()` para importar la hoja de datos excel. En este caso solo utiliza como argumento el nombre, entre comillas, del archivo excel que se quiere importar.

La tercera instrucción utiliza la función `View()` para visualizar el marco de datos `Menu`. En el panel de edición se abre una pestaña con el nombre del marco de datos donde podemos visualizar los datos.

El objeto **Menus** que crea la función `read_excel()` es de la clase `tibble` que una clase especial de marco de datos creada por los desarrolladores del paquete `readxl` además de otros paquetes específicos para el análisis de datos con marcos de datos.

Nosotros no vamos a trabajar con estos paquetes y por lo tanto, vamos a utilizar la función `as.data.frame()` para cambiar la clase del objeto **Menu** a la clase marco de datos.

```
> ## Cambia la clase del objeto Menu a un marco de datos  
> Menus <- as.data.frame(Menus)
```

Esta instrucción esta creando el objeto **Menus** a partir del objeto ya creado **Menus** cambiando a la clase marco de datos.

Podemos comprobar el cambio realizado utilizando la función `str()`.

```
> ## Muestra la estructura del objeto Menus
```

```
> str(Menus)
```

```
'data.frame': 40 obs. of 9 variables:
```

```
$ Persona      : chr  "A Garrido" "A Garrido" "C Dangelo" "C Dang
```

```
$ Fecha        : POSIXct, format: "2020-09-25" ...
```

```
$ Google Map    : chr  "https://g.page/Labrandero?share" "https://
```

```
$ CP           : num  28221 28223 28015 28039 28014 ...
```

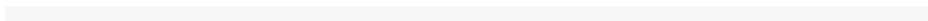
```
$ Renta        : num  51746 72899 41019 48985 61367 ...
```

```
$ Establecimiento: num  2 2 4 3 2 2 2 2 3 2 ...
```

```
$ Empresa      : chr  "Labrandero" "Mercadona" "Carrefour Express
```

```
$ Menu1        : num  10.46 8.3 11.62 8.86 7.58 ...
```

```
$ Menu2        : num  15.1 11.3 13.9 13.8 11.9 ...
```



La función nos indica que **Menu** es un marco de datos con 40 filas (observaciones) y 9 columnas (variables). A continuación muestra cada variable y de que tipo es.

Si queremos convertir una variable en un factor podemos utilizar la función `as.factor()`.

Por ejemplo, podemos convertir en factor la variable **Establecimiento** con la siguiente instrucción.

```
> ## Convierte en factor la columna Empresa de l marco de datos Menu  
> Menu$Establecimiento <- as.factor(Menu$Establecimiento)
```

Esta instrucción le está indicando que sustituya la columna **Establecimiento** del marco de datos **Menus** por la misma columna pero convertida en factor.

Comprobamos el cambio realizado utilizando de nuevo la función `str()`


```
> ## Muestra la estructura del objeto Menus
```

```
> str(Menus)
```

```
'data.frame': 40 obs. of 9 variables:
```

```
$ Persona      : chr  "A Garrido" "A Garrido" "C Dangelo" "C Dang
```

```
$ Fecha        : POSIXct, format: "2020-09-25" ...
```

```
$ Google Map    : chr  "https://g.page/Labrandero?share" "https://
```

```
$ CP           : num  28221 28223 28015 28039 28014 ...
```

```
$ Renta        : num  51746 72899 41019 48985 61367 ...
```

```
$ Establecimiento: Factor w/ 4 levels "1","2","3","4": 2 2 4 3 2 2
```

```
$ Empresa      : chr  "Labrandero" "Mercadona" "Carrefour Express
```

```
$ Menu1        : num  10.46 8.3 11.62 8.86 7.58 ...
```

```
$ Menu2        : num  15.1 11.3 13.9 13.8 11.9 ...
```

Ahora la variable `Establecimiento` ya no es numérica sino que es un factor con 4 niveles.

Utilizamos la función `levels()` para que nos muestre los niveles del factor `Establecimiento`.

```
> ## Muestra los niveles del factor Establecimiento del marco de datos
> levels(Menus$Establecimiento)

[1] "1" "2" "3" "4"
```

A partir de este momento,

cada vez que tengamos que importar una hoja excel de datos utilizaremos estas instrucciones (la instrucción con la función `View()` no es necesaria)

junto con la instrucción `as.data.frame()` y

si es necesario convertir alguna columna en factor utilizamos la función `as.factor()`.

```
> library(readxl)
> Menus <- read_excel("Menus.xlsx")
> Menus <- as.data.frame(Menus)
```

3. Representación de variables cualitativas

3.1. Tabla de frecuencias

Cuando en un conjunto de datos hay una variable cualitativa, cada unidad medida (fila) es asignada a una de las categorías de la variable.

Para describir la variable calculamos el número de observaciones o unidades que hay en cada categoría (frecuencia) o bien, la proporción sobre el total de observaciones (frecuencia relativa).

Si tenemos un conjunto de datos con N observaciones y una variable cualitativa que tiene s categorías (C_1, C_2, \dots, C_s) , podemos construir la siguiente tabla de frecuencias:

Categoría	Frecuencia	Frecuencia relativa
C_1	N_1	$f_1 = \frac{N_1}{N}$
C_2	N_2	$f_2 = \frac{N_2}{N}$
\vdots	\vdots	\vdots
C_i	N_i	$f_i = \frac{N_i}{N}$
\vdots	\vdots	\vdots
C_s	N_s	$f_s = \frac{N_s}{N}$
	N	1

Ejemplo (Censo Ganadero). En el archivo `CensoGan_Madrid2009.xlsx` se recoge la información del tipo de ganadería de las explotaciones ganaderas de la comunidad de Madrid en el año 2009.

El archivo contiene dos columnas: `Explo` es el identificador de la explotación ganadera y `Tip.Gand` es una variable cualitativa que identifica el tipo de ganadería.

Para leer el archivo utilizamos la función `read_excel()`, tal y como vimos en el apartado anterior y se indica a continuación:

```
> ## Importar hoja de datos excel  
> Censo <- read_excel("CensoGan_Madrid2009.xlsx")  
> ## Transforma en un marco de datos  
> Censo <- as.data.frame(Censo)
```


Hemos creado el objeto **Censo** que es un marco de datos.

```
> ## Muestra la estructura del objeto Censo
> str(Censo)

'data.frame': 2799 obs. of  2 variables:
 $ Explo   : num  1 2 3 4 5 6 7 8 9 10 ...
 $ Tip.Gand: chr  "Equinos" "Bovinos" "Ovinos" "Aves" ...
```

A continuación se muestran las 5 primeras observaciones con la función `head()`.

```
> ## las primeras explotaciones  
> head(Censo,n=5)
```

	Explo	Tip.Gand
1	1	Equinos
2	2	Bovinos
3	3	Ovinos
4	4	Aves
5	5	Equinos

En este archivo, cada explotación es una unidad u observación.

En total hay $N = 2799$ explotaciones.

La variable `Tip.Gand` es una variable cualitativa.

Cada explotación pertenece a una de las categorías de la variable `Tip.Gand`.

La convertimos en factor con la función `as.factor()`.

```
> ## Convierte en factor la variable Tip.Gand del marco de datos Cen  
> Censo$Tip.Gand <- as.factor(Censo$Tip.Gand)
```

Las distintas categorías (niveles del factor) de la variable las obtenemos utilizando la función `levels()`

```
> ## Niveles del factor Tip.Gand  
> levels(Censo$Tip.Gand)  
  
[1] "Aves"      "Bovinos"  "Caprino"  "Conejas"  "Equinos"  "Ovinos"  
[7] "Porcino"
```

El número de categorías, s , lo obtenemos con la función `nlevels()`:

```
> ## número de niveles del factor Tip.Gand  
> nlevels(Censo$Tip.Gand)  
  
[1] 7
```

Para construir la tabla de frecuencias, podemos utilizar la función `table()`, con el factor como argumento:

```
> ## Frecuencia en cada categoria de la variable Tip.Gand  
> frec <- table(Censo$Tip.Gand)  
> frec
```

Aves	Bovinos	Caprino	Conejas	Equinos	Ovinos	Porcino
348	1209	157	41	584	393	67

De esta forma vemos que de las $N = 2799$ explotaciones, $N_1 = 348$ son explotaciones de Aves, $N_2 = 1209$ son explotaciones de Bovinos, etc.

Para obtener las frecuencias relativas solo tenemos que dividir las frecuencias por el número total de explotaciones

o, alternativamente, utilizar la función `prop.table()` utilizando como argumento el objeto `frec` que hemos creado:

```
> ## Frecuencia relativa en cada categoría de la variable Tip.Gand  
> frec.rel <- prop.table(frec)  
> frec.rel
```

Aves	Bovinos	Caprino	Conejas	Equinos	Ovinos	Porcino
0.124330	0.431940	0.056091	0.014648	0.208646	0.140407	0.023937

Como cabría esperar, la suma de las frecuencias relativas de todas las modalidades es igual a 1.

```
> ## Verificamos que la suma de las frecuencias relativas es igual a 1  
> sum(frec.rel)
```

```
[1] 1
```


3.2. Gráfico de rectángulos

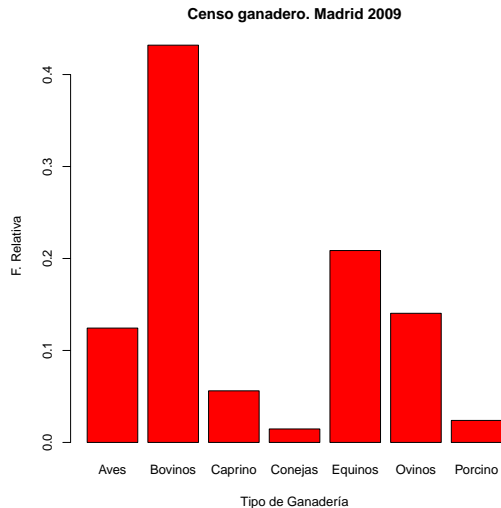
La tabla de frecuencias puede representarse mediante un **gráfico de rectángulos**,

donde cada modalidad es representada por un rectángulo cuya altura es la frecuencia relativa de la modalidad (también se puede representar la frecuencia).

Todos los rectángulos tienen la misma anchura y esta se elige a voluntad.

Para realizar dicho gráfico utilizamos la función `barplot()`, con el vector de frecuencias como argumento.

```
> ## Gráfico de frecuencias de la variable Tipo de Ganadería  
> barplot(frec.rel,col='red',  
+         main='Censo ganadero. Madrid 2009',  
+         xlab='Tipo de Ganadería',ylab='F. Relativa')
```



La función `bar.plot()` es una de las muchas funciones que tiene R para realizar gráficos.

La mayoría de las funciones gráficas tienen los mismos argumentos que hemos utilizado en esta instrucción y que se explican a continuación.

`main`= El título del gráfico. Va entre comillas

`xlab`= El título del eje horizontal (eje X o de abscisas). Va entre comillas

`ylab`= El título del eje vertical (eje Y o de ordenadas). Va entre comillas

`col`= El color de los rectángulos. Puedes ser un número o un carácter entre comillas. En este ejemplo hemos utilizado `'red'` de rojo en inglés. Puedes teclear la función `palette()` o la función `colors()` en la consola de R para ver nombres de colores que se pueden utilizar.

4. Representación de variables cuantitativas

4.1. Gráficos de puntos

Cuando el número de observaciones es pequeño, no superior a 50, una forma adecuada de representar los valores de una variables cuantitativa son los gráficos de puntos.

En un gráfico de punto, se utiliza un eje, normalmente el horizontal, para representar los valores de la variable en cada observación.

Cada observación es un punto en el eje y su posición depende del valor de la variable para esa observación.

Ejemplo (Calidad del aire en Madrid). En el archivo `Aire_Madrid_2019.xlsx` se recoge la concentración de distintas magnitudes, tales como NO , SO_2 , O_3 , etc., en el aire de Madrid medidas diariamente durante el año 2019 en las distintas estaciones de medición que tiene el ayuntamiento de Madrid.

Para importar dicho archivo utilizamos la función `read_excel()`.

```
> ## Importa hoja de datos Aire_Madrid_2019.xlsx
> Aire <- read_excel("Aire_Madrid_2019.xlsx",
+   col_types = c("text", "text", "numeric",
+   "numeric", "numeric", "numeric",
+   "numeric", "numeric", "numeric",
+   "numeric", "numeric"))
>
> ## Convierte el objeto Aire en un marco de datos
> Aire <- as.data.frame(Aire)
```

En este ejemplo hemos utilizado el argumento `col_type` para indicar de que clase es cada columna en la hoja excel de datos. Se ha hecho esto porque en las primeras filas de algunas de las columnas no hay observaciones (están en blanco) y la función `read_excel()` tiene problemas para detectar de que clase son dichas columnas.

A continuación se utiliza la función `str()` para obtener información del marco de datos `Aire`.

```
> ## Muestra estructura del objeto Aire  
> str(Aire)
```

```
'data.frame': 8928 obs. of  11 variables:  
 $ Estacion: chr  "Av. Ramon y Cajal" "Av. Ramon y Cajal" "Av. Ramon  
 $ Mes      : chr  "Ene" "Ene" "Ene" "Ene" ...  
 $ Dia      : num  1 2 3 4 5 6 7 8 9 10 ...  
 $ NO       : num  39 93 50 66 70 84 49 74 26 8 ...  
 $ NO2      : num  68 87 66 78 82 94 81 81 60 35 ...  
 $ NOx      : num  127 229 142 179 189 223 155 195 101 47 ...  
 $ CO       : num  NA NA NA NA NA NA NA NA NA NA ...  
 $ O3       : num  NA NA NA NA NA NA NA NA NA NA ...
```

```
$ S02      : num  NA NA NA NA NA NA NA NA NA NA NA NA ...  
$ PM10     : num  NA NA NA NA NA NA NA NA NA NA NA NA ...  
$ PM2.5    : num  NA NA NA NA NA NA NA NA NA NA NA NA ...
```

De este archivo sólo vamos a utilizar la concentración de O_3 (Ozono) medida en el mes de Enero en la estación de Barajas. La concentración se mide en $\mu g/m^3$.

Utilizamos la función `subset()` para seleccionar estas observaciones.

```
> ## Selecciona solo las estación de Barajas y el mes de Enero. Solo  
> ## las variables Dia y O3  
> Barajas.Ene <- subset(Aire,subset=Estacion=='Barajas' &  
+      Mes=='Ene',select=c('Dia','O3'))
```

El marco de datos `Barajas.Ene` contiene dichas observaciones.

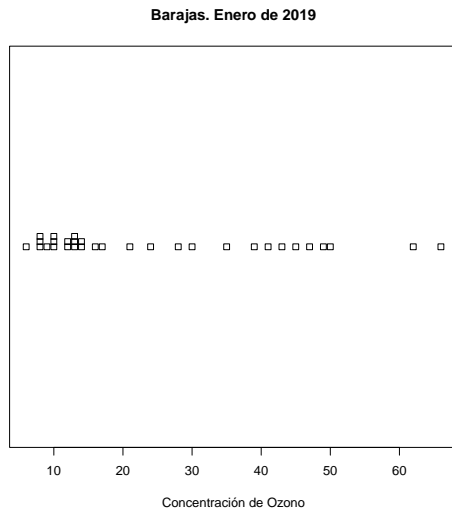
```
> ## Muestra que hay almacenado en el objeto Barajas.Ene
> str(Barajas.Ene)

'data.frame': 31 obs. of  2 variables:
 $ Dia: num  1 2 3 4 5 6 7 8 9 10 ...
 $ 03 : num  12 8 13 10 9 13 14 12 24 41 ...
```

El número de observaciones es de $N = 31$.

Para representar con un gráfico las concentraciones durante los 31 días del mes de Enero, utilizamos el gráfico de puntos, que en R podemos obtener con la función `stripchart()`:

```
> ## Gráfico de puntos para la variable O3  
> stripchart(Barajas.Ene$O3, main='Barajas. Enero de 2019',  
+           xlab='Concentración de Ozono', method='stack')
```



El argumento `method` se utilizará para separar los puntos coincidentes (con valores similares).

El valor predeterminado es `'overplot'` que hace que dichos puntos se sobrescriban.

Es posible especificar `'jitter'` para separar los puntos verticalmente, o `'stack'` para apilar los puntos coincidentes.

En este ejemplo hemos utilizado `stack`, pero podéis utilizar `jitter` para ver la diferencia.

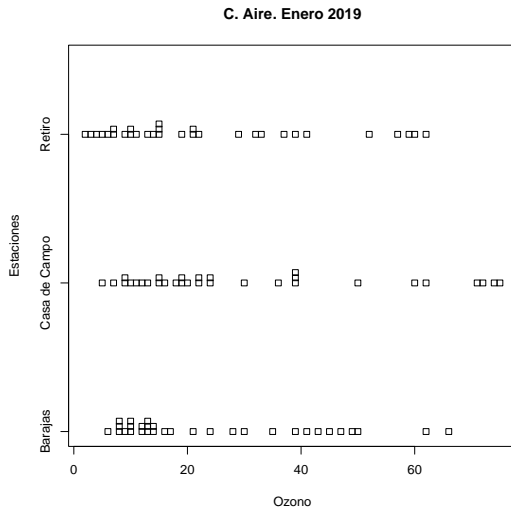
También podemos utilizar la función `stripchart()` para representar la concentración de O_3 en dos o más estaciones.

En el siguiente ejemplo seleccionamos la concentración de O_3 en el mes de Enero de las estaciones de Barajas, Retiro y Casa de Campo:

```
> ## Selecciona las estaciones de Barajas, Retiro y Casa de Campo
> ## Solo registros del mes de Enero y solo almacena las variables
> ## Estacion, Dia y O3
> Esta.Ene <- subset(Aire,
+   subset=Estacion%in%c('Barajas','Retiro','Casa de Campo')
+   & Mes=='Ene',select=c('Estacion','Dia','O3'))
```


El siguiente gráfico muestra la concentración de O_3 en el mes de Enero en las tres estaciones seleccionadas.

```
> ## Gráfico de puntos de la variable O3 separado para cad estación  
> stripchart(O3~Estacion,data=Esta.Ene,method="stack",  
+           main="C. Aire. Enero 2019",xlab="Ozono",  
+           ylab="Estaciones")
```



4.2. Histograma

Un histograma es el gráfico que se suele utilizar para representar la distribución de una variable cuantitativa cuando el número de observaciones es elevado, normalmente mayor de 50.

Para realizar un histograma,

el rango de variación de la variable (diferencia entre el valor máximo y mínimo) se divide en intervalos, normalmente de la misma amplitud,

y se calcula la frecuencia de observaciones que están dentro de cada intervalo.

En el eje de abscisas se representan los intervalos y en el eje de ordenadas la frecuencia de cada intervalo, levantando un rectángulo cuya anchura es la amplitud del intervalo y la altura la frecuencia del intervalo.

El número y amplitud de los intervalos puede elegirse a voluntad aunque existen diferentes algoritmos que determinan el número de intervalos necesarios para una óptima representación de la variable.

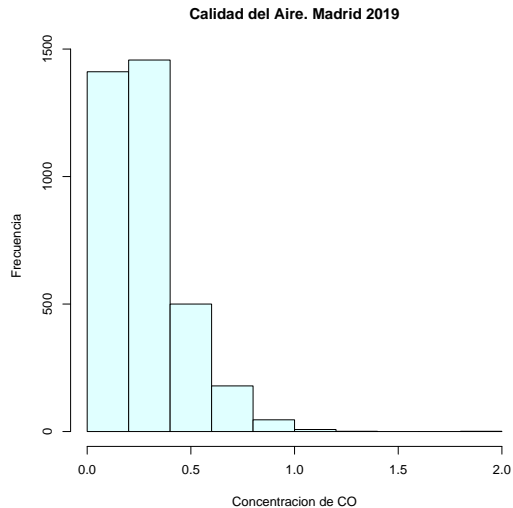
En R disponemos de la función `hist()` para construir un histograma.

El argumento principal de esta función es el vector numérico que deseamos representar.

Por defecto utiliza el algoritmo de *Sturges* para determinar el número y amplitud de los intervalos.

Con la siguiente instrucción representamos el histograma de la concentración diaria de la variable CO en Madrid a lo largo del año 2019:

```
> # histograma de la concentración diaria de la variable CO  
> hist(Aire$CO,main="Calidad del Aire. Madrid 2019",  
+      xlab="Concentracion de CO",ylab="Frecuencia",col="lightcyan")
```

El número de observaciones del vector `Aire$CO` son las mediciones diarias durante el año 2019 en las 24 estaciones de medición distribuidas por Madrid.

Hay días y estaciones donde no se dispone de dicha observación.

Si queremos saber de cuantas observaciones disponemos, podemos utilizar la función `na.omit()` para eliminar del vector los datos faltantes (NA o NaN).

```
> ## Longitud del vector Aire$CO  
> length(Aire$CO)
```

```
[1] 8928
```

```
> ## Longitud del vector Aire$CO eliminando los datos  
> ## faltantes con la funcion na.omit()  
> length(na.omit(Aire$CO))
```

```
[1] 3603
```

La función `hist()`, además de hacer el histograma, nos proporciona información sobre los intervalos creados y la frecuencia de cada intervalo.

Para ello, sólo es necesario asignar el resultado de la función a un objeto donde se guarda dicha información.

De forma que, si ejecutamos la siguiente instrucción (se ha utilizado el argumento `plot=FALSE` para no volver a representar el histograma):

```
> ## Intervalos y frecuencias creado por la función hist()  
> tabla <- hist(Aire$CO,plot=FALSE)
```

El objeto `tabla` es una lista que tiene, entre otros, los siguientes componentes:

- El componente `break`, con los límites de los intervalos creados.

```
> ## Límites de los intervalos creados  
> tabla$breaks  
  
[1] 0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4 1.6 1.8 2.0
```

La función ha creado 10 intervalos. El primer intervalo va desde el valor 0 hasta el valor 0.2, el segundo desde el valor 0.2 hasta el valor 0.4 y así sucesivamente.

- El componente `counts`, con la frecuencia observada en cada intervalo.

```
> ## Frecuencia de observaciones en cada intervalo  
> tabla$counts  
  
[1] 1411 1457 500 179 46 8 1 0 0 1
```

Podemos comprobar, que la suma de este vector de frecuencias coincide con el total de observaciones en el vector `Aire$CO`, una vez eliminados los datos faltantes.

```
> ## Verificamos que la suma es igual al total de observaciones  
> sum(tabla$counts)  
  
[1] 3603
```

La función `hist()` tiene el argumento `breaks` para cambiar el número de intervalos que debe utilizar. Admite distintas opciones.

Pueden ser:

- un vector con los límites de los intervalos.
- un número que indica el número de intervalos en el histograma.
- Una cadena de caracteres con el nombre del algoritmo utilizado para calcular el número de intervalos.

Por defecto la función tiene asignado el valor `breaks='Sturges'`, pero puede utilizarse también `'Scott'` y `'FD'` (*Freedman-Diaconis*).

La función `hist()` también tiene el argumento `freq` que admite un valor lógico. `TRUE` para indicar que en el eje de ordenadas represente las frecuencias en cada intervalo.

Si utilizamos el valor `FALSE`, la altura de los histogramas se calcula para que el área de cada rectángulo sea igual a la frecuencia relativa en cada intervalo.

De esta forma la suma de las áreas de todos los rectángulos es igual a 1.

Si N_i es la frecuencia y f_i la frecuencia relativa en el intervalo i con amplitud a .

Entonces, para que el área del rectángulo sea igual a la frecuencia relativa, la altura de dicho intervalo (h_i) ha de cumplir la siguiente igualdad:

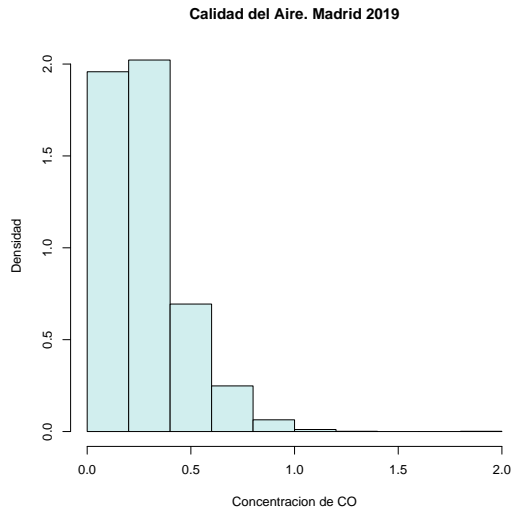
$$h_i \times a = f_i$$

y por tanto $h_i = \frac{f_i}{a}$.

A este ratio se le denomina densidad de frecuencia (frecuencia por unidad de longitud).

La siguiente instrucción construye el histograma de la variable `CO` con el argumento `freq=FALSE`:

```
> ## histograma de la variable CO
> hist(Aire$CO,main="Calidad del Aire. Madrid 2019",
+      xlab="Concentracion de CO",ylab="Densidad",
+      col="lightcyan2",freq=FALSE)
```



4.3. Diagrama de barras

Un caso particular del histograma es cuando la variable es discreta.

Como sólo puede tomar un número finito de valores dentro de un intervalo, no tiene mucho sentido construir intervalos de valores tal y como hacemos en un histograma.

El archivo `Vivienda.xlsx` contiene los precios de ventas 93 viviendas obtenidos del portal `pisos.com` en el año 2015.

Como siempre, para importar dicho archivo utilizamos la función `read_excel()`.

```
> ## Importar el archivo Vivienda.xlsx  
> Vivienda <- read_excel("Vivienda.xlsx")  
> ## Convierte el objeto en un marco de datos  
> Vivienda <- as.data.frame(Vivienda)
```


El archivo contiene 93 observaciones y 6 variables.

La variable `Habit`, que contiene el número de habitaciones, es una variable discreta.

Para representar dicha variable, construimos una tabla con la frecuencia de cada uno de los valores de la variable `Habit`,

de la misma forma que hicimos cuando teníamos una variable cualitativa.

Utilizamos para ello las funciones `table()` y `prop.table()`.

```
> ## Frecuencia relativa del número de habitaciones en las viviendas  
> tabla=prop.table(table(Vivienda$Habit))  
> tabla
```

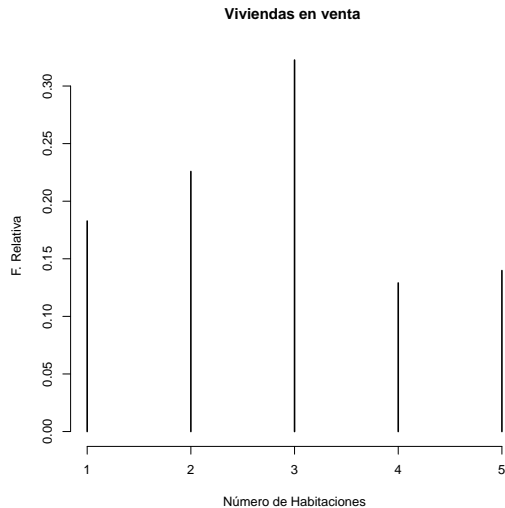
1	2	3	4	5
0.18280	0.22581	0.32258	0.12903	0.13978

El diagrama de barras, representa cada valor de la variable `Habit` con una barra cuya altura es la frecuencia relativa de dicho valor.

La interpretación es la misma que la de un histograma, destacando los valores más frecuentes y los menos frecuentes.

Para obtener dicho gráfico, utilizamos la función `plot()` con el objeto `tabla` como argumento.

```
> ## Diagrama de barras del número de habitaciones  
> plot(tabla,xlab="Número de Habitaciones",ylab="F. Relativa",  
+       main="Viviendas en venta ",frame=FALSE)
```



La función `plot()` es una función gráfica genérica que permite realizar distintos tipos de gráficos.

En este ejemplo, además hemos utilizado el argumento `frame=FALSE` para eliminar el marco (recuadro) alrededor del gráfico.

5. Sumario numérico de variables cuantitativas

Cuando la variable es cuantitativa, además de realizar representaciones gráficas, podemos calcular valores numéricos que resuman la distribución de la variable.

A este tipo de valores numéricos se les suele llamar **características**.

En general hablamos de características de **tendencia central** para referirnos a las características que nos indican alrededor de que valor numérico se distribuye la variable estudiada.

Denominamos como características de **dispersión** a las que nos proporcionan información sobre la variabilidad de la variable estudiada.

5.1. Características de tendencia central

- **Media aritmética**

La característica de tendencia central más utilizada es la **media** o media aritmética.

Es la suma de todos los valores de la variable dividida por el número de valores sumados.

Si denominamos por y_i al valor de la variable y en la observación i -ésima y por \bar{y} a la media, su cálculo se expresa matemáticamente de la siguiente forma:

$$\bar{y} = \frac{1}{N} \sum_{i=1}^N y_i$$

En la figura 3 se muestra el gráfico para la concentración de ozono en la estación de Barajas en el mes de Enero de 2019 y el valor de la media.

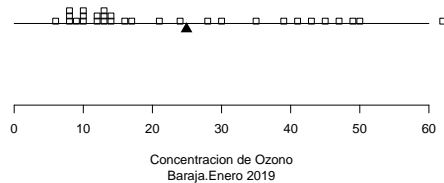


Figura 3: Representación de la concentración de Ozono y el valor medio en Enero de 2019 en Barajas(Madrid)

En R tenemos la función `mean()` para calcular la media de cualquier vector numérico.

En el ejemplo de la calidad del aire en Madrid 2019, podemos calcular la media de la concentración de *CO*.

```
> ## Media de la variable CO en Madrid durante 2019
> med.CO <- mean(Aire$CO,na.rm=TRUE)
> ## Muestra el valor almacenado en med.CO
> med.CO

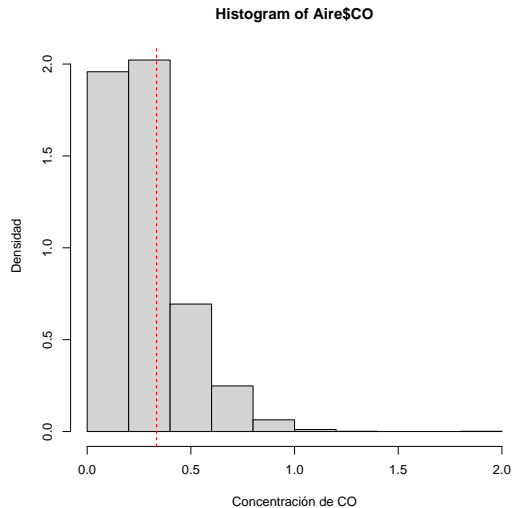
[1] 0.33419
```

hemos utilizado el argumento `na.rm=TRUE` para que no tenga en cuenta

los datos faltantes.

La dos siguientes instrucciones realiza el histograma de la variable `CO` y dibuja una línea vertical que representa la media de la variable.

```
> ## Histograma de la variable CO y su valor medio  
> hist(Aire$CO,freq=FALSE,xlab="Concentración de CO",  
+      ylab="Densidad")  
> abline(v=med.CO,col="red",lty=2)
```



La función `abline()` con el argumento `v=med.CO` nos permite dibujar una línea vertical en el valor de `med.CO` de color rojo (argumento `col='red'`) y que sea discontinua (con el argumento `lty=2`).

- **Mediana**

La mediana de una variable es el valor numérico que divide a las observaciones en dos mitades (50 %).

La mitad de las observaciones con valores menores a la mediana y la mitad de las observaciones con valores superiores.

Tal y como se ha definido, la mediana es un cuantil de orden 0.5.

En la figura 4 se muestran los valores de concentración de ozono medidos en Barajas en Enero del 2019,

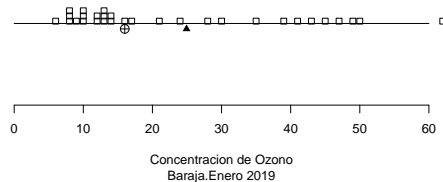


Figura 4: Representación de la concentración de Ozono y el valor medio (triángulo negro) y la mediana (circulo con una cruz) en Enero de 2019 en Barajas (Madrid)

En este segundo ejemplo, la media y la mediana son valores muy distintas.

Esto es debido a que el valor de la media se ve muy afectado si hay observaciones con valores muy extremos (en este caso las concentraciones de ozono de $45\mu g/m^3$ y superiores).

Sin embargo, en el cálculo de la mediana estos valores extremos no le afectan, podrían ser mayores de 60 y la mediana seguiría siendo 16.

Para calcular la mediana en R tenemos la función `median()`.

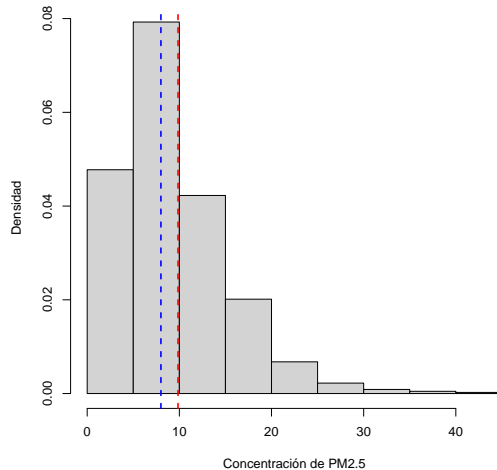
Para los valores de la variable 03 en `Barajas.Ene`.

```
> ## Mediana de la variable 03 en Barajas en el mes de Enero  
> median(Barajas.Ene$03)
```

```
[1] 16
```

La siguientes instrucciones realiza el histograma de la variable PM2.5 (concentración de partículas de menos de $2.5\mu m$) y dibuja dos linea verticales que representa la media de la variable (color rojo) y la mediana (en color azul).

```
> ## media de la variable PM2.5
> med.PM2.5 <- mean(Aire$PM2.5,na.rm=TRUE)
> ## mediana de la variable PM2.5
> medn.PM2.5 <- median(Aire$PM2.5,na.rm=TRUE)
> ## histograma
> hist(Aire$PM2.5,freq=FALSE,xlab="Concentración de PM2.5",
+      ylab="Densidad",main="")
> abline(v=med.PM2.5,col="red",lty=2,lwd=2)
> abline(v=medn.PM2.5,col="blue",lty=2,lwd=2)
```



Observando el histograma, comprobamos que la distribución de la variable PM2.5 en el marco de datos **Aire** no es simétrica.

Consideramos como eje de simetría una recta paralela al eje de ordenadas que pasa por la media de la distribución (línea de color rojo).

Si una distribución es simétrica, existe el mismo número de valores a la derecha que a la izquierda de la media, por tanto, el mismo número de desviaciones con signo positivo que con signo negativo.

Decimos que hay asimetría positiva (o a la derecha) si la cola a la derecha de la media es más larga que la de la izquierda, es decir, si hay valores más separados de la media a la derecha.

Diremos que hay asimetría negativa (o a la izquierda) si la cola a la izquierda de la media es más larga que la de la derecha, es decir, si hay valores más separados de la media a la izquierda.

En el ejemplo de la variable $PM2.5$ la asimetría es positiva, dado que la cola a la derecha de la media es más larga que a la izquierda.

Cuando las distribuciones son asimétricas la mediana es una mejor medida de tendencia central que la media.

A los estadísticos nos suele gustar trabajar con distribuciones simétricas.

Muchos de los métodos estadísticos que utilizamos asumen que la distribución de la variable es simétrica, en concreto que tienen una distribución normal (la distribución normal la estudiaremos en los siguientes temas).

Para conseguir esto, muchas veces trabajamos con la variable a la que aplicamos una transformación no lineal,

tal como la transformación logarítmica, la raíz cuadrada, la transformación inversa, etc.

Cuando trabajamos con concentraciones, como es el caso de la variable PM2.5, lo habitual es usar la transformación logarítmica.

En la siguiente instrucciones creamos una nueva columna a la que llamamos `lPM2.5` que contiene el logaritmo neperiano (función `log()`) de la variable PM2.5.

```
> ## Logaritmo de la variable PM2.5  
> Aire$lPM2.5 <- log(Aire$PM2.5)
```

y calculamos la media

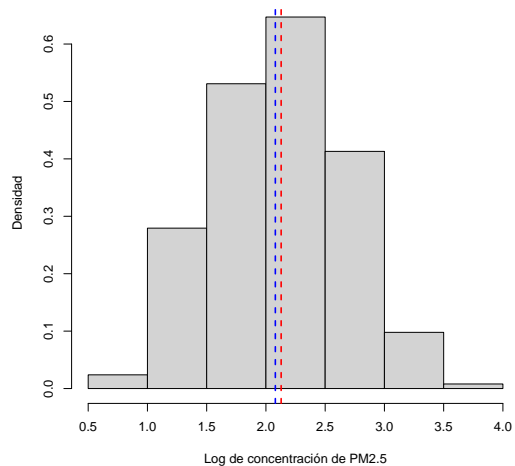
```
> ## media de la variable lPM2.5  
> med.lPM2.5 <- mean(Aire$lPM2.5,na.rm=TRUE)  
> med.lPM2.5  
  
[1] 2.1285
```

y la mediana de la nueva variable transformada.

```
> ## mediana de la variable lPM2.5  
> medn.lPM2.5 <- median(Aire$lPM2.5, na.rm=TRUE)  
> medn.lPM2.5  
  
[1] 2.0794
```

El siguiente código muestra el histograma, la media y la mediana de la variable transformada `lPM2.5`.

```
> # histograma, la media y la mediana de la variable transformada lPM2.5
> hist(Aire$lPM2.5,xlab="Log de concentración de PM2.5",
+      freq=FALSE,ylab="Densidad",main="",breaks=10)
> abline(v=med.lPM2.5,col="red",lty=2,lwd=2)
> abline(v=medn.lPM2.5,col="blue",lty=2,lwd=2)
```



5.2. Primer y tercer cuartil

Aunque no son características de tendencia central, el primer y tercer cuartil son características muy útiles para describir la distribución de una variable.

El **primer cuartil**, es el cuantil de orden $\frac{1}{4} = 0.25$. Es el valor de la variable que divide a las observaciones en dos subconjuntos, el 25 % con valores inferiores y el 75 % con valores superiores.

El **tercer cuartil**, es el cuantil de orden $\frac{3}{4} = 0.75$. Es el valor de la variable que divide a las observaciones en dos subconjuntos, el 75 % con valores inferiores y el 25 % con valores superiores.

Para calcular cuantiles, en R disponemos de la función `quantile()`. El primer argumento es el vector numérico sobre el que se quiere calcular el cuantil. El argumento `prob` es un vector con la proporción (probabilidad) a la izquierda.

La función devuelve un vector de la misma longitud que el vector `prob`.

```
> ## Primer y tercer cuartil de 03  
> cuartil <- quantile(Barajas.Ene$03,prob=c(0.25,0.75))  
> cuartil
```

```
25% 75%  
11  40
```

En la figura 5 se muestran el primer y tercer cuartil de la concentración de ozono en Enero de 2019 medido en la estación de Barajas.

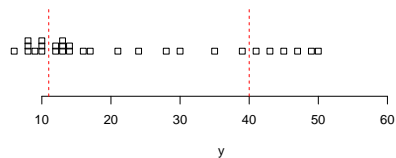


Figura 5: Representación del primer y tercer cuartil de la concentración de ozono en Enero del 2019 medido en Barajas (Madrid)

Otra opción, es utilizar la función `summary()` que nos proporciona información del valor mínimo, máximo, media, mediana, primer y tercer cuartil de la variable.

```
> summary(Barajas.Ene$03)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
6.0	11.0	16.0	24.9	40.0	66.0

- **Gráficos de cajas y bigotes**

Un gráfico de caja y bigotes, es un gráfico que está basado en los cuartiles y mediante el cual se visualiza la distribución de un conjunto de datos.

Está compuesto por un rectángulo (la caja) y dos brazos (los bigotes).

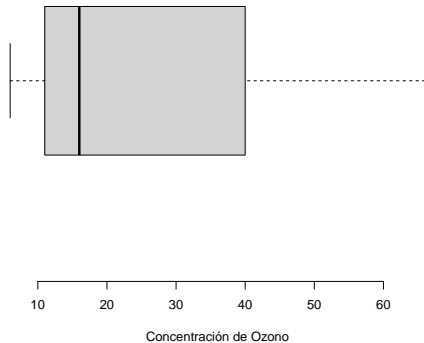
En este gráfico, los extremos de la caja son los cuartiles (primer cuartil y tercer cuartil) y los bigotes corresponden a aquellos datos que no se alejan de los extremos de la caja más de 1.5 veces la anchura de la caja.

Los puntos son observaciones extremas que se alejan más de 1.5 veces la anchura de la caja.

La línea dibujada dentro de la caja corresponde con la mediana de los datos.

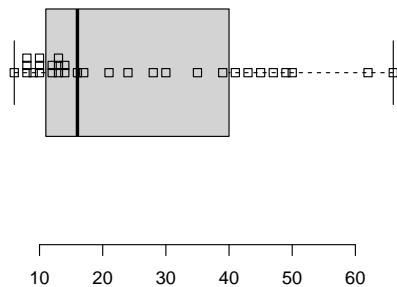
En R, para dibujar un gráfico de cajas y bigotes tenemos la función `boxplot()`.

```
> # Gráfico de cajas y bigotes de la variable O3 en Barajas durante
> boxplot(Barajas.Ene$O3, horizontal=TRUE,
+         main="Gráfico de Cajas y Bigotes",
+         xlab="Concentración de Ozono",
+         frame=FALSE)
```

Gráfico de Cajas y Bigotes

Cuando disponemos de pocas observaciones, podemos realizar una representación conjunta del gráfico de cajas y bigotes y el gráfico de punto, tal y como se hace a continuación combinando las funciones `boxplot()` y `stripchart()`.

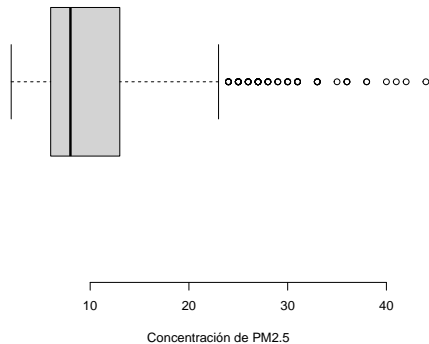
```
> # Grafico de Cajas y bigotes  
> boxplot(Barajas.Ene$03, horizontal=T, frame=FALSE)  
> # gráfico de Puntos superpuesto  
> stripchart(Barajas.Ene$03, method="stack", add=T)
```



Los gráfico de cajas y bigotes son útiles para verificar la simetría de la distribución de una variable y detectar la presencia de valores extremos.

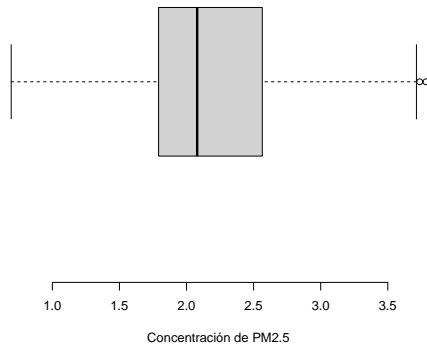
La siguiente instrucción representa el gráfico de cajas y bigotes de la variable PM2.5 del marco de datos **Aire**

```
> # Gráfico de cajas y bigotes de la variable PM2.5 en Madrid durant  
> boxplot(Aire$PM2.5,main="",frame=FALSE,  
+         xlab="Concentración de PM2.5",horizontal=TRUE)
```



Por el contrario, si realizamos el gráfico de cajas y bigotes para la misma variable con la transformación logarítmica (variable `lPM2.5`).

```
> # Gráfico de cajas y bigotes de la variable logaritmo de PM2.5 en  
> boxplot(Aire$lPM2.5, main="", frame=FALSE,  
+         xlab="Concentración de PM2.5",  
+         horizontal=TRUE)
```



5.3. Características de dispersión

Además de las características de tendencia central, es necesario también ver el grado de dispersión de las observaciones.

Determinar en que medida las observaciones son muy parecidas y por tanto muy parecidas al valor medio

o por el contrario son muy diferentes, habiendo valores mucho menores y/o mayores que la media.

Las características que solemos utilizar para describir la dispersión de un conjunto de datos son la **varianza**

y su raíz cuadrada, a la que se denomina **desviación típica**.

- **Varianza y desviación típica**

La varianza es la media de las desviaciones al cuadrado de cada observación al valor medio de la variable.

Normalmente utilizamos la letra s^2 para referirnos a la varianza de una variable y a veces añadimos un subíndice s_y^2 , para indicar que es de la variable y .

La expresión matemática de la varianza de la variable y es:

$$s_y^2 = \frac{1}{N} \sum_{i=1}^N (y_i - \bar{y})^2$$

Cuanto más pequeña sea la varianza más parecida a la media son los valores de la variable.

Si la varianza es pequeña, decimos que hay poca dispersión en los datos.

Un caso extremo, es que la varianza sea 0. En este caso, todos los valores son iguales e iguales a la media.

Para calcular la varianza con R tenemos la función `var()`.

Sin embargo, el cálculo de la varianza no lo realiza dividiendo por el número total de observaciones (N), sino que divide por $N - 1$.

$$s_y^2 = \frac{1}{N-1} \sum_{i=1}^N (y_i - \bar{y})^2$$

A continuación calculamos la varianza de la variable PM2.5 en el marco de datos `Aire`, utilizando la función `var()`.

```
> ## varianza de la variable PM2.5  
> varPM2.5 <- var(Aire$PM2.5, na.rm=TRUE)  
> varPM2.5  
  
[1] 33.294
```


Al signo positivo de la raíz cuadrada de la varianza se le denomina **desviación típica**.

$$s_y = \sqrt{s_y^2} = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (y_i - \bar{y})^2}$$

El valor de la desviación típica es más fácil de interpretar que el de la varianza.

La desviación típica puede interpretarse como la desviación media de las observaciones con respecto a la media de la variable.

La desviación típica se mide en las mismas unidades que la variable, mientras que la varianza se mide en las unidades elevadas al cuadrado.

Para calcular la desviación típica en R utilizamos la función `sd()`

```
> ## Desviación típica de la variable PM2.5  
> sd(Aire$PM2.5, na.rm=TRUE)  
  
[1] 5.7701
```

que podemos comprobar que coincide con la raíz cuadrada de la varianza (33.2935).