
VISUALIZACION DE DATOS EN RSTUDIO

EDUARD LARA

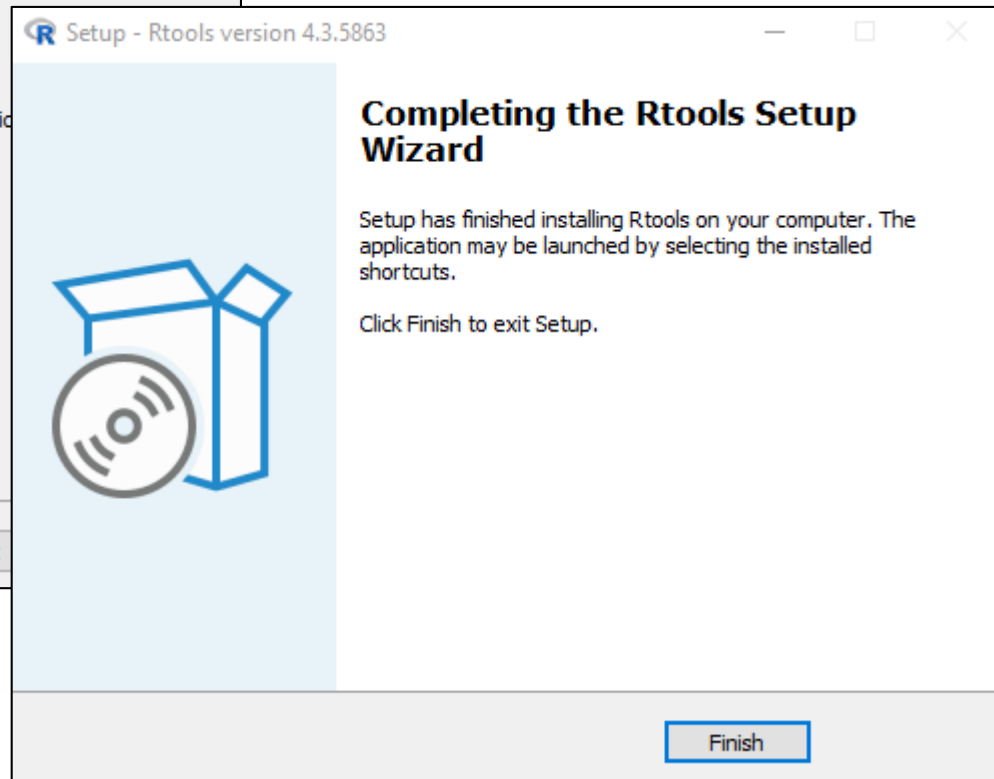
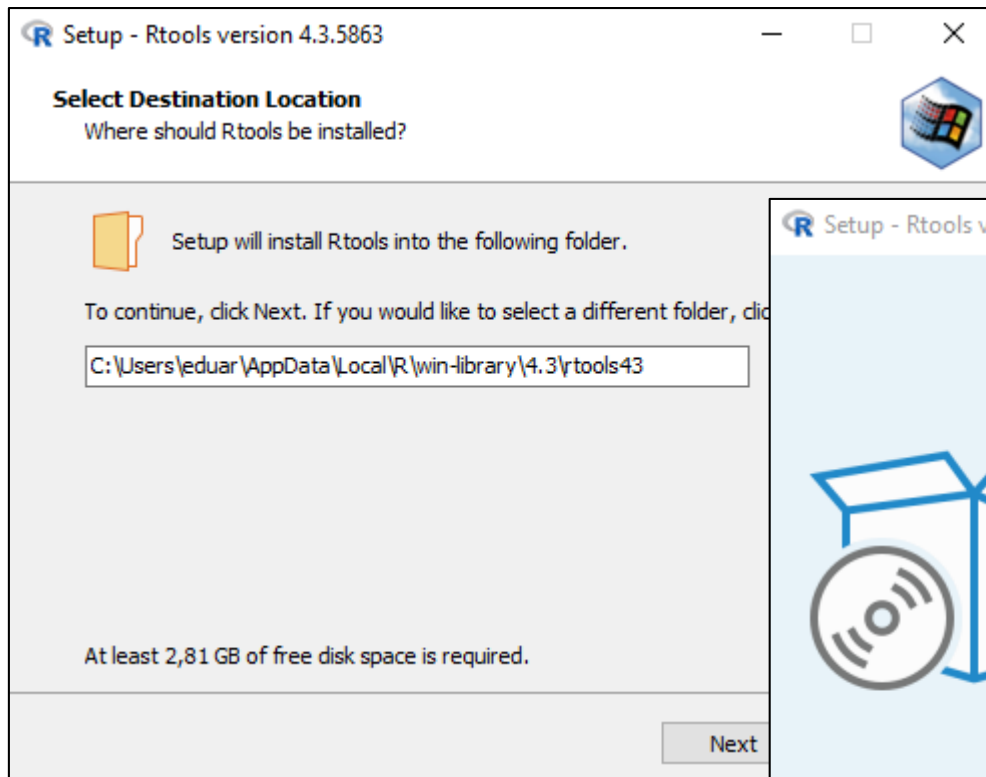
1. INDICE

1. Histogramas
2. ScatterPlots
3. Barplots

1. HISTOGRAMAS

Paso 1. Instalaremos la herramienta Rtools

<https://cran.rstudio.com/bin/windows/Rtools/>



1. HISTOGRAMAS

Paso 2. Los histogramas son gráficos de una única variable con valores continuos en el tiempo. Para explicar los histogramas, usaremos dos paquetes que vamos a instalar:

- `ggplot2`, paquete visualización más importante de datos
- `ggplot2movies`, contiene un data set con información de películas

```
Console Terminal Background Jobs
R 4.3.2 ~./workspace/
> install.packages('ggplot2')
Installing package into 'C:/Users/eduar/AppData/Local/R/win-library/4.3'
(as 'lib' is unspecified)
probando la URL 'https://cran.rstudio.com/bin/windows/contrib/4.3/ggplot2_3.4.4.zip'
Content type 'application/zip' length 4299768 bytes (4.1 MB)
downloaded 4.1 MB

package 'ggplot2' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
  C:\Users\eduar\AppData\Local\Temp\Rtmp80ZQHI\downloaded_packages
> install.packages('ggplot2movies')
Installing package into 'C:/Users/eduar/AppData/Local/R/win-library/4.3'
(as 'lib' is unspecified)
probando la URL 'https://cran.rstudio.com/bin/windows/contrib/4.3/ggplot2movies_0.0.1.zip'
Content type 'application/zip' length 1250961 bytes (1.2 MB)
downloaded 1.2 MB

package 'ggplot2movies' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
  C:\Users\eduar\AppData\Local\Temp\Rtmp80ZQHI\downloaded_packages
> |
```

1. HISTOGRAMAS

Paso 3. Utilizamos la función library para cargar en memoria estos dos paquetes.

```
> library(ggplot2)
want to understand how all the pieces fit together? Read R for Data Science:
https://r4ds.had.co.nz/
> library(ggplot2movies)
> |
```

1. HISTOGRAMAS


Paso 4. Una documentación bastante útil para ggplot2, la encontramos en esta ruta:

https://diegokoz.github.io/intro_ds/fuentes/ggplot2-cheatsheet-2.1-Spanish.pdf

donde aparecen ejemplos de todos los gráficos que podemos hacer

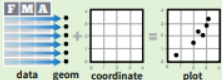
Visualización de Datos usando ggplot2

Guía Rápida

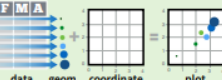


Conceptos Básicos

ggplot2 se basa en la idea que cualquier gráfica se puede construir usando estos tres componentes: **datos**, **coordenadas** y **objetos geométricos (geoms)**. Este concepto se llama: **gramática de las gráficas**.



Para visualizar resultados, asigne variables a las propiedades visuales, o **estéticas**, como **tamaño**, **color** y **posición** x ó y.



Geoms

- Funciones geom se utilizan para visualizar resultados. Asigne variables a las propiedades estéticas del geom. Cada geom forma una capa.

Geométricas Elementales

```
a <- ggplot(economics, aes(date, unemploy))
b <- ggplot(seals, aes(x = long, y = lat))
```

- a + geom_blank()**
(Bueno para expandir límites)
- b + geom_curve()**
`aes(yend = lat + 1, xend=long+1, curvature=z)` - x, xend, y, yend, alpha, angle, color, curvature, linetype, size
- a + geom_path()**
`(lineend="butt", linejoin="round", linemitre=1)` - x, y, alpha, color, group, linetype, size
- a + geom_polygon()**
x, y, alpha, color, fill, group, linetype, size
- b + geom_rect()**
`aes(xmin = long, ymin=lat, xmax=long + 1, ymax = lat + 1)` - xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size
- a + geom_ribbon()**
`aes(ymin=unemploy - 900, ymax=unemploy + 900)` - x, ymax, ymin, alpha, color, fill, group, linetype, size

Segmentos Lineales

```
b + geom_abline(aes(intercept=0, slope=1))
b + geom_hline(aes(yintercept = lat))
```

Dos Variables

X Continua, Y Continua

```
e <- ggplot(mpg, aes(cty, hwy))
```

- e + geom_label()**
`aes(label = cty)`, `nudge_x = 1`, `nudge_y = 1`, `check_overlap = TRUE` - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust
- e + geom_jitter()**
`(height = 2, width = 2)` - x, y, alpha, color, fill, shape, size
- e + geom_point()**
x, y, alpha, color, fill, shape, size, stroke
- e + geom_quantile()**
x, y, alpha, color, group, linetype, size, weight
- e + geom_rug()**
`(sides = "bl")` - x, y, alpha, color, linetype, size
- e + geom_smooth()**
`(method = lm)` - x, y, alpha, color, fill, group, linetype, size, weight
- e + geom_text()**
`aes(label = cty)`, `nudge_x = 1`, `nudge_y = 1`, `check_overlap = TRUE` - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

Distribución Bivariada Continua

```
h <- ggplot(diamonds, aes(carat, price))
```

- h + geom_bin2d()**
x, y, alpha, color, fill, linetype, size, weight
- h + geom_density2d()**
x, y, alpha, colour, group, linetype, size
- h + geom_hex()**
x, y, alpha, colour, fill, size

Función Continua

```
i <- ggplot(economics, aes(date, unemploy))
```

- i + geom_area()**
x, y, alpha, color, fill, linetype, size
- i + geom_line()**
x, y, alpha, color, group, linetype, size
- i + geom_step()**
`(direction = "hv")` - x, y, alpha, color, group, linetype, size

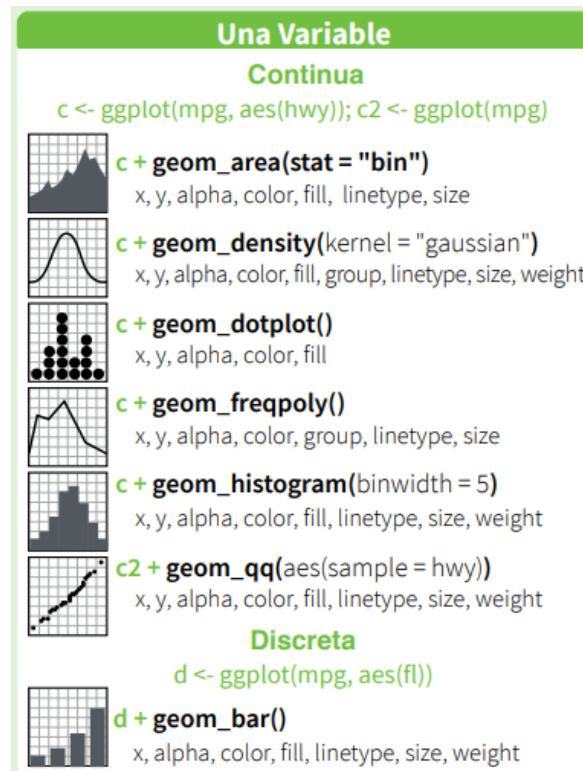
Visualizando el Error

```
df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
i <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))
```

X Discreta, Y Continua

1. HISTOGRAMAS

Paso 5. Vamos a utilizar el gráfico `geom_histogram`, que necesita de una única variable, continua en el tiempo, y el diagrama de barras es un histograma. Podemos utilizar todos estos atributos para configurar nuestro gráfico.



1. HISTOGRAMAS

Paso 6. En Rstudio, primero creamos una variable películas que vamos a crear con los datos del data set movies que viene en el paquete ggplot2movies

Si hacemos un head de películas para ver las primeras líneas de este dataset, aquí tenemos el título, el año, la longitud, el rating, etc. Hay 9 variables más que están abajo

```
> películas=movies
> head(películas)
# A tibble: 6 x 24
  title    year length budget rating votes   r1    r2    r3    r4    r5    r6    r7    r8    r9
  <chr>   <int>   <int>   <int>   <dbl> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1 $      1971    121    NA     6.4   348   4.5   4.5   4.5   4.5  14.5  24.5  24.5  14.5  4.5
2 $100... 1939     71    NA     6     20    0   14.5   4.5  24.5  14.5  14.5  14.5   4.5  4.5
3 $21 ... 1941      7    NA     8.2    5    0    0    0    0   24.5    0  44.5  24.5
4 $40,... 1996     70    NA     8.2    6  14.5    0    0    0    0    0    0    34.5
5 $50,... 1975     71    NA     3.4   17  24.5   4.5    0  14.5  14.5   4.5    0    0
6 $spent 2000     91    NA     4.3   45   4.5   4.5   4.5  14.5  14.5  14.5   4.5   4.5  14.5
# i 9 more variables: r10 <dbl>, mpaa <chr>, Action <int>, Animation <int>, Comedy <int>,
#   Drama <int>, Documentary <int>, Romance <int>, Short <int>
> |
```


1. HISTOGRAMAS

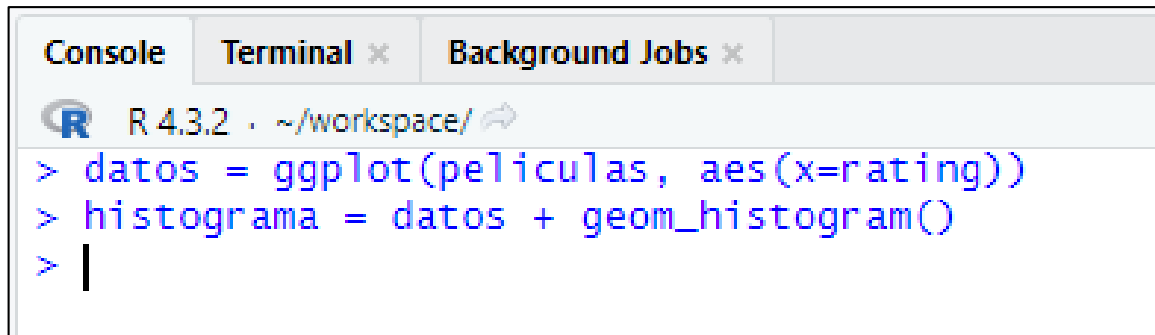
Paso 7. Para ver mejor el título, se puede hacer una selección de algunas columnas: título, año y rating. Así se ve mejor el título de las películas, el año y la puntuación

```
Console Terminal x Background Jobs x
R 4.3.2 · ~/workspace/ ↗
> peliculas[c('title','year','rating')]
# A tibble: 58,788 × 3
  title                                year rating
  <chr>                                <int> <dbl>
1 $                                     1971    6.4
2 $1000 a Touchdown                   1939     6
3 $21 a Day Once a Month              1941    8.2
4 $40,000                             1996    8.2
5 $50,000 climax Show, The            1975    3.4
6 $pent                               2000    4.3
7 $windle                             2002    5.3
8 '15'                                2002    6.7
9 '38'                                1987    6.6
10 '49-'17                             1917     6
# i 58,778 more rows
# i Use `print(n = ...)` to see more rows
> |
```

1. HISTOGRAMAS

Paso 8. Para crear el histograma, creamos una variable datos que van a ser los datos que vamos a poner en el histograma. Mediante ggplot pasaremos el dataset películas e indicamos que el valor en el eje X va a ser la columna rating.

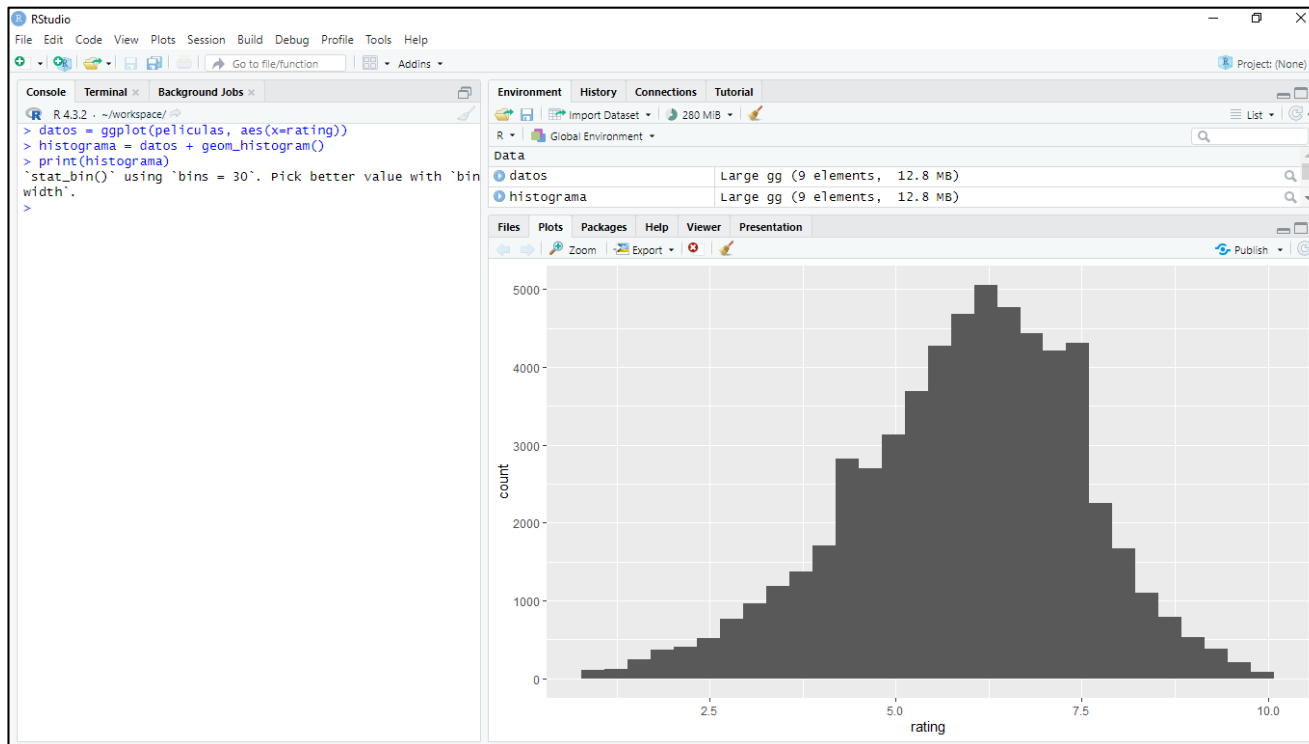
Creamos otra variable histograma que sean estos datos más geom_histogram(), para crear un histograma con una variable continua, sin ningún parámetro



```
Console Terminal x Background Jobs x
R 4.3.2 · ~/workspace/ ↗
> datos = ggplot(películas, aes(x=rating))
> histograma = datos + geom_histogram()
> |
```

1. HISTOGRAMAS

Paso 9. Hacemos **print(histograma)**, para mostrar el gráfico. Inmediatamente se abre la pestaña pilots, donde aparece el histograma de la variable rating. Aparece el n° de veces que se repite cada uno de los valores de la variable.

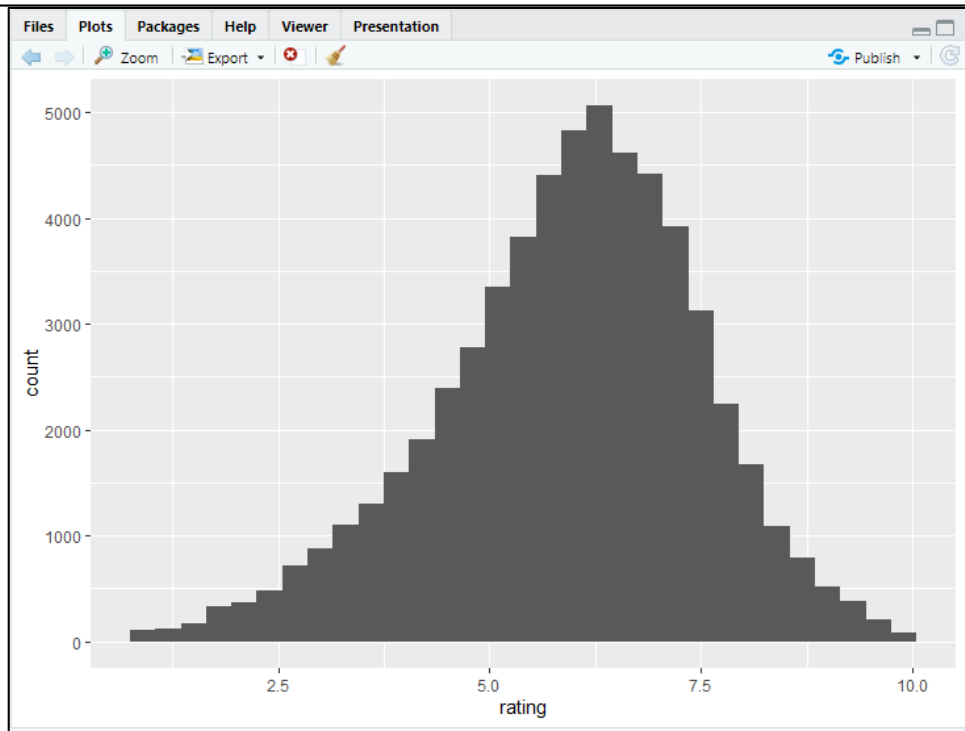


El rating 5 se repetiría unas 3 mil y pico veces.

1. HISTOGRAMAS

Paso 10. Podemos configurar el histograma para que tenga otra configuración. Por ejemplo podemos hacer que las columnas sean un poco más estrechas que la anterior a 0.3.

```
> histograma = datos + geom_histogram(binwidth = 0.3)
> print(histograma)
> |
```

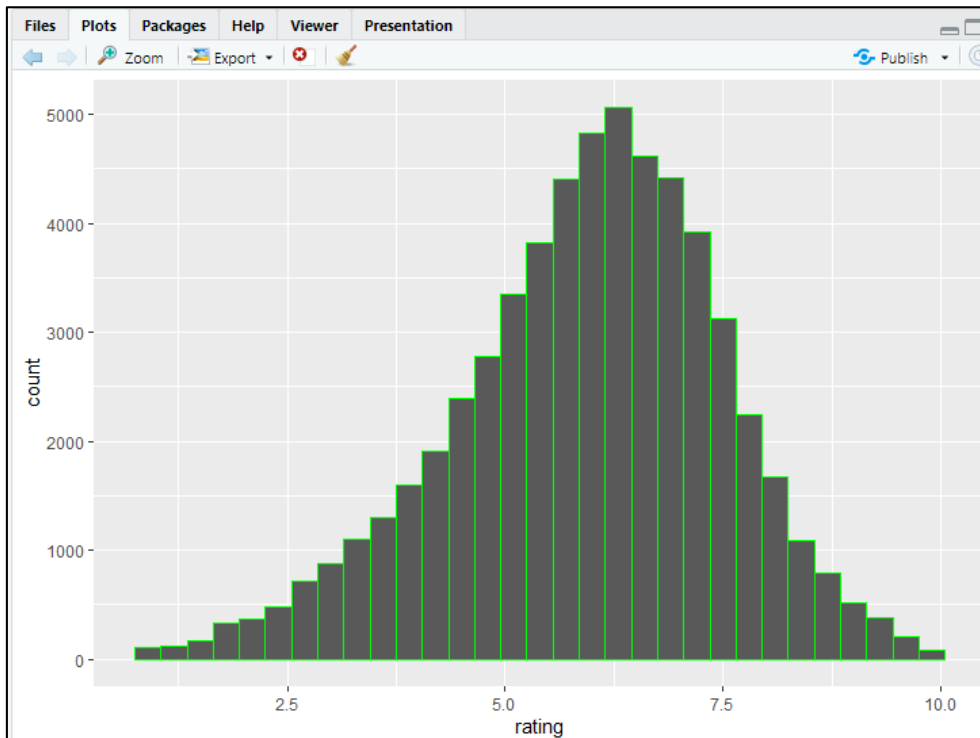


Con el botón flecha debajo de Files, podemos ir al gráfico anterior y volver al nuevo para comprobar que se ha estrechado el ancho de estas columnas

1. HISTOGRAMAS

Paso 11. También podemos cambiar el color, por ejemplo de color verde.

```
> histograma = datos + geom_histogram(binwidth = 0.3, color='green')  
> print(histograma)  
>
```

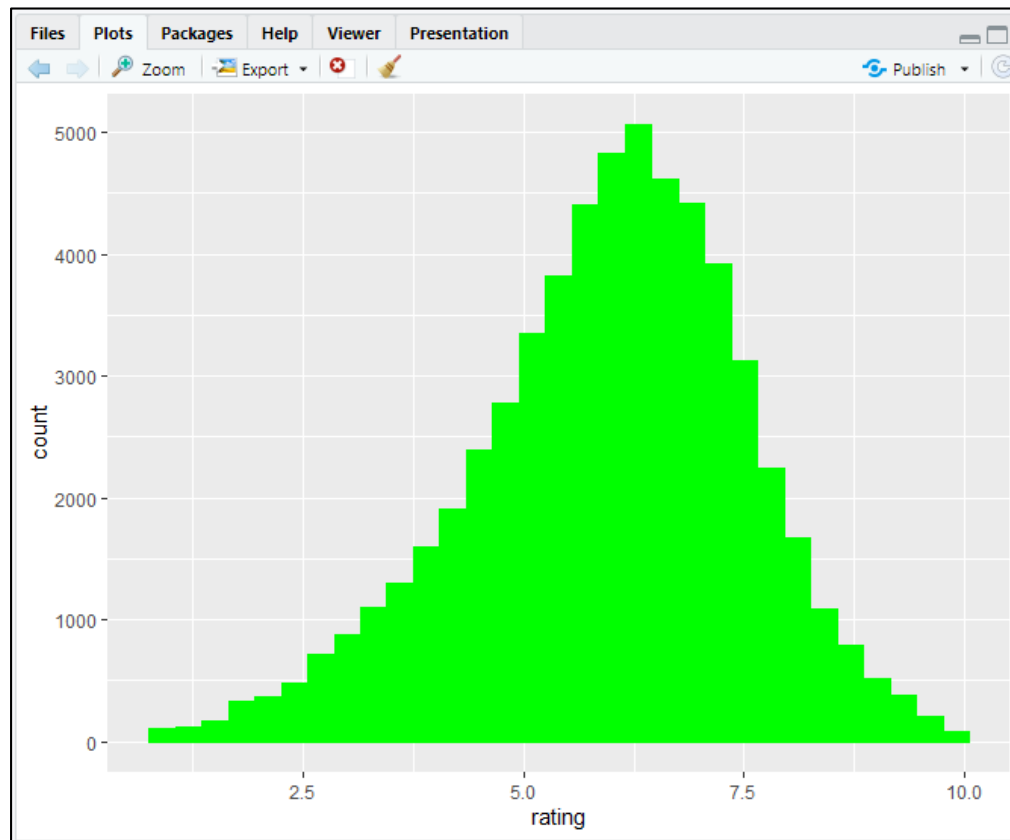


Aparece el contorno de color verde y el interior lo ha dejado en el color que ya está.

1. HISTOGRAMAS

Paso 12. Podemos cambiar el color del interior con fill.
Ponemos tanto el contorno como el relleno del gráfico verde

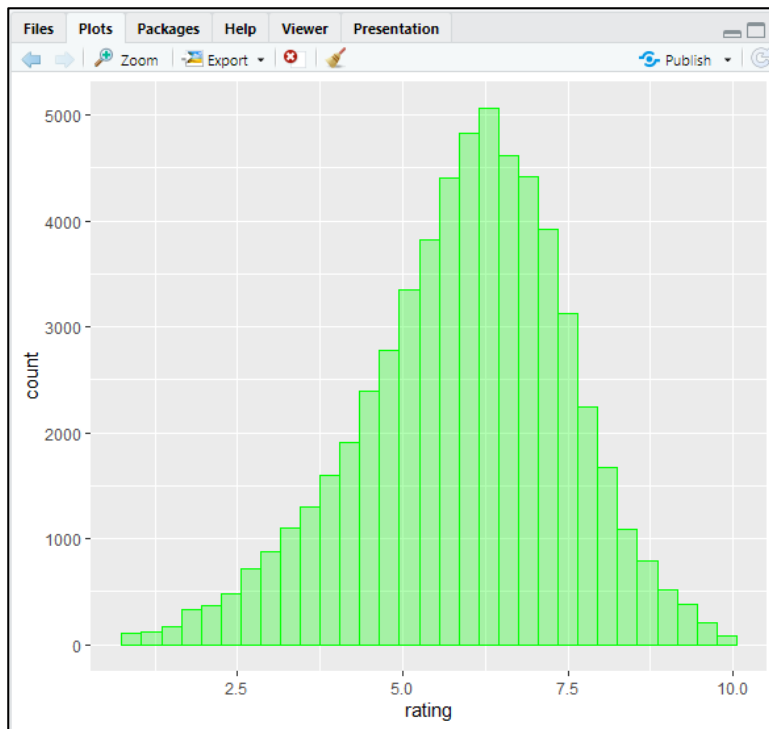
```
> histograma = datos + geom_histogram(binwidth = 0.3, color='green',fill='green')  
> print(histograma)  
> |
```



1. HISTOGRAMAS

Paso 13. Las líneas horizontales no se ven por detrás del gráfico. Podemos hacerlo un poco transparente, con el atributo `alpha`, para que se vean las líneas que pasan por detrás.

```
> histograma = datos + geom_histogram(binwidth = 0.3, color='green', fill='green', alpha=0.3)
> print(histograma)
> |
```

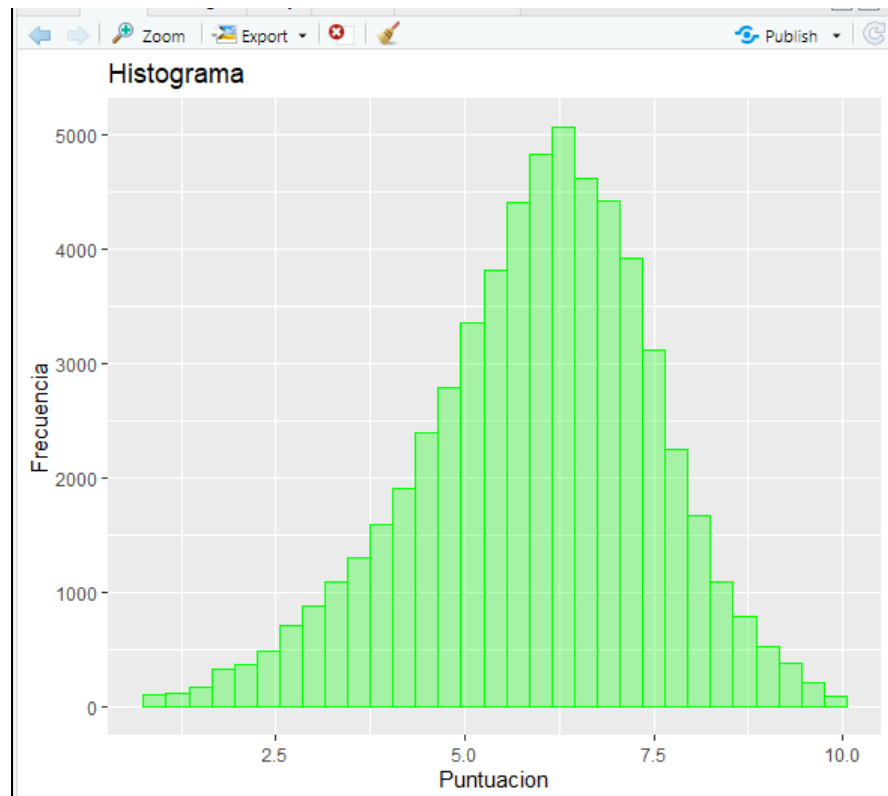


Vemos que se ha suavizado un poco el color del relleno para que se vean las líneas con los números de la frecuencia

1. HISTOGRAMAS

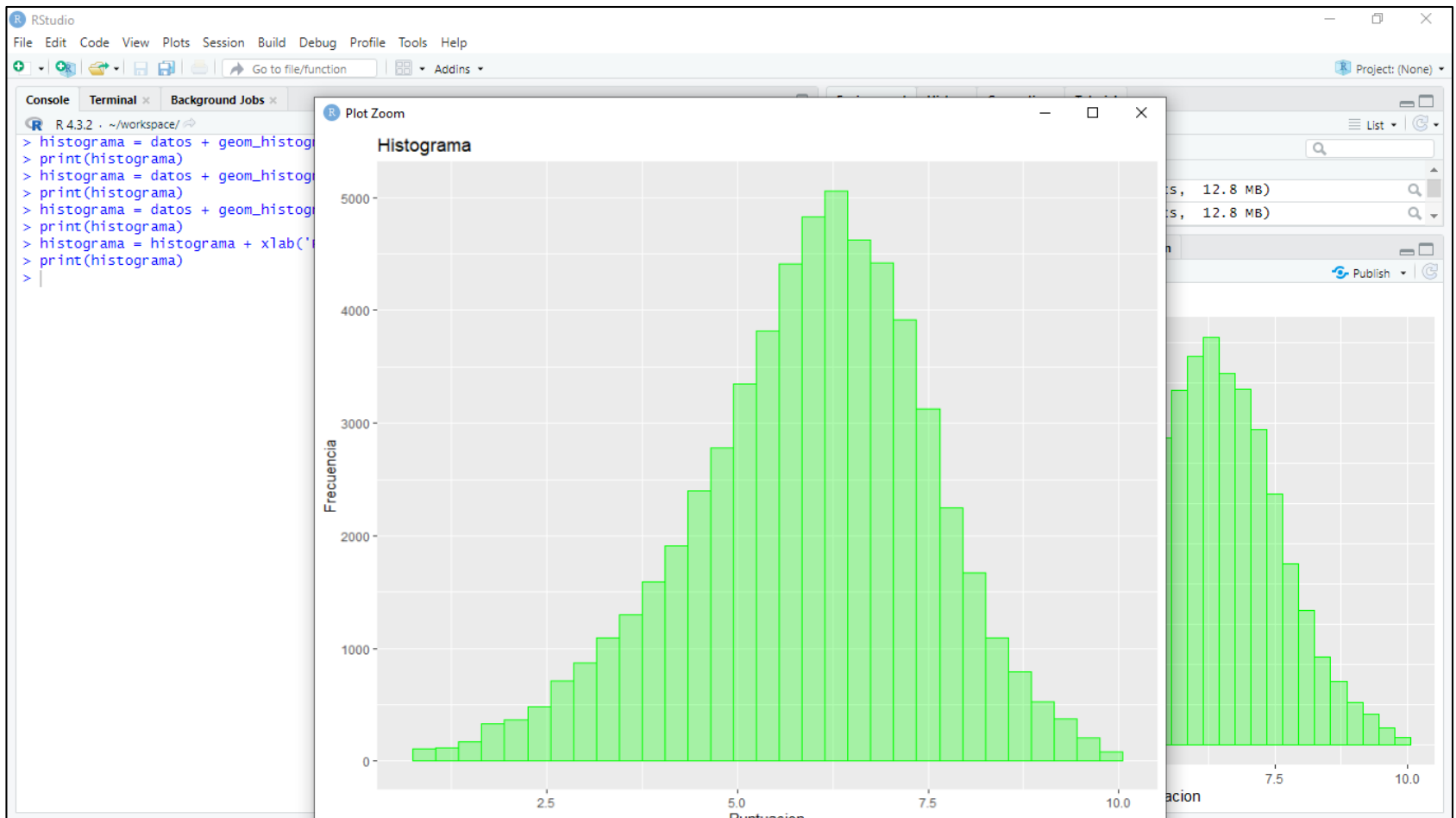
Paso 14. También podemos cambiarle el nombre del eje de las X, el eje de las y ponerle un título al gráfico.

```
> histograma = histograma + xlab('Puntuacion') + ylab('Frecuencia') + ggtitle('Histograma')  
> print(histograma)  
>
```



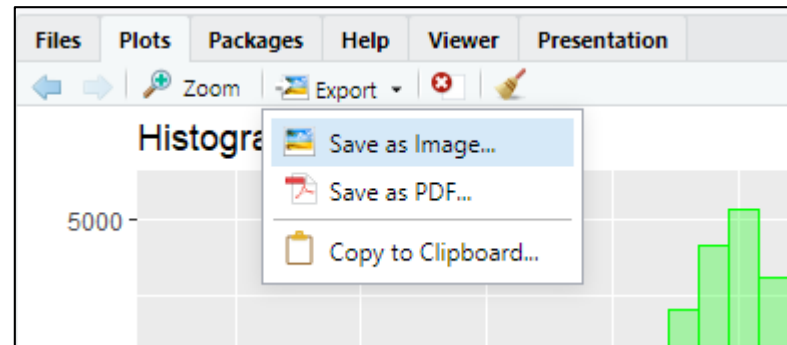
1. HISTOGRAMAS

Paso 15. Si le damos al botón de zoom, se abre una pantalla grande donde podemos ver mejor el gráfico



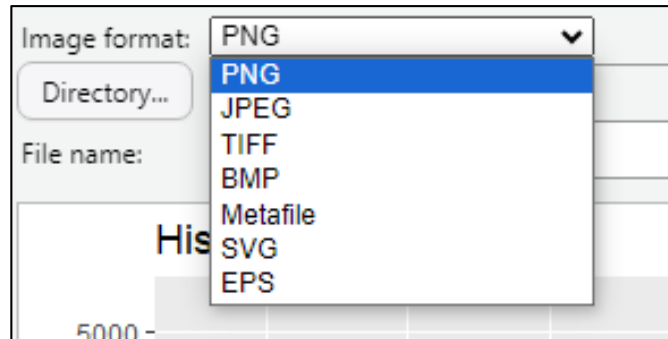
1. HISTOGRAMAS

Paso 16. Mediante el botón export podemos exportar el gráfico como su propia imagen, como si fuera un PDF o al portapapeles para pegarlo en otra aplicación

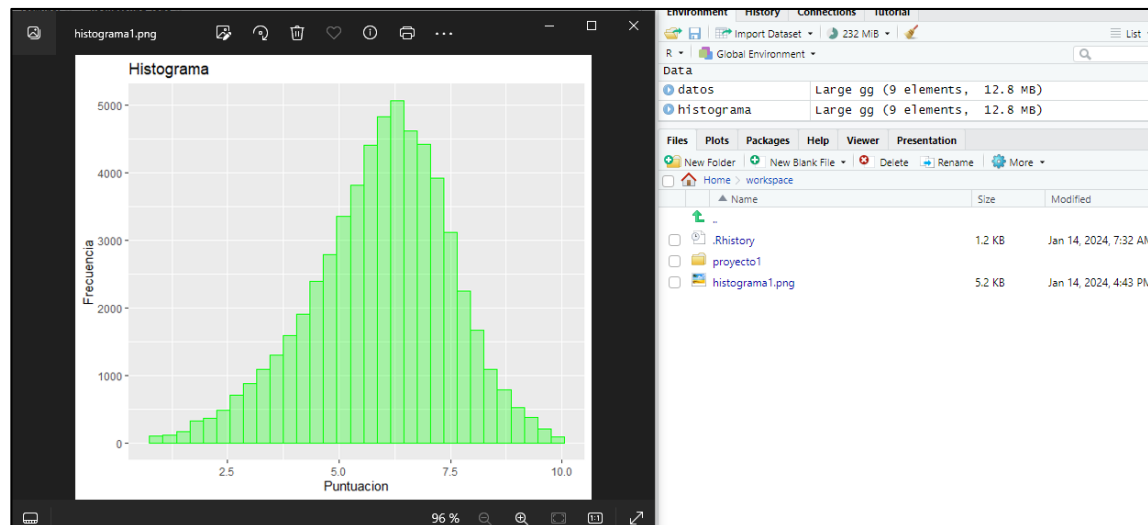


1. HISTOGRAMAS

Paso 17. Si seleccionamos exportar como imagen, lo guardaremos dentro de nuestro workspace en diferentes formatos

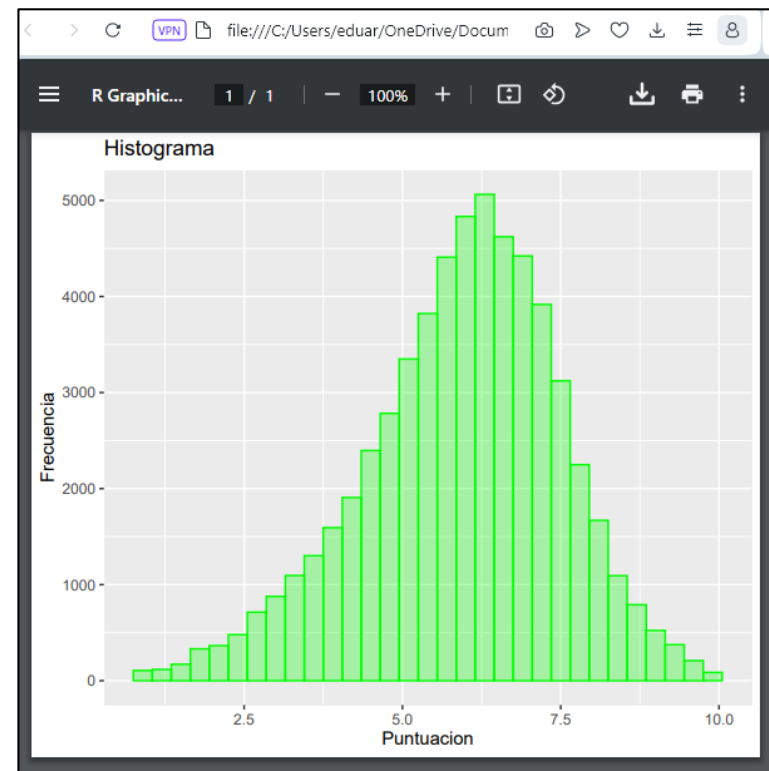
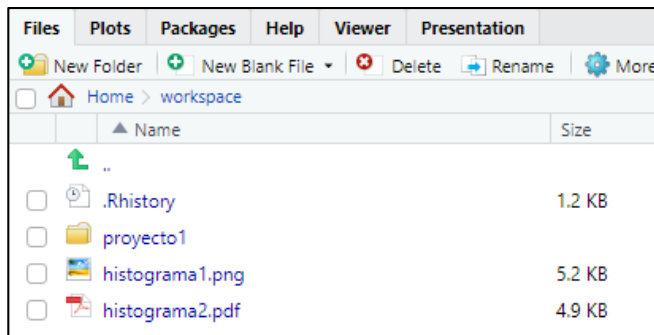
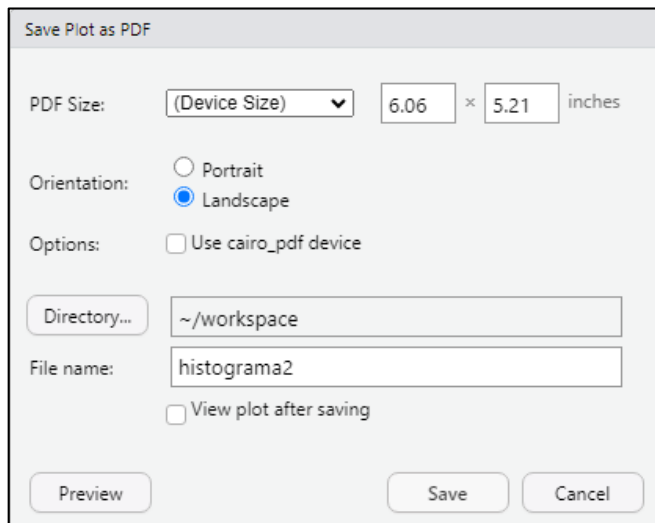


Si lo pulsamos en Files nos abrirá el fichero



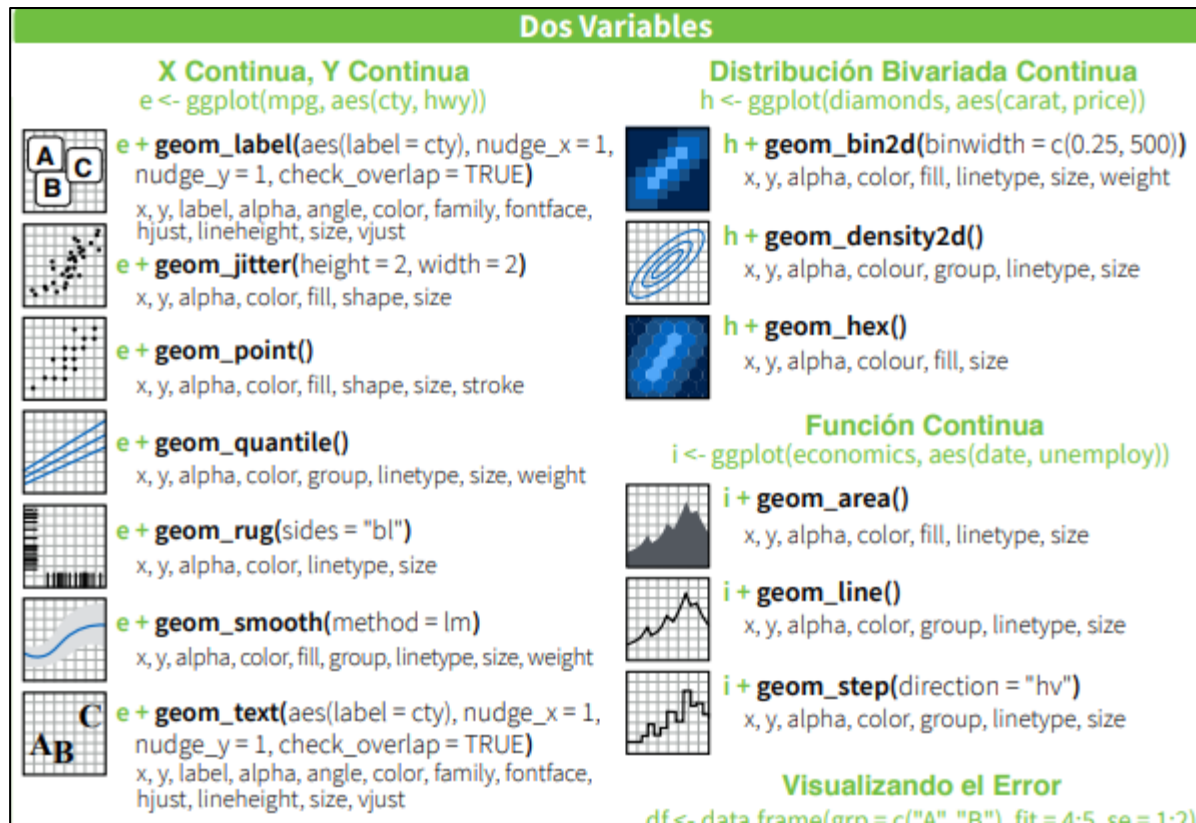
1. HISTOGRAMAS

Paso 18. También lo podemos exportar como PDF. Si vamos a Files nos aparece como fichero pdf. Si hacemos click nos abrirá el PDF en el navegador



2. SCATTERPLOTS

Paso 1. Los Scatterplots son gráficos con dos variables continuas. Si vamos a la pagina de ayuda del paquete ggplot2, seleccionaremos scatterplot mediante la función `geom_point()`



2. SCATTERPLOTS

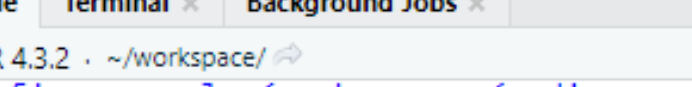
Paso 2. Primero cargamos ggplot2 con library. Despues creamos una variable coches que va a contener la información del mtcars que viene como ejemplo en Rstudio
Contiene información de los coches en las diferentes columnas del dataset

```
> library(ggplot2)
> coches = mtcars
> head(coches)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

2. SCATTERPLOTS

Paso 3. Entonces vamos a crear un scatterplot, donde pondremos en el eje X la columna disp y en el eje y la columna mpg. Añadiremos la función `geom_point` e imprimimos el gráfico.

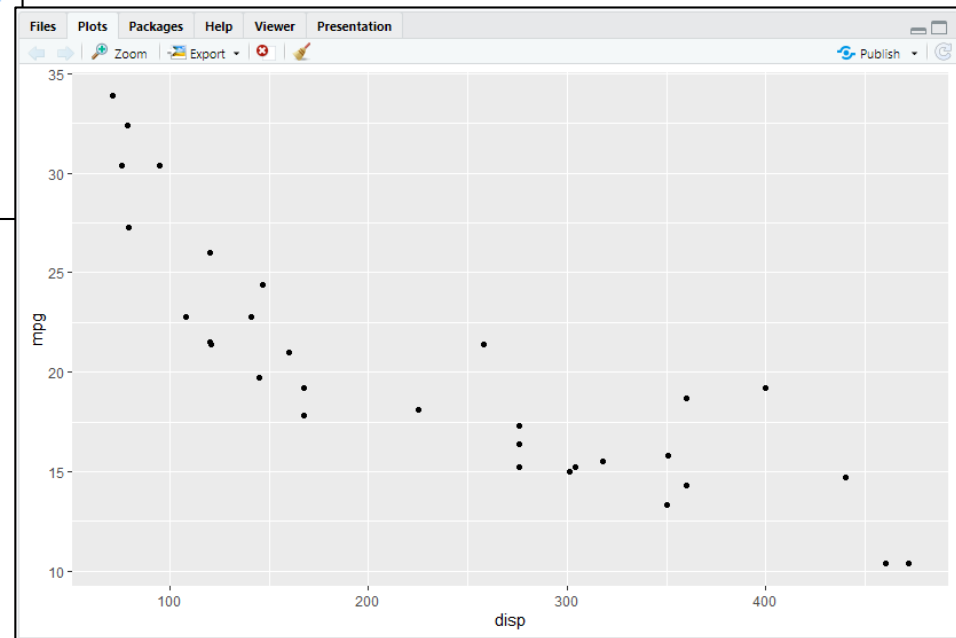


Console Terminal × Background Jobs ×

R 4.3.2 · ~/workspace/ ↗

```
> grafico = ggplot(coches, aes(x=disp, y=mpg))  
>  
> grafico = grafico + geom_point()  
>  
> print(grafico)  
>
```

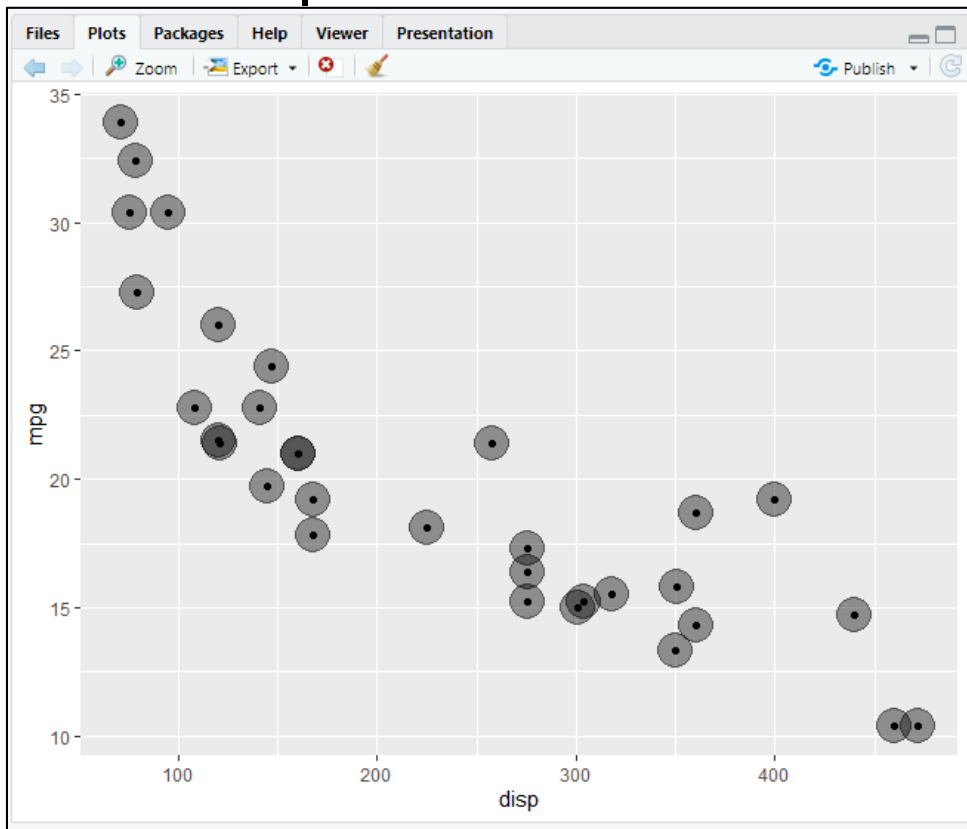
Mediante un scatterplot hemos relacionado las dos variables, la columna disp, y la columna mpg (eje y)



2. SCATTERPLOTS

Paso 4. Podemos cambiar el tamaño de los puntos. Ponemos un tamaño de 8, y una transparencia del 40%.

```
> grafico = grafico + geom_point(size=8, alpha=0.4)
> print(grafico)
> |
```

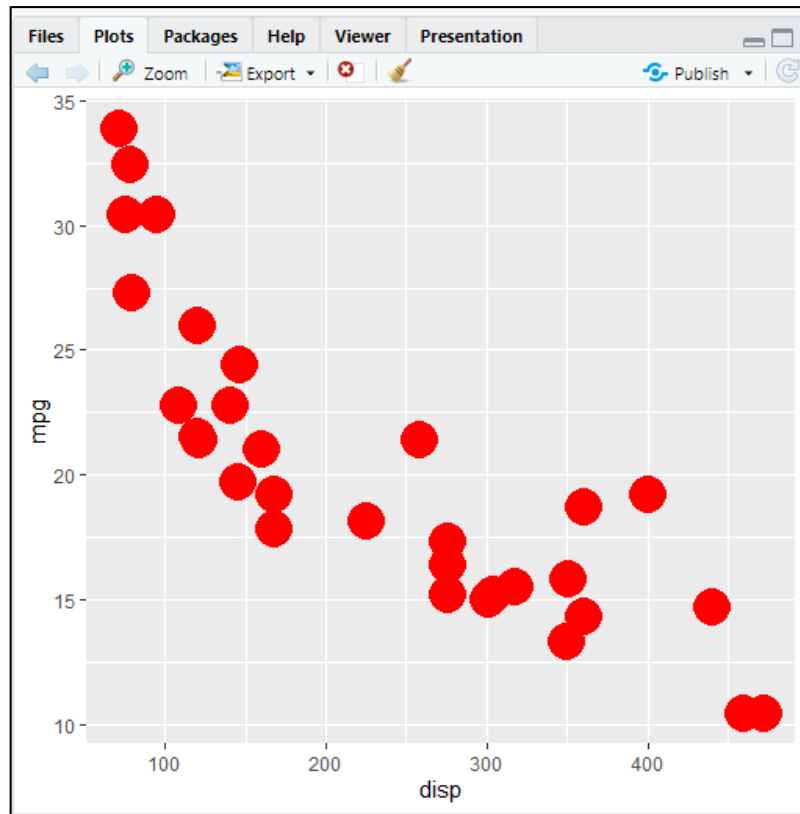


Ha aumentado el círculo al tamaño de 8 y ha hecho una transparencia, donde se ve más clarito el gris. Así permite ver dónde se junta con otros puntos.

2. SCATTERPLOTS

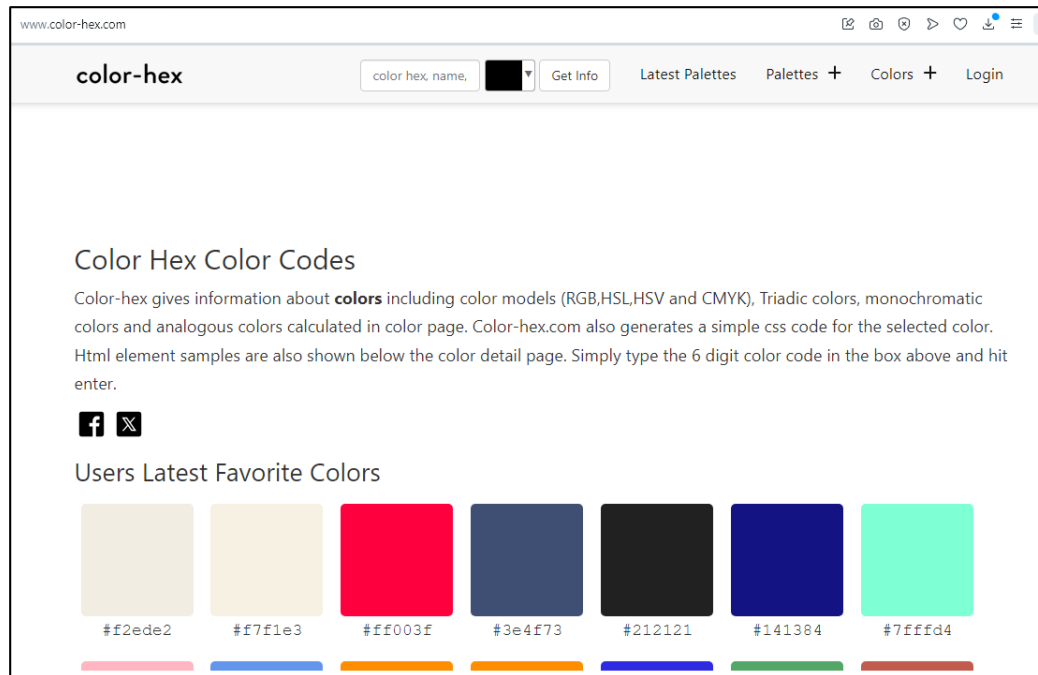
Paso 5. Ahora vamos a cambiar el color. En vez de gris le ponemos el color rojo.

```
> grafico = grafico + geom_point(size=8, color='red')  
> print(grafico)  
> |
```



2. SCATTERPLOTS

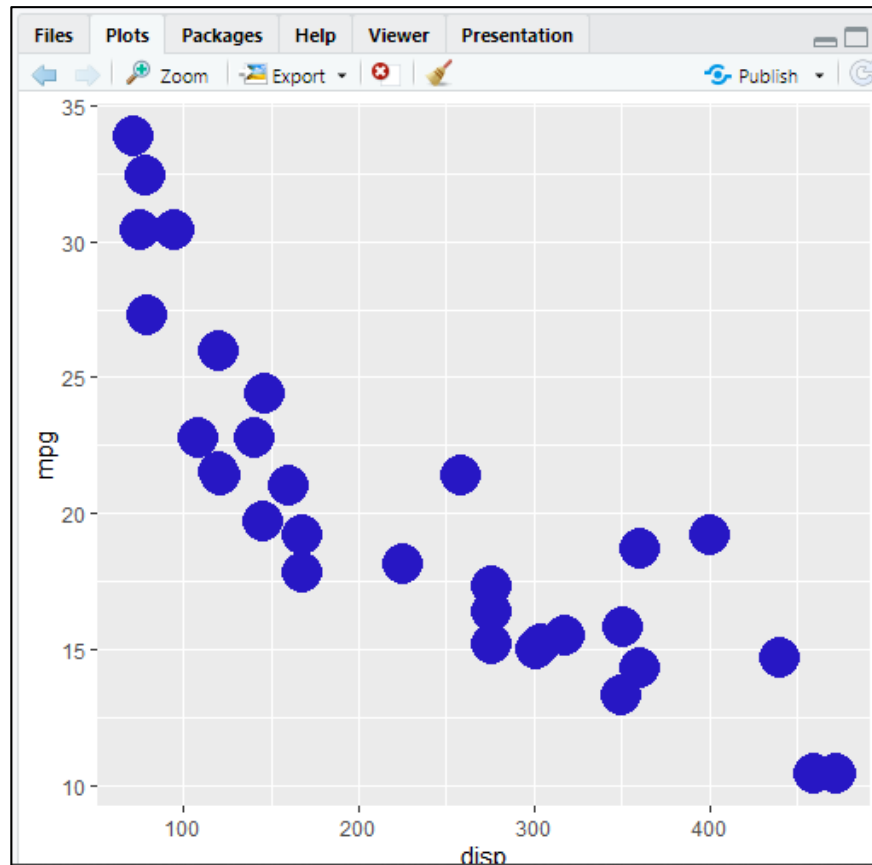
Paso 6. Existe una página en internet www.color-hex.com donde podemos seleccionar diferentes tipos de colores. Cada color que podemos elegir lleva una almohadilla seguida de un código hexadecimal, que lo identifica. Los utilizaremos para crear nuestro gráfico, nuestros puntos del gráfico con ese color.



2. SCATTERPLOTS

Paso 7. Configuramos un color con su código hexadecimal, en el gráfico

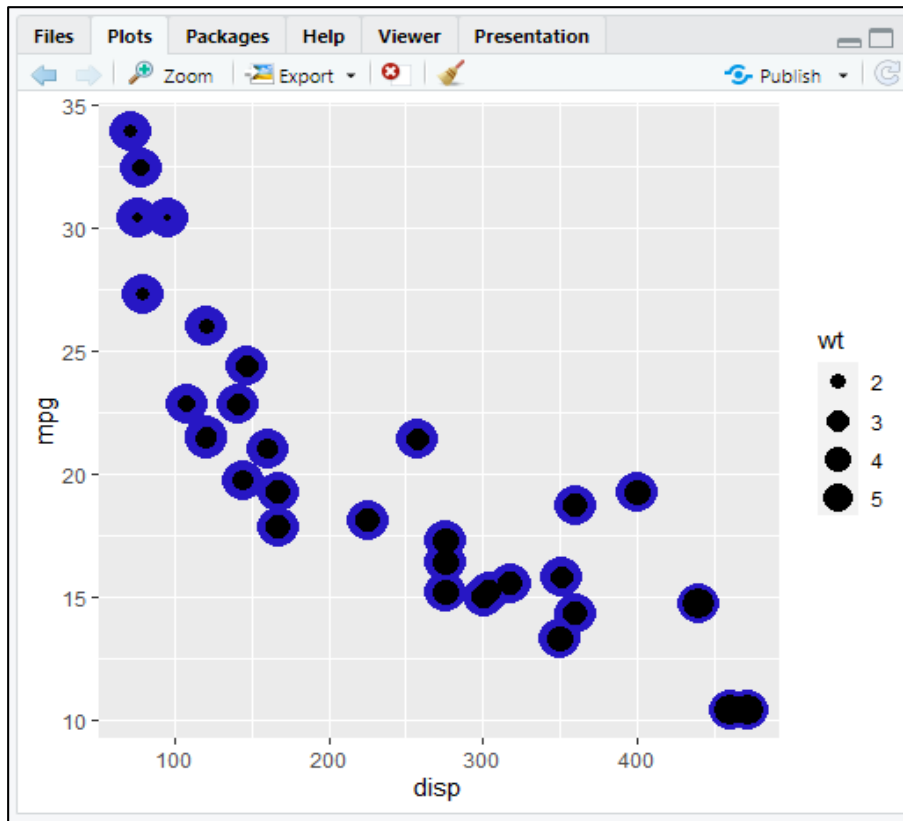
```
> grafico = grafico + geom_point(size=8, color='#2717c4')  
> print(grafico)  
>
```



2. SCATTERPLOTS

Paso 8. Podemos configurar también el tamaño del punto, y ponerlo en función del valor de otra columna. .

```
> grafico = grafico + geom_point(aes(size=wt))  
> print(grafico)  
> |
```



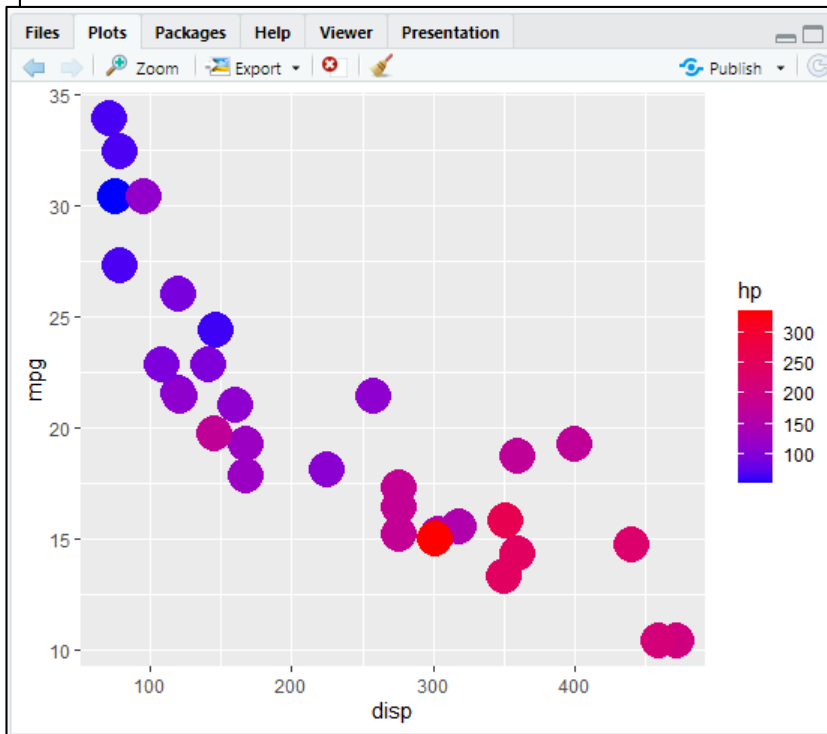
Va cambiando el tamaño del punto en función del valor de otra columna.

Es otra forma de hacerlo también para relacionar, en este caso tres columnas.

2. SCATTERPLOTS

Paso 9. Si queremos hacer un tipo de gráfico, como una especie de gradiente de colores

```
> grafico = ggplot(coches, aes(x=disp,y=mpg))  
> grafico = grafico + geom_point(size=8, aes(color=hp))  
> grafico = grafico + scale_color_gradient(low='blue',high='red')  
> print(grafico)  
>
```



De esta forma podemos combinar tres variables con una gráfica de colores gradientes en función de la columna HP. Comparamos tres variables de una manera rápida y sencilla.







3. BARPLOTS

Paso 1. Un barplot es un gráfico con una única variable discreta. Si vamos a la ayuda sobre el paquete de ggplot2, elegiremos la función `geom_bar()` que genera un diagrama de barras, donde el eje de las X pondremos una variable de tipo discreto con valores alfanuméricos etc

Una Variable


Continua

`c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)`

-  `c + geom_area(stat = "bin")`
x, y, alpha, color, fill, linetype, size
-  `c + geom_density(kernel = "gaussian")`
x, y, alpha, color, fill, group, linetype, size, weight
-  `c + geom_dotplot()`
x, y, alpha, color, fill
-  `c + geom_freqpoly()`
x, y, alpha, color, group, linetype, size
-  `c + geom_histogram(binwidth = 5)`
x, y, alpha, color, fill, linetype, size, weight
-  `c2 + geom_qq(aes(sample = hwy))`
x, y, alpha, color, fill, linetype, size, weight

Discreta

`d <- ggplot(mpg, aes(fl))`

-  `d + geom_bar()`
x, alpha, color, fill, linetype, size, weight

3. BARPLOTS

Paso 2. Cargamos la librería ggplot2 y crearemos una variable datos que contendrá la dataset mpg. Si visualizamos las primeras líneas de mpg veremos que tiene diferentes columnas a nivel de coches, el constructor, el modelo, el año, etc.

```
> library(ggplot2)
> datos = mpg
> head(datos)
# A tibble: 6 x 11
  manufacturer model displ  year   cyl trans      drv    cty   hwy fl      class
  <chr>         <chr> <dbl> <int> <int> <chr>    <chr> <int> <int> <chr> <chr>
1 audi         a4      1.8  1999     4 auto(l5) f       18    29 p      compact
2 audi         a4      1.8  1999     4 manual(m5) f       21    29 p      compact
3 audi         a4      2    2008     4 manual(m6) f       20    31 p      compact
4 audi         a4      2    2008     4 auto(av) f       21    30 p      compact
5 audi         a4      2.8  1999     6 auto(l5) f       16    26 p      compact
6 audi         a4      2.8  1999     6 manual(m5) f       18    26 p      compact
```

3. BARPLOTS

Paso 3. Si queremos ver la estructura de la base de datos, vemos que tiene 234 ocurrencias, con 11 columnas

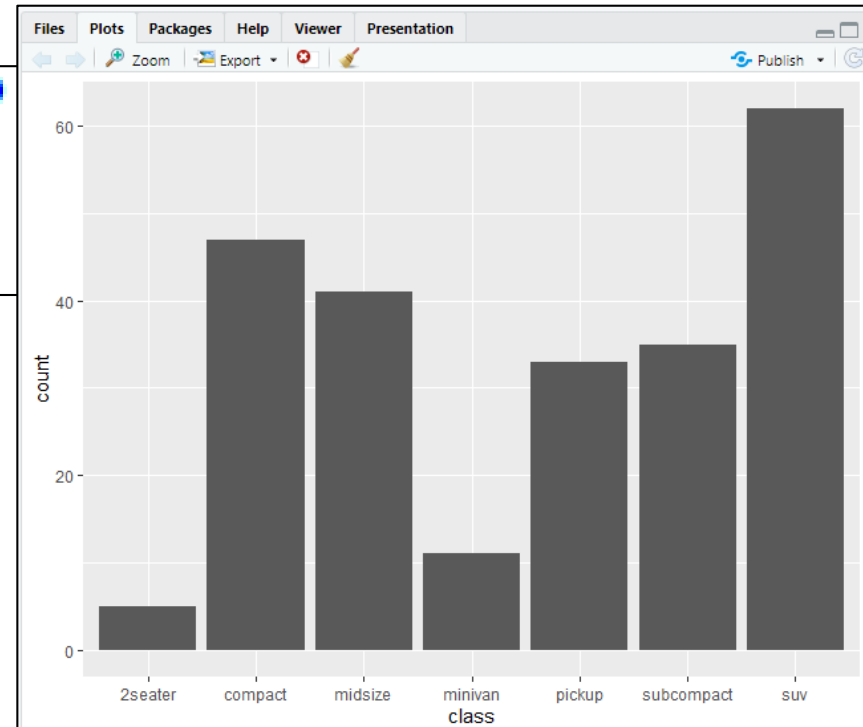
```
> str(datos)
tibble [234 × 11] (S3: tbl_df/tbl/data.frame)
 $ manufacturer: chr [1:234] "audi" "audi" "audi" "audi" ...
 $ model       : chr [1:234] "a4" "a4" "a4" "a4" ...
 $ displ      : num [1:234] 1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
 $ year       : int [1:234] 1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
 $ cyl        : int [1:234] 4 4 4 4 6 6 6 4 4 4 ...
 $ trans      : chr [1:234] "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
 $ drv        : chr [1:234] "f" "f" "f" "f" ...
 $ cty        : int [1:234] 18 21 20 21 16 18 18 18 16 20 ...
 $ hwy        : int [1:234] 29 29 31 30 26 26 27 26 25 28 ...
 $ fl         : chr [1:234] "p" "p" "p" "p" ...
 $ class      : chr [1:234] "compact" "compact" "compact" "compact" ...
>
```


3. BARPLOTS

Paso 4. Haremos un barplot con una variable discreta que será class. Agruparemos los valores en función de sus palabras. Creamos un gráfico con datos y le decimos la columna del eje de las X. Le añadimos la función `geom_bar()` para crear un diagrama de barras con una variable discreta.

```
> grafico = ggplot(datos, aes(x=class))  
> grafico = grafico + geom_bar()  
> print(grafico)  
> |
```

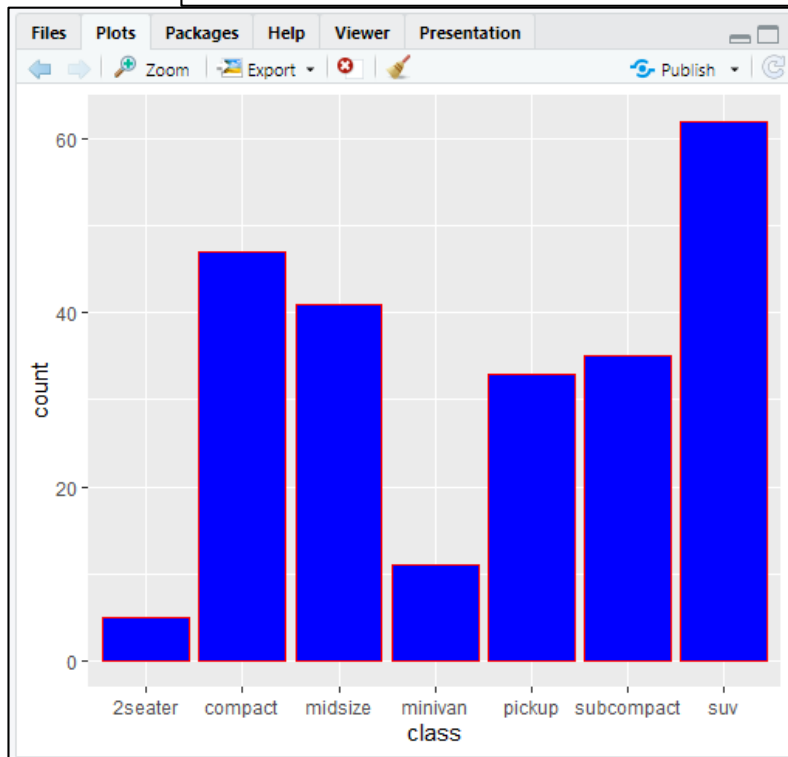
Ha creado un barplot con la variable discreta `class` que tiene diferentes valores como: Compact, Midsize, Minivan, pickup junto con el la frecuencias que aparecen.



3. BARPLOTS

Paso 5. Podemos cambiar el `geom_bar` y ponerle atributos, como por ejemplo el color: el contorno de color rojo y el relleno de color azul

```
> grafico = grafico + geom_bar(color='red',fill='blue')  
> print(grafico)  
> |
```

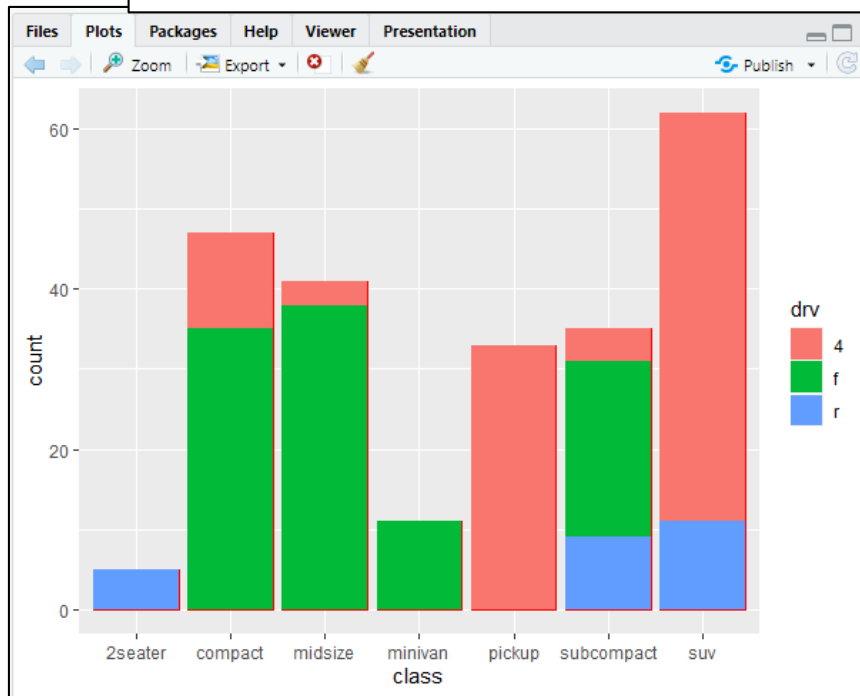


En lugar de color rojo,
se puede poner su
código alfanumérico

3. BARPLOTS

Paso 6. También podemos configurar en lugar de un color sólido, diferentes tipos de colores en función de una columna, por ejemplo, drv que tiene diferentes valores.

```
> grafico = grafico + geom_bar(aes(fill=drv))  
> print(grafico)  
>
```



En lugar de ser todo azul, ahora lo ha rellenado con los valores de drv. Podemos así comparar dos columnas, la columna class con su frecuencia y además sabiendo de qué tipo son por medio de otra columna drv

3. BARPLOTS

Paso 7. Podemos buscar información de `geom_bar` dentro de la pestaña help. Nos indica que es un diagrama de barras con la descripción de todos los argumentos que podemos ponerle. Al final suelen venir ejemplos de utilización del plot que estamos buscando.

