Tidy Data con R

José Ignacio Escribano

10 de abril de 2016

Contents

1 Introducción

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

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

```
install.packages(c("dplyr", "tidyr"))
```

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:

```
head(tidy_data)
```

```
## Source: local data frame [6 x 4]
##
##     year geo_entity geo_type data_value
```

```
##   (fctr)      (int)   (fctr)      (dbl)
## 1   2005          1  Borough        2.8
## 2   2005          2  Borough        2.8
## 3   2005          3  Borough        4.7
## 4   2005          4  Borough        1.9
## 5   2005          5  Borough        1.6
## 6   2005          1 Citywide        2.9
```

Comprobamos el tamaño de este nuevo conunto 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.

```r
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.

3 Ejercicio 1

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
```

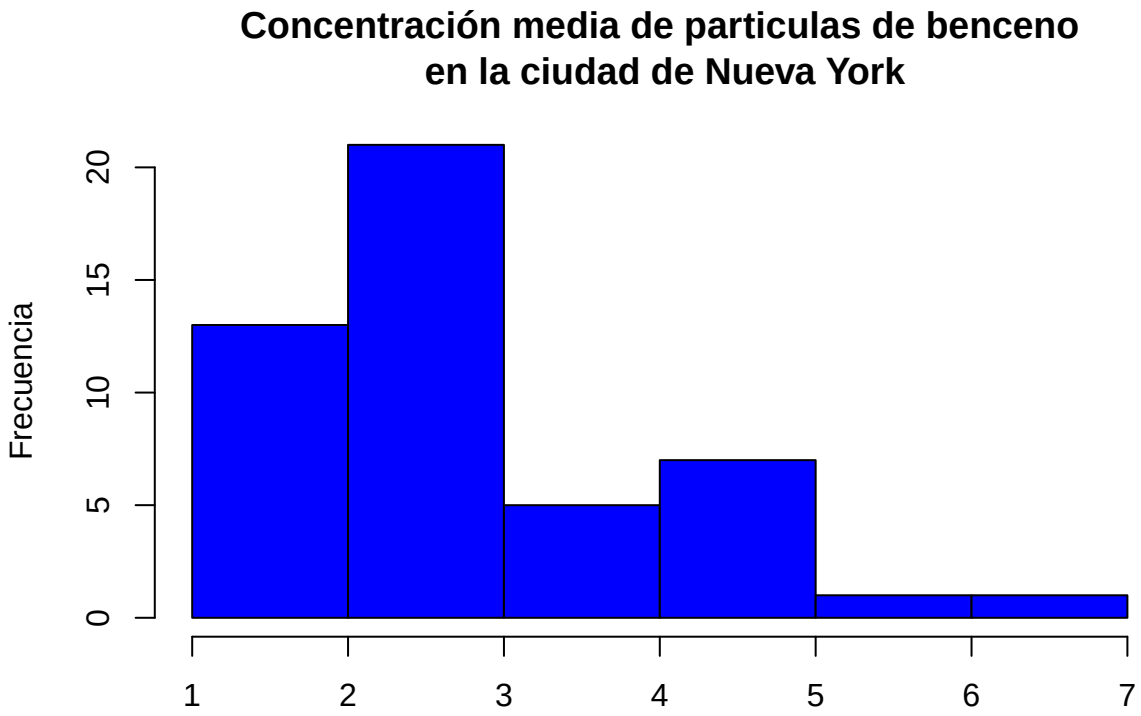
```r
cat("El primer cuartil es", quantile(data_value, 0.25)[[1]],
    "y el tercer cuartil es", quantile(data_value, 0.75)[[1]])
```

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

4 Ejercicio 2

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

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



5 Ejercicio 3

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)
```