Tidy Data con R

José Ignacio Escribano

10 de abril de 2016

Índice

1. Carga de los datos

En primer lugar, cargamos los paquetes necesarios para limpiar los datos.

```
library("dplyr")
library("tidyr")
```

En caso de que de se muestre el siguiente error

```
Error in library(dplyr) : there is no package called 'dplyr'
Error in library(dplyr) : there is no package called 'tidyr'
```

debemos ejecutar el siguiente comando, que instalará los paquetes anteriores desde el repositorio http://cran.rstudio.com.

```
# Repositorio desde el que descargar los paquetes
repository = "http://cran.rstudio.com"
install.packages(c("dplyr", "tidyr"), repos = repository)
```

Una vez instalados los paquetes, volvemos a ejecutar los comandos anteriores.

Leemos el archivo `Air_Quality.csv`, que se encuentra en la misma carpeta que este documento usando los siguientes comandos.

```
filename = "Air_Quality.csv"
data_csv = tbl_df(read.csv(file = filename, sep = ",", header = TRUE))
```

2. Limpieza de los datos

Ya estamos en disposición para limpiar los datos. Para ello:

1. Seleccionamos de nuestro fichero de datos `data` las variables `year_description`, `geo_entity_id`, `geo_type_name`, `data_valuemessage` y `indicator_id`. Notar que en esta última variable nos servirá para filtrar aquellas filas que se corresponden con el identificador 646, que es el del benceno.
2. Filtramos las que contengan el identificador 646, es decir, el del benceno.
3. Cambiamos el nombre a las columnas.
4. Eliminamos la columna `indicator_id`.

```
attach(data_csv)
# Paso 1
selected_columns = select(data_csv,
                          year_description,
                          geo_entity_id,
                          geo_type_name,
                          data_valuemessage,
                          indicator_id)

# Paso 2
filtered_rows = filter(selected_columns, indicator_id == 646)

# Paso 3
renamed_columns = rename(filtered_rows,  year= year_description,
                         geo_entity = geo_entity_id,
                         geo_type = geo_type_name,
                         data_value = data_valuemessage)


# Paso 4
tidy_data = select(renamed_columns, -indicator_id)
```

Nuestro nuevo conjunto de datos tiene el siguiente aspecto:

```
# Sólo es necesario para mostrar el resultado de la función
# `head` como tabla en R Markdown

# Si no está el paquete pander, lo descargamos
if (!require("pander"))
  install.packages("pander", repos = repository)

# Cargamos el paquete pander
library("pander")

head_data = head(tidy_data)

pander(head_data, caption = "Subconjunto de los datos tras ser limpiados")
```

Cuadro 1: Subconjunto de los datos tras ser limpiados

year	geo_entity	geo_type	data_value
2005	1	Borough	2.8
2005	2	Borough	2.8
2005	3	Borough	4.7
2005	4	Borough	1.9
2005	5	Borough	1.6
2005	1	Citywide	2.9

Comprobamos el tamaño de este nuevo conjunto de datos:

```
dim(tidy_data)
```

```
## [1] 48  4
```

Es decir, tenemos unos datos con 48 filas y 4 columnas.

Otra forma de hacer lo anterior es usando el operador `%>%`

```
tidy_data2=data_csv %>%

  select(year_description,
         geo_entity_id,
         geo_type_name,
         data_valuemessage,
         indicator_id) %>%
```

```
  filter(indicator_id == 646)  %>%

  rename(year= year_description,
         geo_entity = geo_entity_id,
         geo_type = geo_type_name,
         data_value = data_valuemessage) %>%

  select(-indicator_id)
```

Comprobamos que las dimensiones coinciden con lo obtenido anteriormente.

```
dim(tidy_data2)
```

```
## [1] 48  4
```

Y, por último comprobamos que los vectores son idénticos, es decir, si todas las posiciones son iguales.

```
equal_index = sum(tidy_data == tidy_data2)

cat("El número de índices iguales es", equal_index)
```

```
## El número de índices iguales es 192
```

Esto nos devuelve 192 (48*4) índices que concuerdan (todos los valores son iguales posición a posición), por lo que ambos resultados son idénticos.

Otra forma más directa de comprobarlo es usar la función `identical`.

```
if(identical(tidy_data,tidy_data2)){
  cat("Los datos son idénticos")
}else{
  cat("Los datos no son idénticos")
}
```

```
## Los datos son idénticos
```

3. Estadísticos básicos

Obtenemos algunas medidas estadísticas para la columna `data_value`.

```r
# Guardamos los datos en la variable data_value
data_value = tidy_data$data_value

cat("La media es", mean(data_value), "y la media es", median(data_value))
```

```
## La media es 2.910417 y la media es 2.75
```

```r
cat("La desviación típica es", sd(data_value))
```

```
## La desviación típica es 1.166599
```

```r
cat("El mínimo es", min(data_value), "y el máximo es", max(data_value))
```

```
## El mínimo es 1.1 y el máximo es 6.3
```

```r
cat("El primer cuartil es", quantile(data_value, 0.25)[[1]],
    "y el tercer cuartil es", quantile(data_value, 0.75)[[1]])
```

```
## El primer cuartil es 1.975 y el tercer cuartil es 3.7
```

Otra forma de obtener estos indicadores es usar la función `summary`.

```r
# La función `pander` sólo es necesaria para mostrar el resultado de
# la función `summary` como tabla en R Markdown

summary = summary(data_value)
pander(summary, caption = "Estadísticos básicos de la variable")
```

Cuadro 2: Estadísticos básicos de la variable

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.1	1.975	2.75	2.91	3.7	6.3

4. Histograma

Representamos los datos de la variable `data_value` como un histograma.

```r
title_hist = "Concentración media de partículas de benceno
  en la ciudad de Nueva York"
ylabel_hist = "Frecuencia"
xlabel_hist = ""
hist(data_value,
     main = title_hist,
     xlab = xlabel_hist,
     ylab = ylabel_hist,
     col = "lightblue")
```

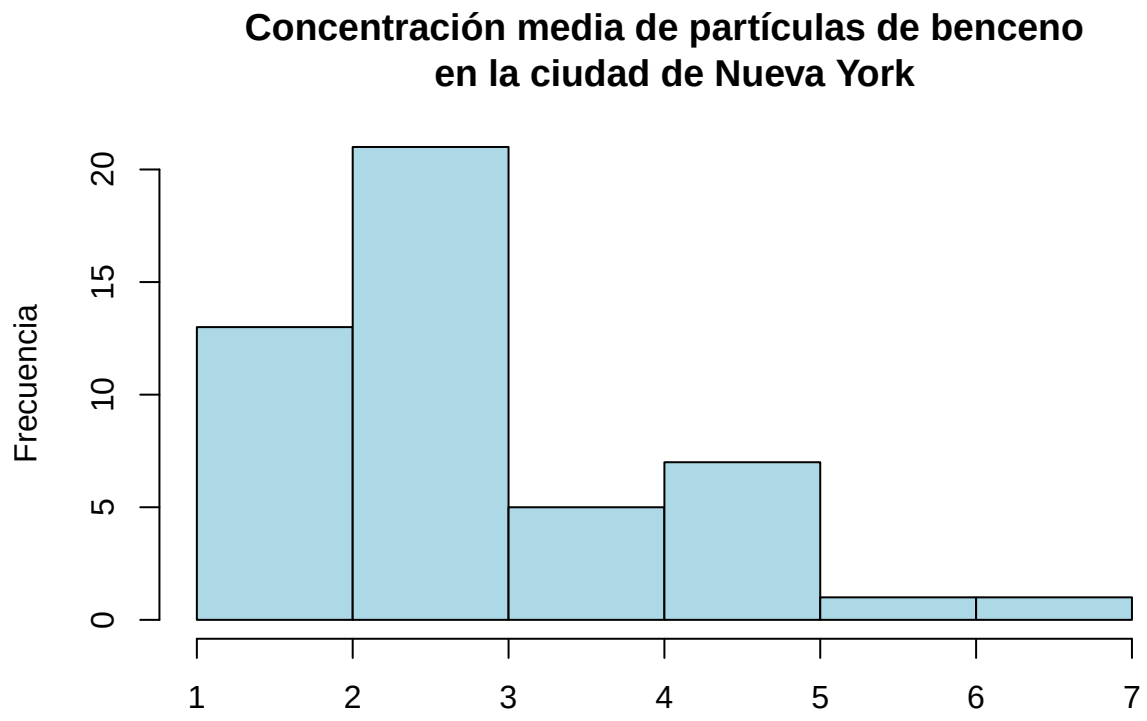

Figura 1: Histograma de la concentración de partículas de benceno en NYC

Otra forma de obtener un histograma es usar el paquete `ggplot2`.

```r
# Si no está el paquete `ggplot2`, lo instalamos
if (!require("ggplot2"))
  install.packages("ggplot2", repos = repository)

# Cargamos el paquete ggplot2
library("ggplot2")
```

```
ggplot() +  aes(data_value) +
  geom_histogram(col = "blue",
                 fill = "lightblue",
                 binwidth = 0.5,
                 alpha = I(0.2)) +
  ggtitle(title_hist) +
    labs(x = xlabel_hist,
         y = ylabel_hist)
```

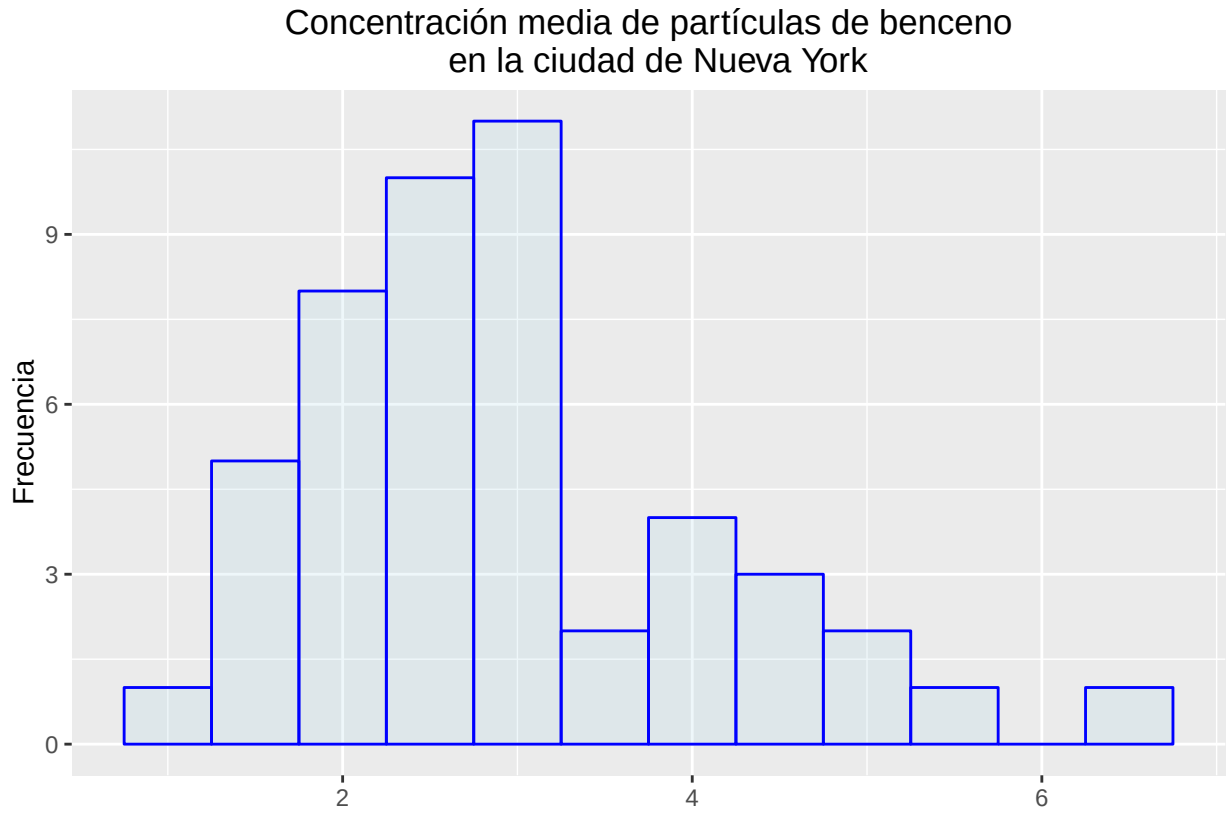

Figura 2: Histograma de la concentración de partículas de benceno en NYC con el paquete `ggplot2`

Observamos que dependiendo el número de grupos, varía el histograma que se muestra.

5. Función de densidad y de probabilidad

Dibujamos la función de densidad y de distribución de la variable `data_value`.

```r
# Dividimos la pantalla en dos
par(mfrow=c(1,2))

# Función de densidad
title_density = "Función de densidad"
ylabel_density = "Densidad"
xlabel_density = ""

plot(density(data_value),
     main = title_density,
     xlab = xlabel_density,
     ylab = ylabel_density)

# Función de probabilidad
title_cdf = "Función de probabilidad"
ylabel_cdf = "Probabilidad acumulada"
xlabel_cdf = ""
plot(ecdf(data_value),
     main = title_cdf,
     xlab = xlabel_cdf,
     ylab = ylabel_cdf)
```

Otra forma de hacerlo, es usando el paquete `ggplot2`.

```r
# Dividimos la pantalla en dos
# Para ello, usamos el paquete `gridExtra`.

# Si no está el paquete `gridExtra`, lo instalamos
if (!require("gridExtra"))
  install.packages("gridExtra", repos = repository)

# Cargamos el paquete `gridExtra`
require(gridExtra)

# Guardamos el plot de la función de densidad en una variable
density = ggplot() + aes(data_value) +
  geom_density(col = "blue",
               fill = "lightblue",
               alpha = 0.2) +
   ggtitle(title_density) +
    labs(x = xlabel_density,
         y = ylabel_density)

# Guardamos el plot de la función de probabilidad en una variable
```

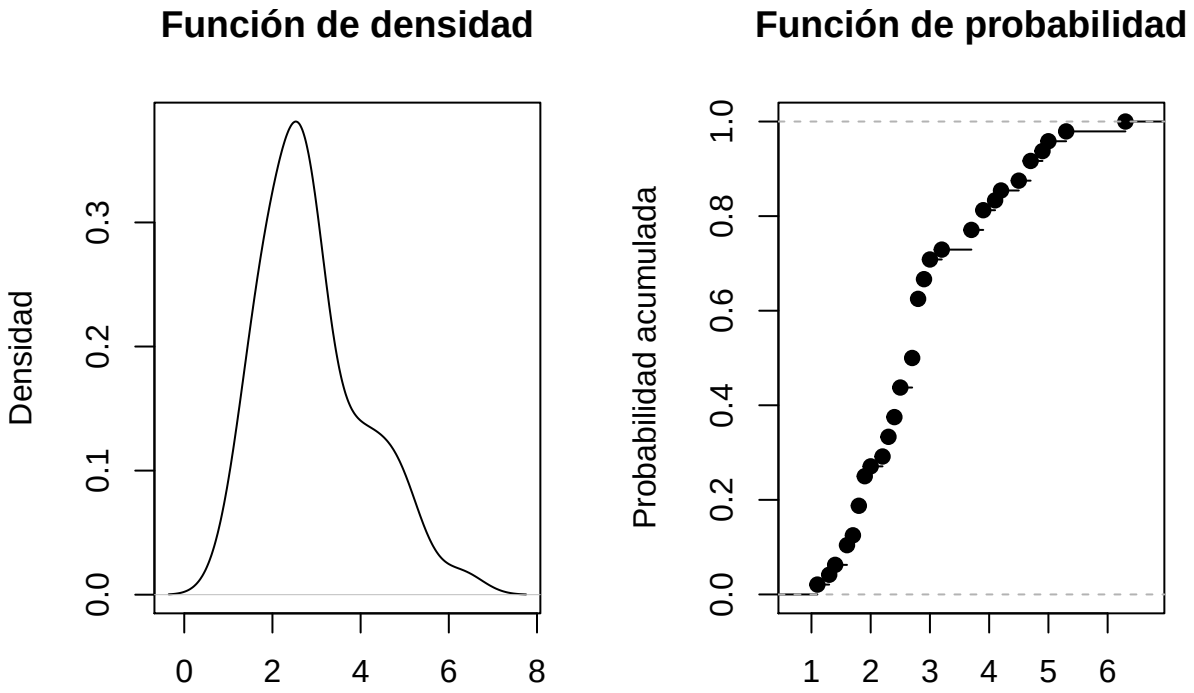

Figura 3: Funciones de densidad y de probabilidad de la concentración de partículas de benceno en NYC

```
probability = ggplot(main = title_density) +
  aes(data_value) + stat_ecdf(col = "blue") +
    ggtitle(title_cdf) +
      labs(x = xlabel_cdf,
           y = ylabel_cdf)


# Mostramos cada plot en dos columnas
grid.arrange(density, probability, ncol=2)
```

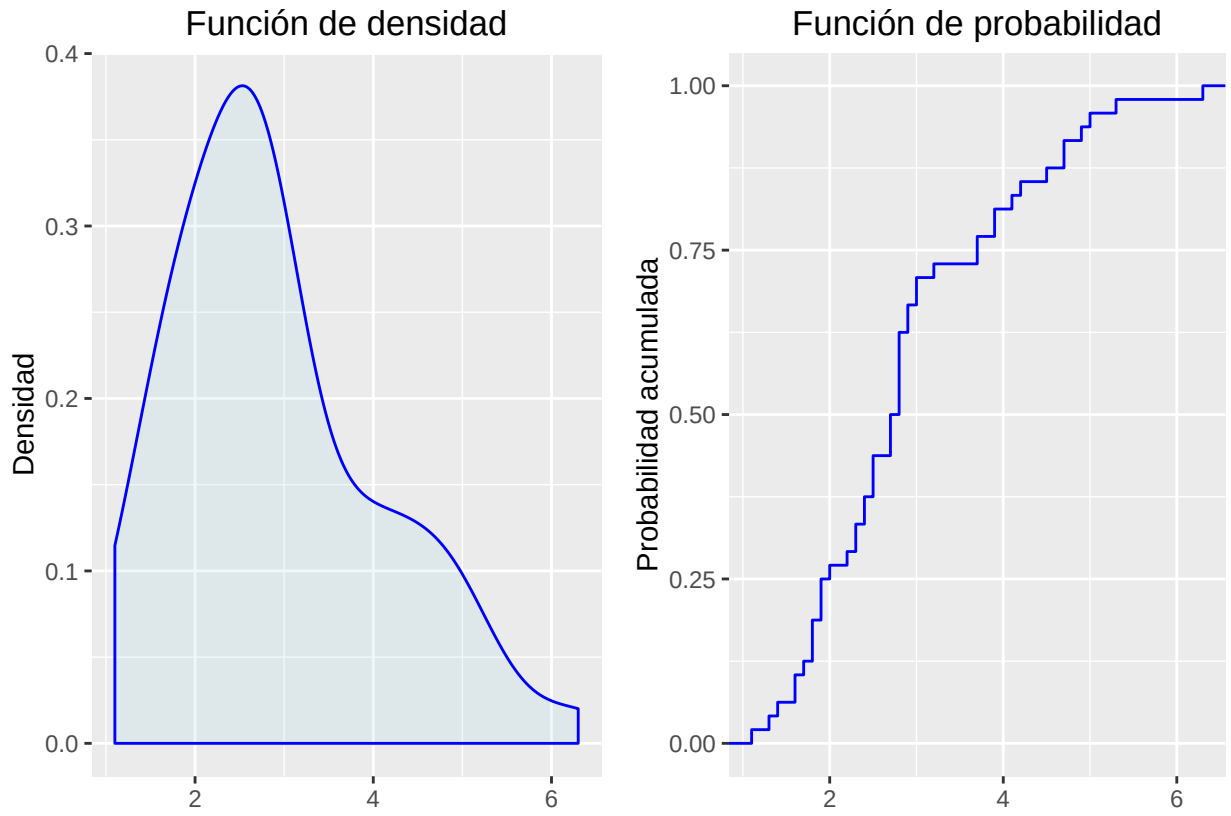

Figura 4: Funciones de densidad y de probabilidad de la concentración de partículas de benceno en NYC con el paquete `ggplot2`

Tanto los estadísticos descriptivos (media, desviación típica, mediana, etc) como los gráficos (histograma y función de densidad) nos dan mucha información de la concentración de benceno en la ciudad de Nueva York.

Sabemos que el mínimo de concentración es 1.10 y el máximo es 6.30. La media y la mediana son muy parecidas (2.91 y 2.80 respectivamente), lo que indica que estas medidas son representativas como estadístico de centralidad de la muestra de benceno.

El rango intercuartílico se sitúa entre 1.75 y 3.70, es decir, que el 50 % (eliminando un 25 % de los valores más bajos y otro 25 % de los más altos) de las veces la concentración de benceno en Nueva York estuvo entre 1.75 y 3.70.

Echando un vistazo a la función de densidad o al histograma, observamos que la moda se sitúa en torno a 2.25. Además, se observa que la densidad crece de forma muy rápida hasta la moda, para reducirse a medida que aumenta la concentración de benceno. En torno a los valores de 3.75 y 4.50, la densidad de la concentración de benceno se mantiene casi constante, con un leve descenso.