Programación científica en R

ggplot2

Marcos Ehekatzin García Guzmán

Septiembre de 2024

Introducción

En R existen distintos sistemas para hacer gráficas, pero una de las más versátiles es **ggplot2**.

- **ggplot2** implementa la *gramática de gráficos*, que es un sistema para describir y construir gráficas
- **ggplot2** es parte de **tidyverse** por lo que debemos cargar la librería para utilizarlo:

```
> library('tidyverse')
```

NOTA: Recuerden que solo es necesario instalar la paquetería una vez, pero cada que la utilicemos hay que cargarla en el script.

Utilizaremos un data frame que se incluye en **tidyverse** llamado *mpg*:

- Este data frame contiene infromación de 38 modelos de autos recolectada por la US Environment Protection Agency.

```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl trans      drv
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr>
## 1 audi         a4      1.8  1999     4 auto(l5)   f
## 2 audi         a4      1.8  1999     4 manual(m5) f
## 3 audi         a4      2    2008     4 manual(m6) f
## 4 audi         a4      2    2008     4 auto(av)   f
## 5 audi         a4      2.8  1999     6 auto(l5)   f
## 6 audi         a4      2.8  1999     6 manual(m5) f
```

Para hacer una gráfica con **ggplot2** empezaremos usando la función **ggplot()**:

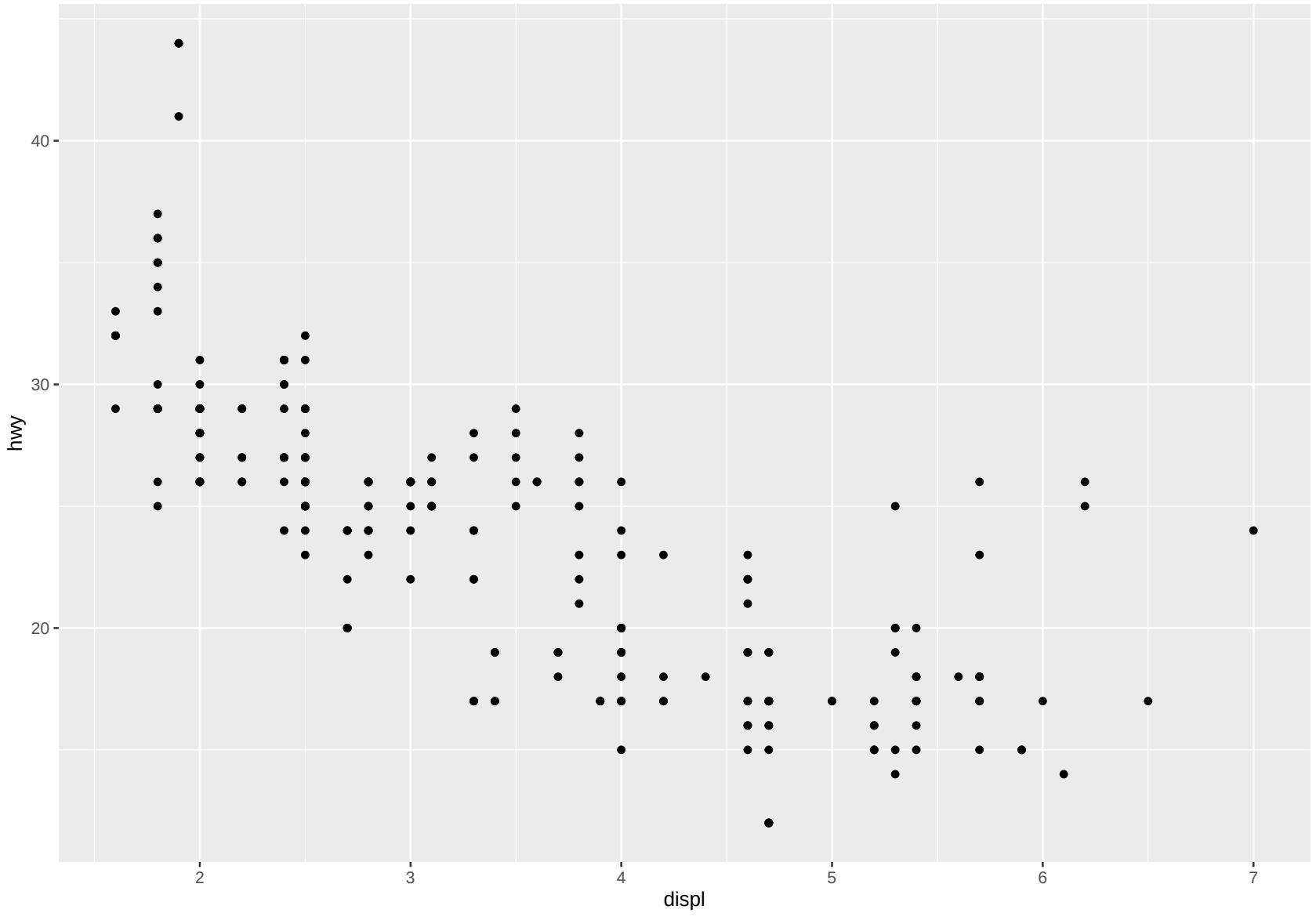
- El primer argumento de la función es el data frame que usaremos:

```
>   ggplot(data = mpg)
```

Para completar la gráfica hay que añadir capas.

- Para hacer una gráfica de dispersión, añadiremos una capa de puntos con la función **geom_point()**

```
>   ggplot(data = mpg) +
      geom_point(mapping = aes(x = displ, y = hwy))
```


hwy
40
30
20
2
3
4
5
6
7
displ

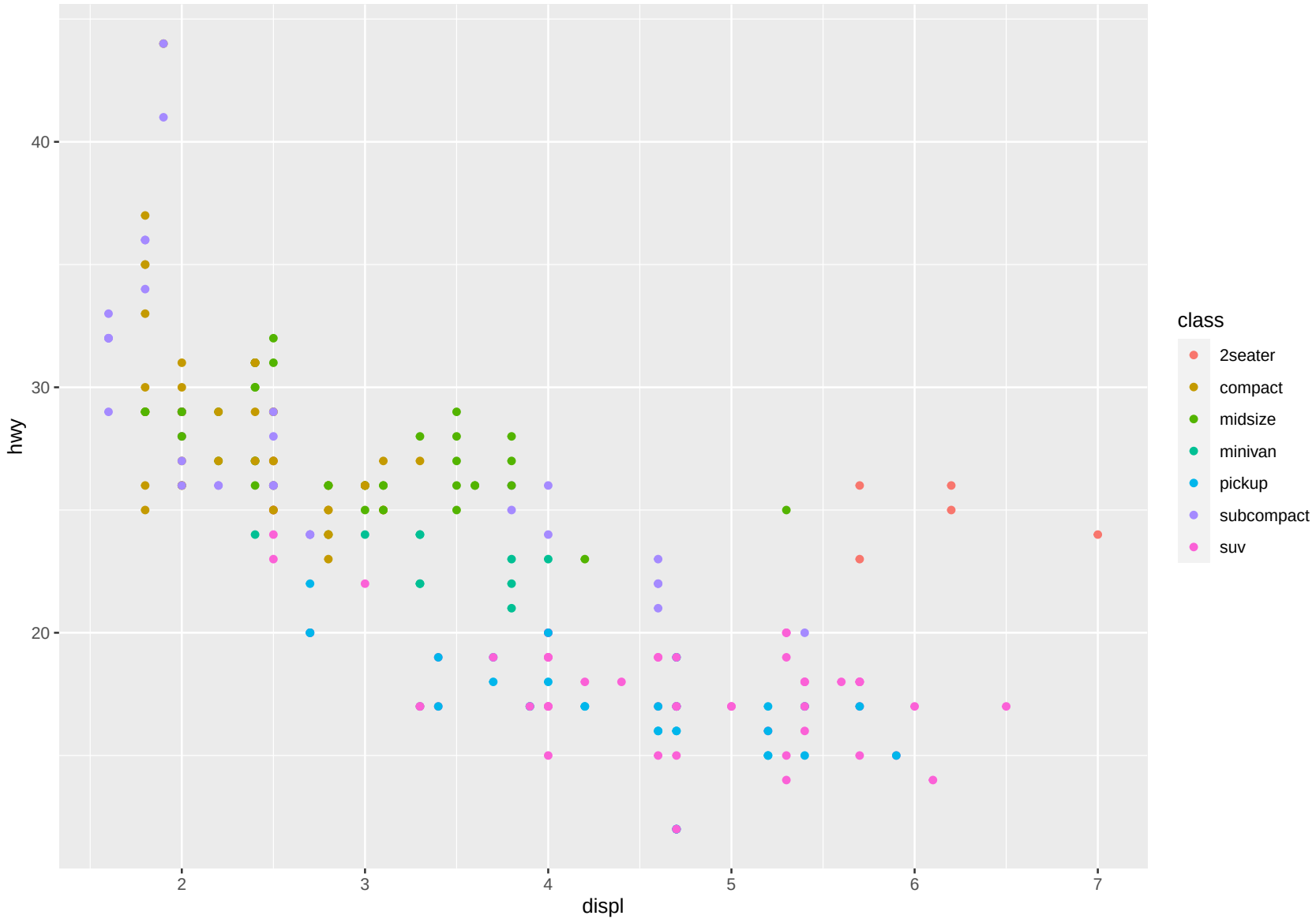
Podemos agregar una tercera variable a un scatterplot asignándola a un *aesthetic*:

- Un *aesthetic* es una propiedad visual de los objetos de la gráfica (tamaño, color, forma, etc.)

```
>   ggplot(data = mpg) +
      geom_point(mapping = aes(x=displ, y = hwy,
                               color = class))
```

- En este caso agragamos la estética color sobre la variable class. **ggplot2** asignará un color único para cada valor único de la variable.
  - Otras esteticas son el tamaño (**size**), la forma (**shape**) y la transparencia (**alpha**)

Mappings

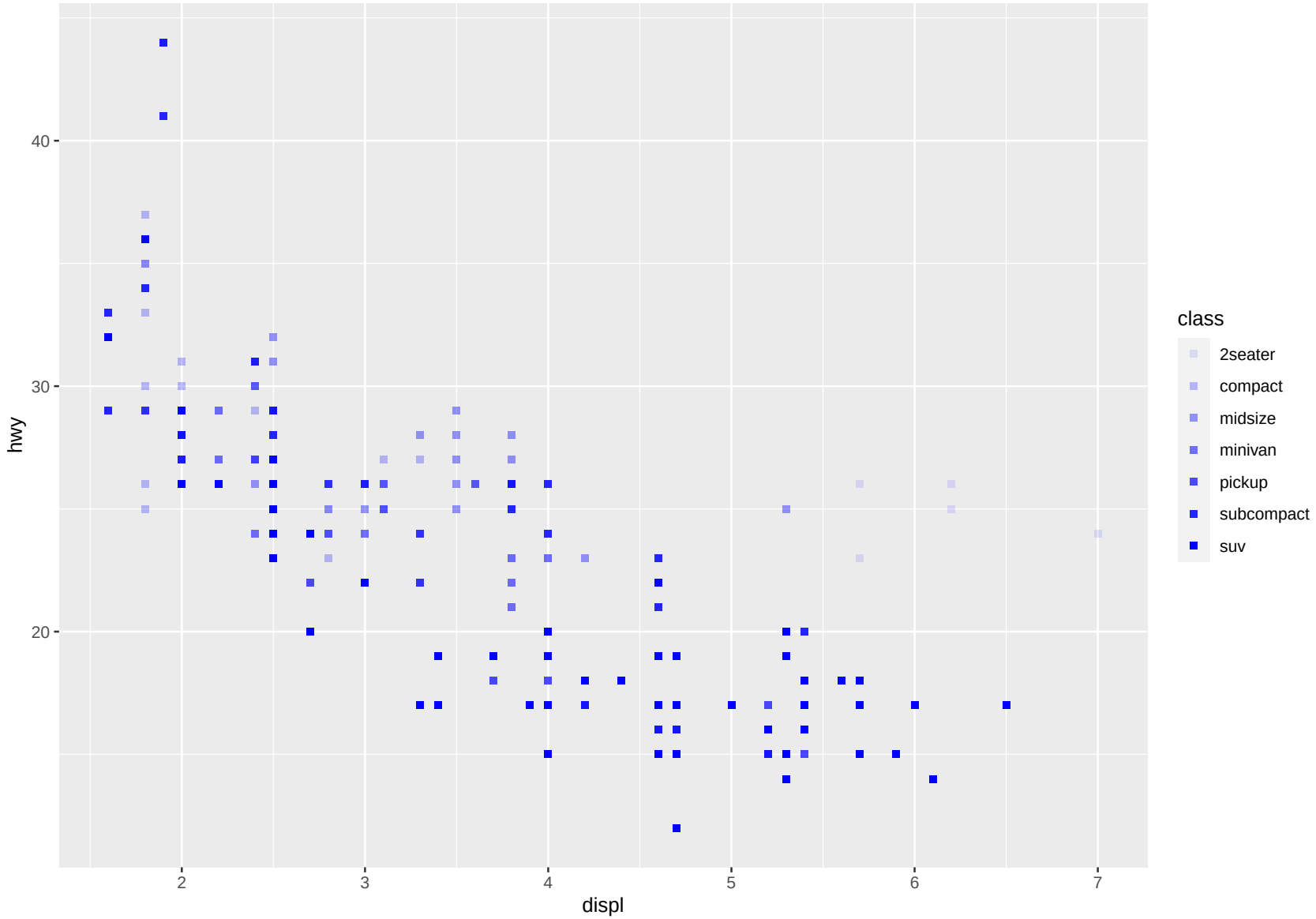

hwy
40
30
20
2
3
4
5
6
7
displ
class
2seater
compact
midsize
minivan
pickup
subcompact
suv

También podemos agregar estéticas a la gráfica sin que estas aporten información sobre las variables.

- Por ejemplo, hagamos la misma gráfica pero cambiemos la forma y el color de todos los puntos.
- La única estética que aportará información será la transparencia.

```
> ggplot(data = mpg)+
    geom_point(mapping = aes(x = displ, y = hwy,
                             alpha = class),
                             color = "blue", shape =0)
```

```
## Warning: Using alpha for a discrete variable is not advi
```



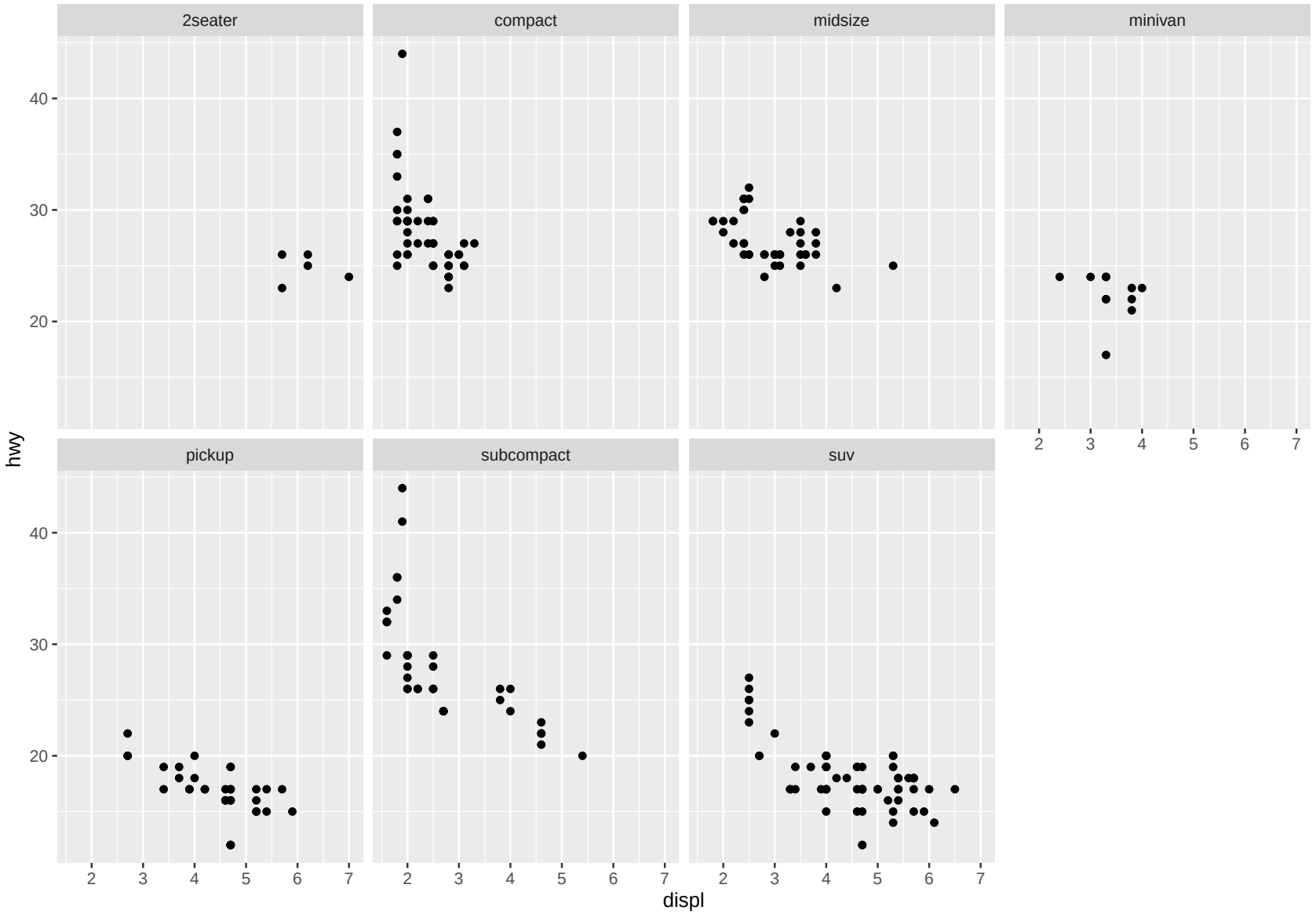
Facets

También podemos separar nuestra gráfica en *facetas* (subgráficas que muestran distintos sobconjuntos de nuestros datos)

Usaremos la función **facet_wrap()**: - El primer argumento debe ser una fórmula, la cual crearemos con el símbolo ~ y después pondremos el nombre de una variable **discreta**.

```
>   ggplot(data = mpg)+
      geom_point(mapping = aes(x= displ, y = hwy)) +
      facet_wrap(~ class, nrow = 2)
```

Facets


2seater
compact
midsize
minivan
pickup
subcompact
suv
hwy
displ
40
30
20
2
3
4
5
6
7

Objetos geométricos

Objetos geométricos

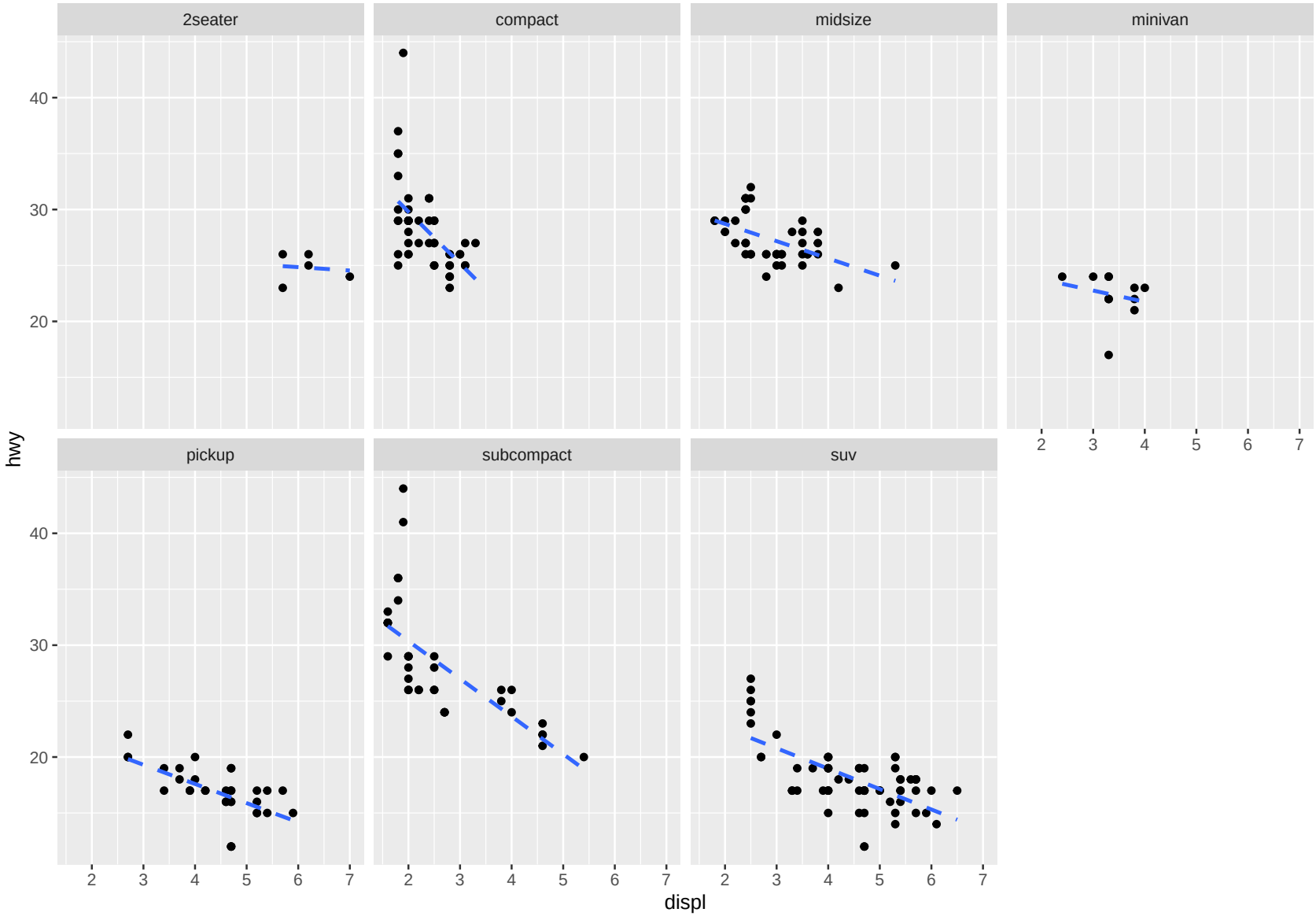
Las gráficas con ggplot utilizan distintos *geoms* (objetos geométricos que se utilizan para representar datos).

- Las gráficas de dispersión usan puntos, pero hay muchos otros objetos (lineas, barras, polígonos, etc.)
- Podemos utilizar varios objetos para crear una sola gráfica y asignar distintas estéticas a cada uno:

```
>   ggplot(data = mpg, mapping = aes(x=displ, y =hwy)) +
      geom_point() +
      geom_smooth(method = lm,
                  linetype = "dashed", se = F) +
      facet_wrap(~ class, nrow = 2)
```

Objetos geométricos

```
## `geom_smooth()` using formula 'y ~ x'
```



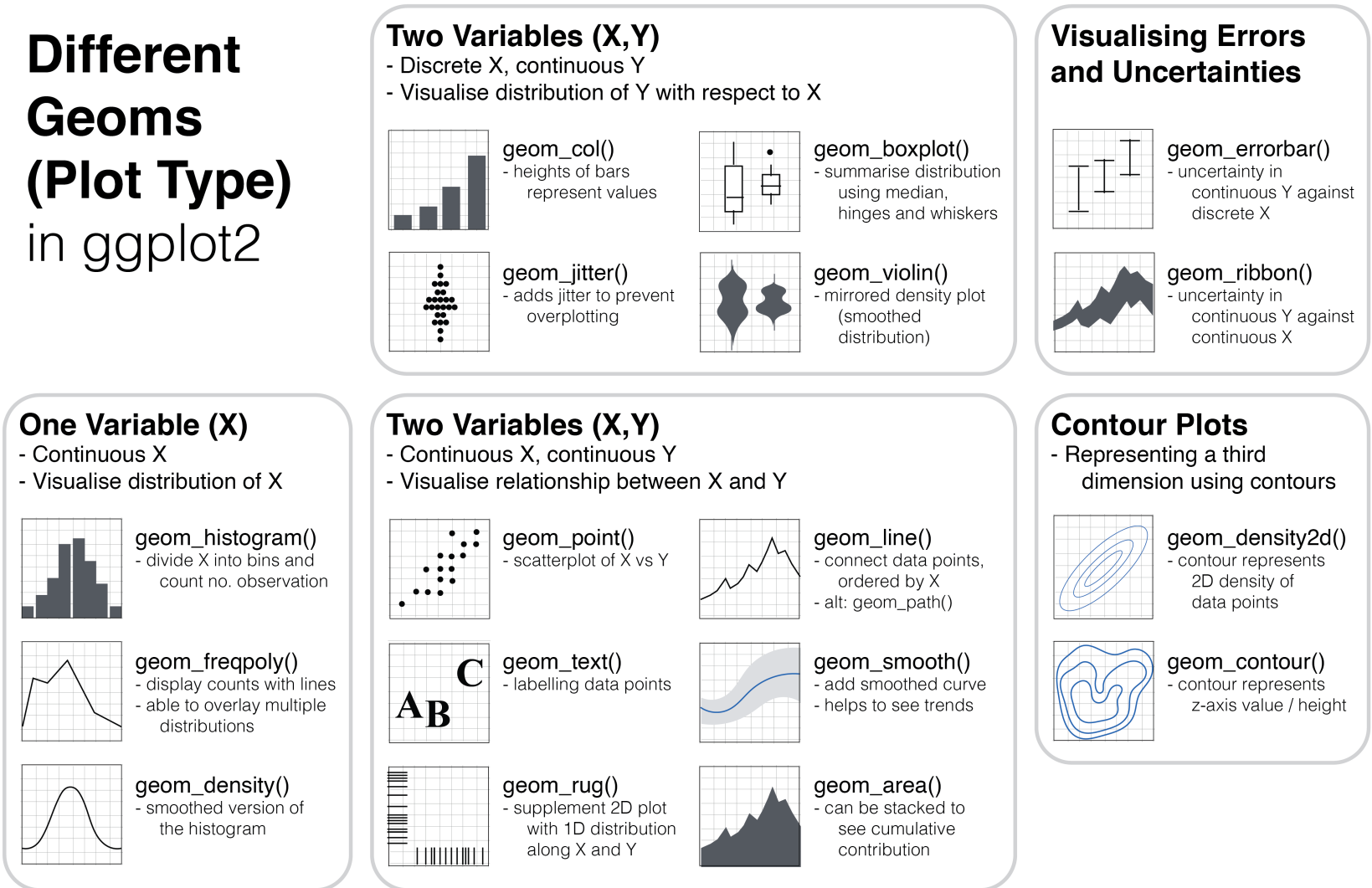


Figura 1: Objetos geométricos

Transformaciones estadísticas

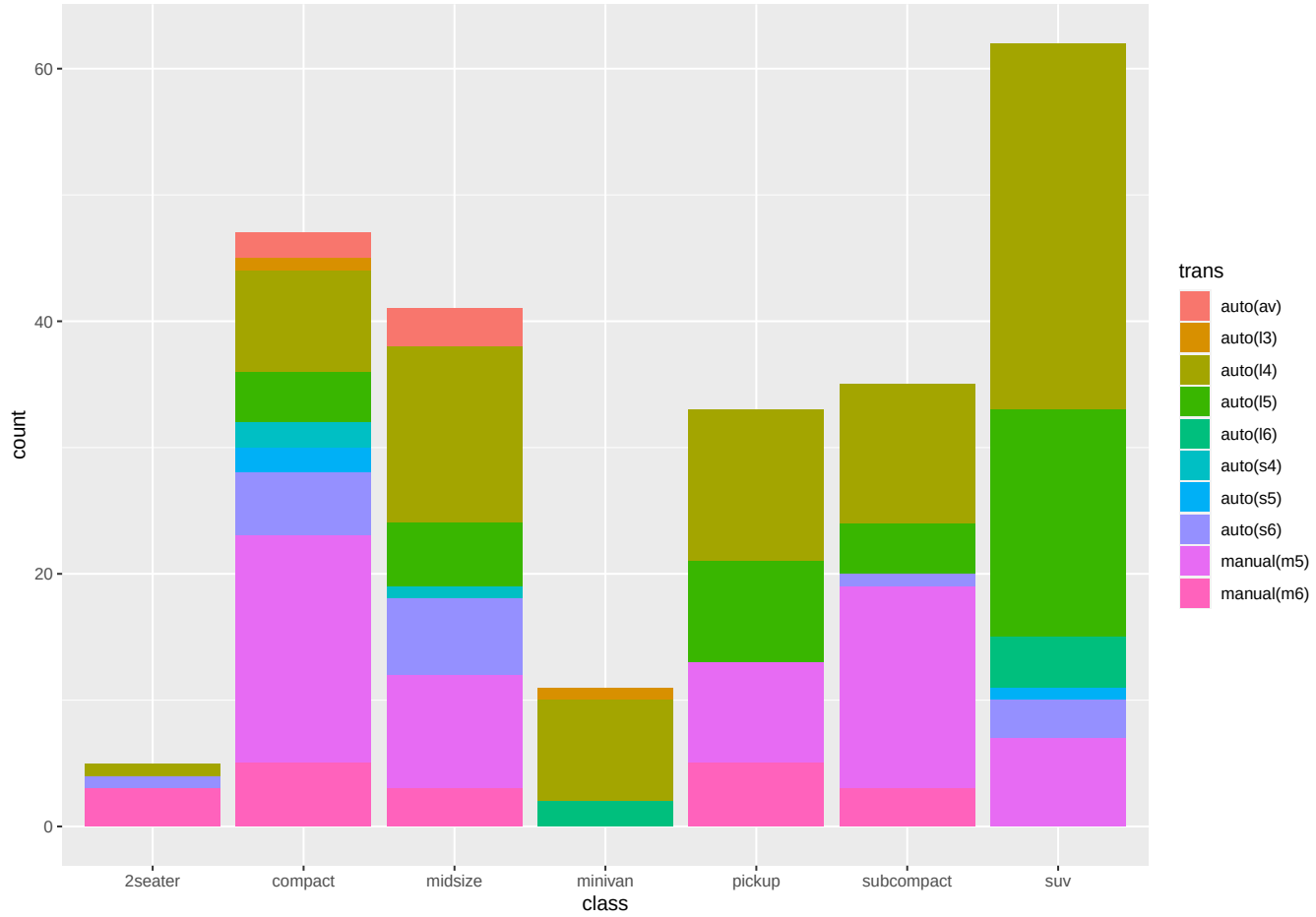
Transformaciones estadísticas

Las gráficas de dispersión grafican los datos directamente del data frame, mientras que otro tipo de gráficas calculan nuevas variables.

- Las gráficas de barras, histogramas y polígonos de frecuencia agrupan los datos en bins y grafican el número de puntos en cada uno de esos bins.
- Las gráficas de caja hacen un resumen de la distribución.

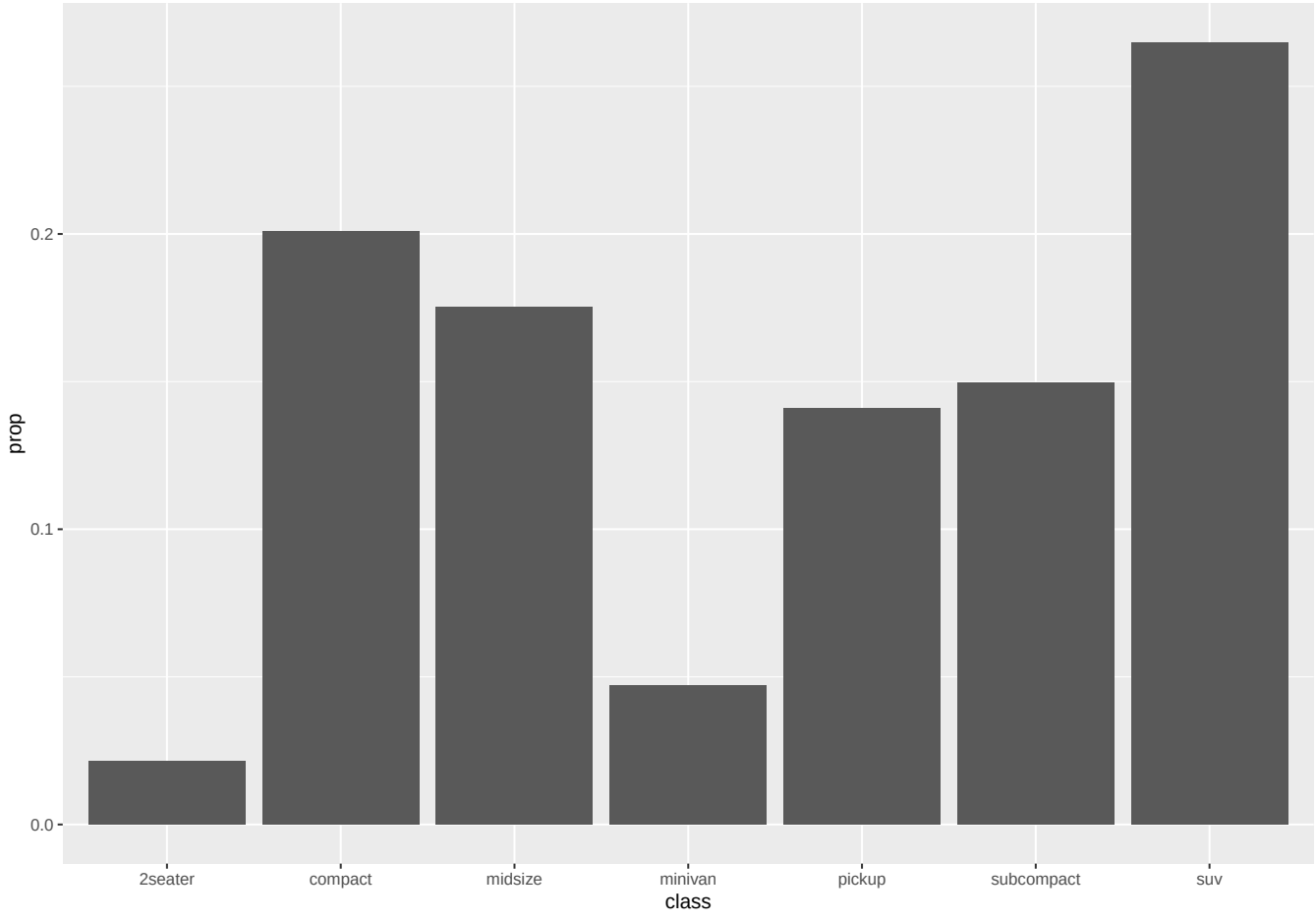
Ejemplo: Gráficas de barras (count)

```
ggplot(data = mpg)+
  geom_bar(mapping = aes(x = class, fill = trans))
```



Ejemplo: Gráficas de barras (prop)

```
ggplot(data = mpg)+
  geom_bar(mapping = aes(x = class,y = after_stat(prop),
                         group = 1,))
```



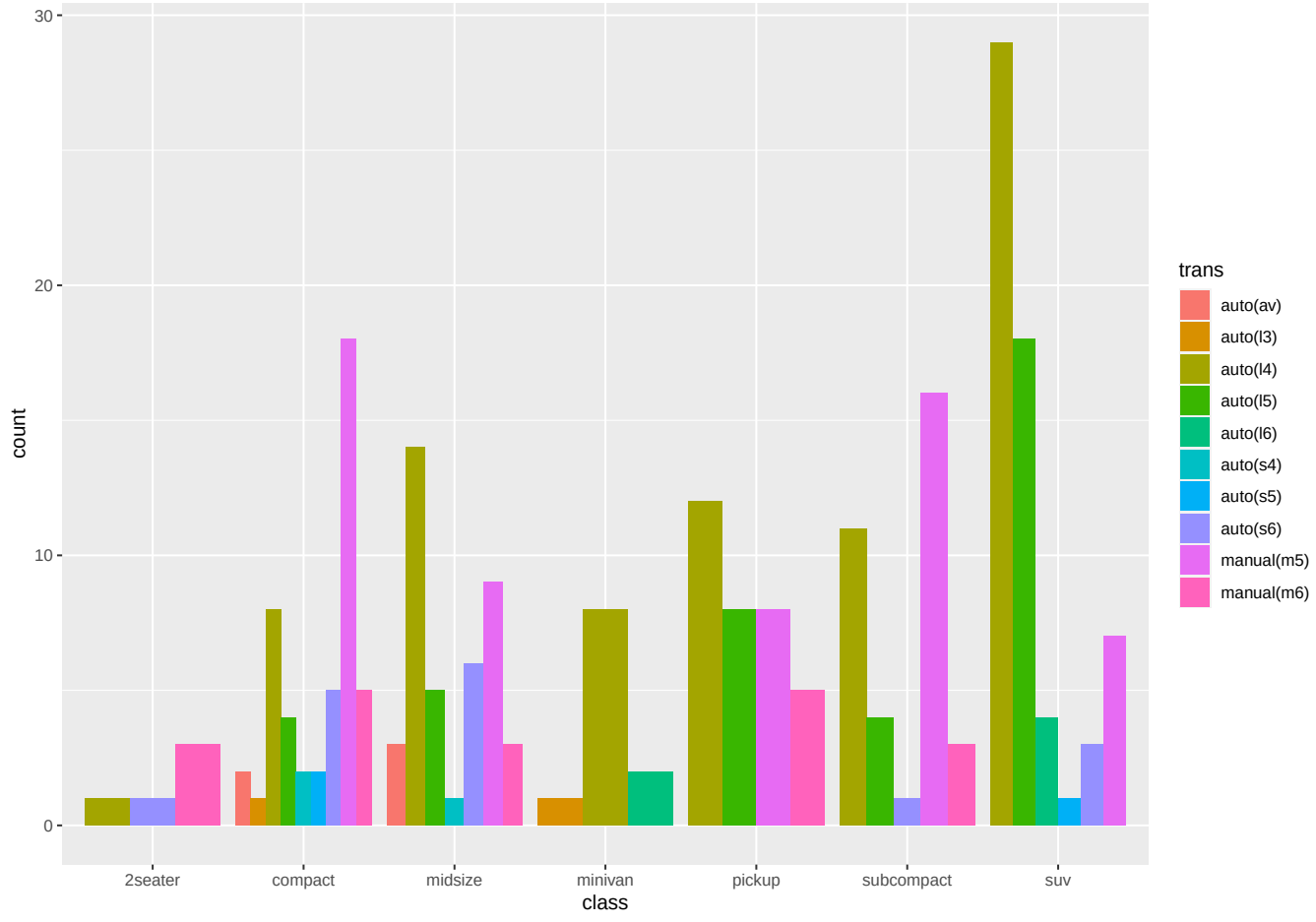
Ajustes de posisión

Ajustes de posisión

- **position = "fill"**: Genera una gráfica de barras apilada en donde todas las barras suman 1
- **position = "dodge"**: En lugar de hacer una gráfica de barras apilada, pone cada componente a un lado de otro.
- **position = "jitter"**: [Solo para scatterplot] Agrega una cantidad de ruido aleatorio a cada punto.

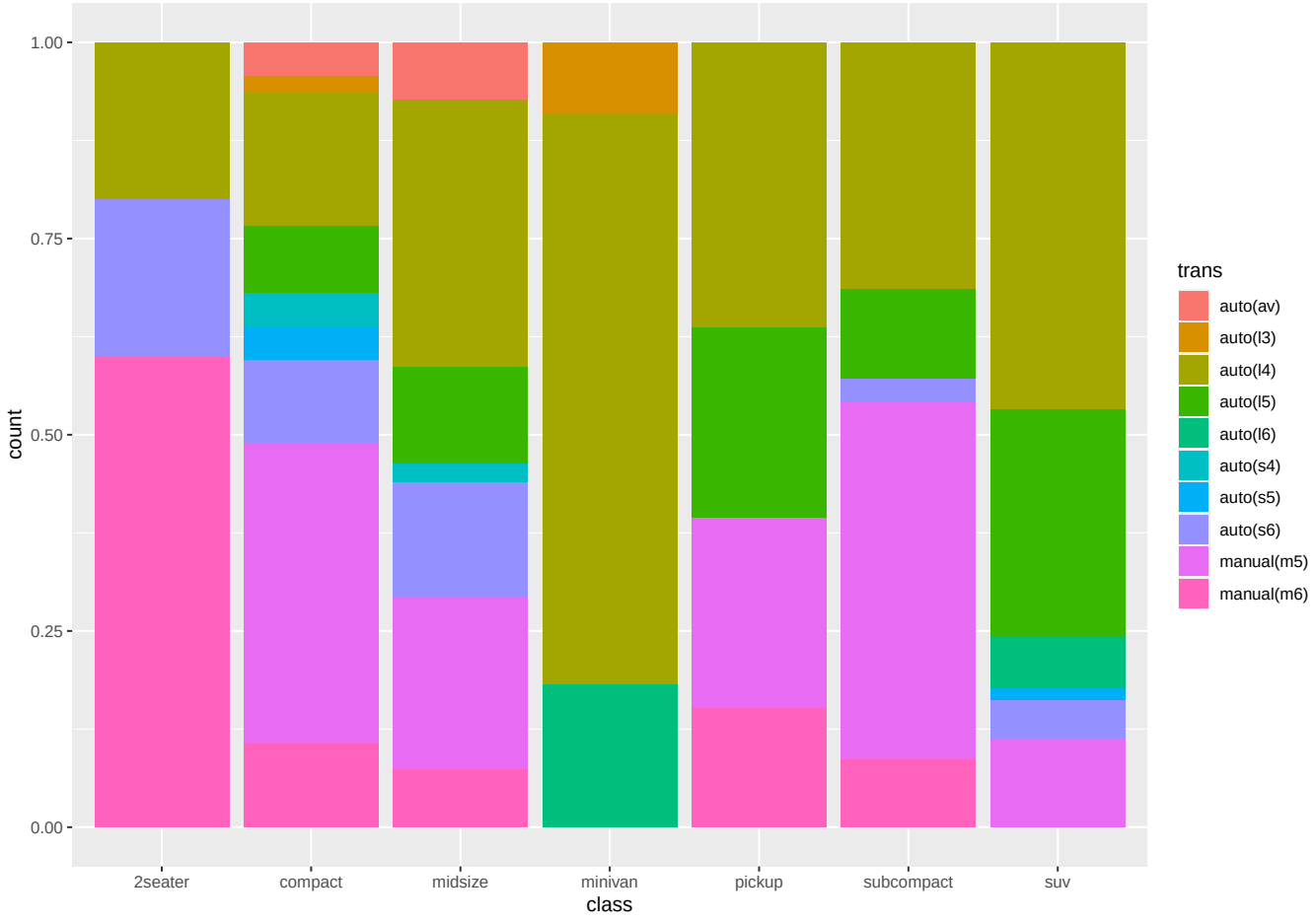
Ajustes de posisión (dodge)

```
ggplot(data = mpg)+
  geom_bar(mapping = aes(x = class, fill = trans),
           position = "dodge")
```



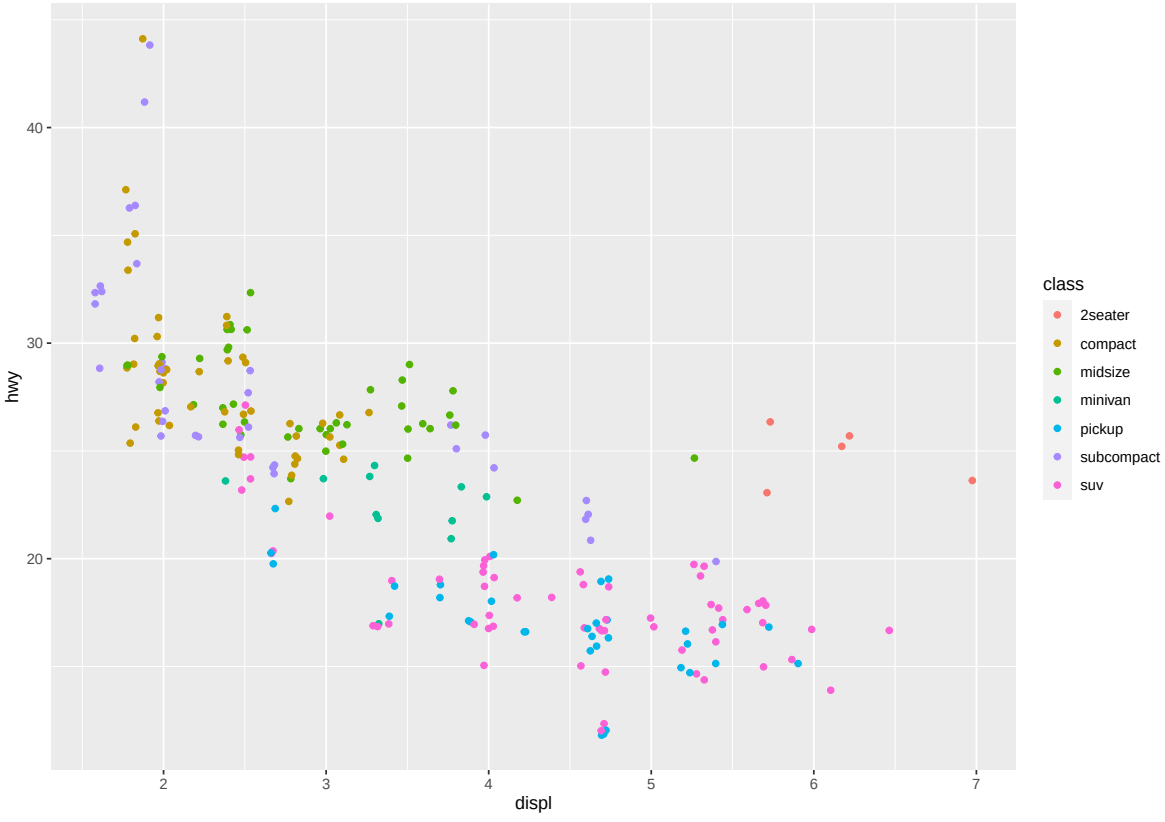
Ajustes de posisión (fill)

```
ggplot(data = mpg)+
  geom_bar(mapping = aes(x = class, fill = trans),
           position = "fill")
```



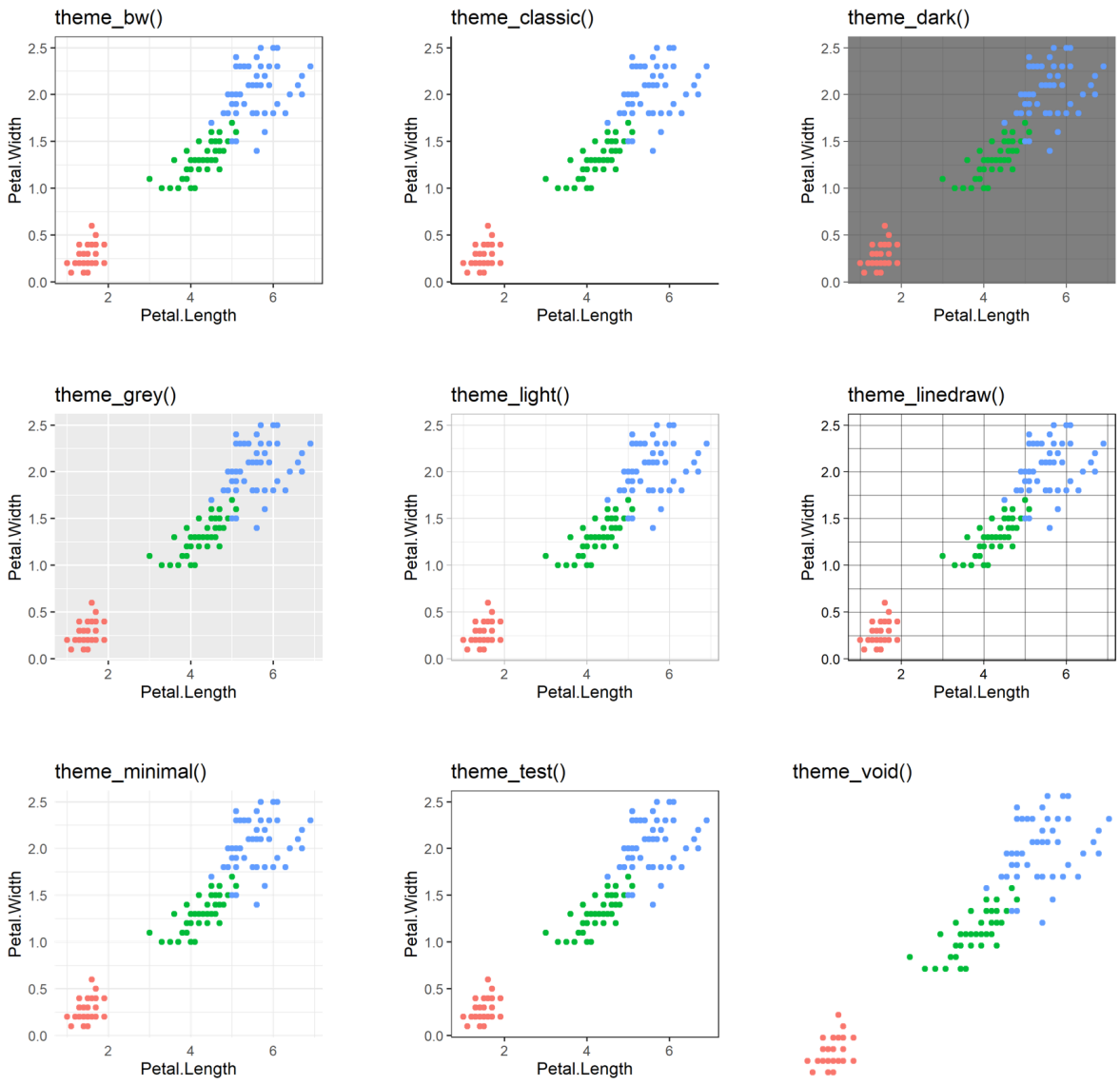
Ajustes de posisión (jitter)

```
ggplot(data = mpg) +
  geom_point(mapping = aes(x=displ, y = hwy,
                           color = class),
             position = "jitter")
```



Temas

Figura 2: Temas



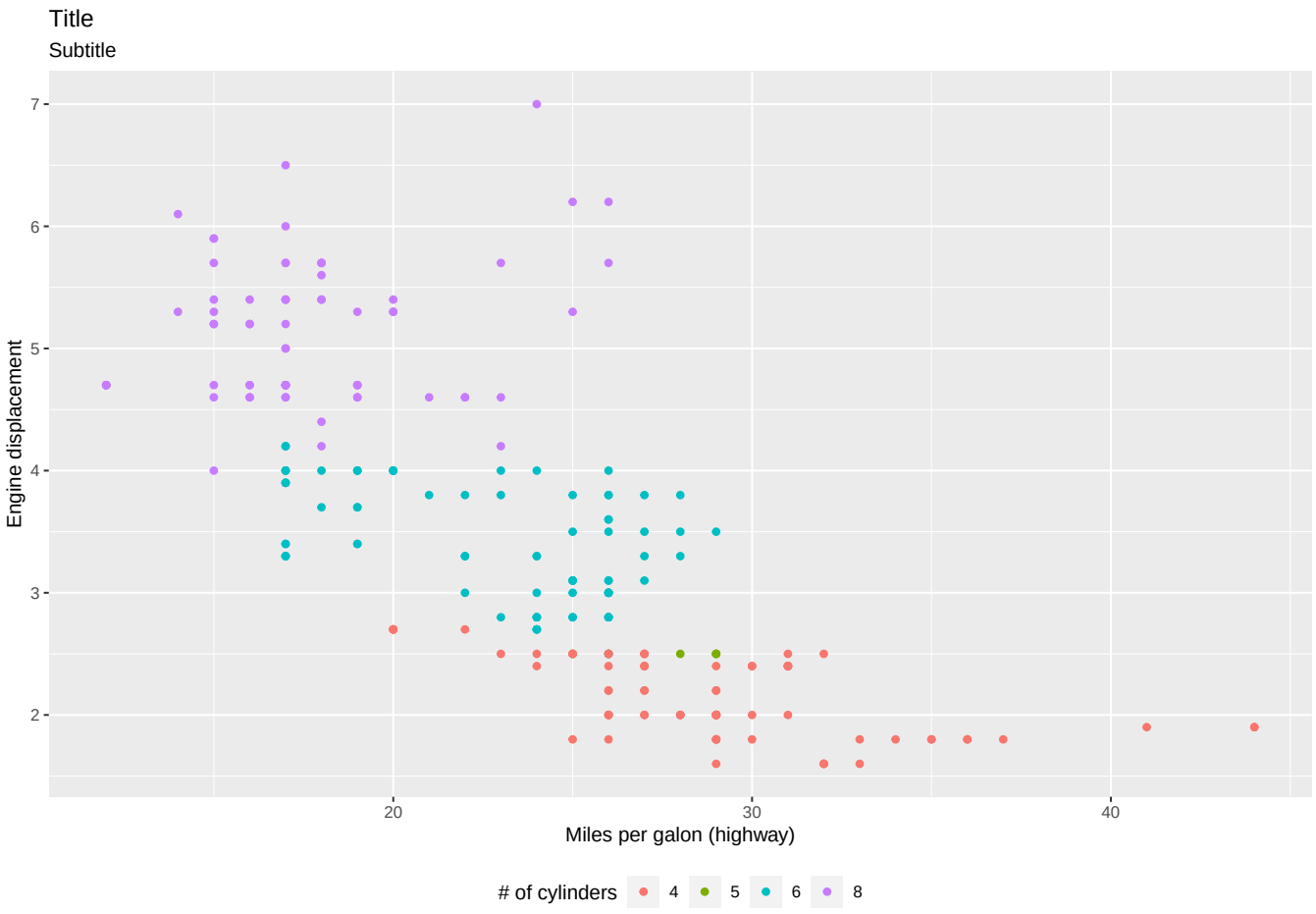
Títulos, labels y legends

- Para modificar títulos y etiquetas utilizaremos la función `labs()`:

```
p <- ggplot(data = mpg) +
  geom_point(mapping = aes(x = hwy, y = displ,
                           color = as.factor(cyl)))+
  labs(title = "Title",
       subtitle = "Subtitle",
       colour = "# of cylinders",
       x = "Miles per galon (highway)",
       y = "Engine displacement") +
  theme(legend.position ="bottom")
```

Títulos, labels y legends

```
p
```



Ejercicios:

Ejercicio ENOE

- Para este ejercicio descargaremos la ENOE del primer y segundo trimestre de 2023 y seguiremos los siguintes pasos:

1. Después de cargar cada base, eliminaremos todas las observaciones en donde *ingocup*==999998 | *ingocup*==999999 y nos quedaremos solo con los individuos con edad (eda) entre 15 y 65 años y las observaciones donde *tipo* == 1.

2. Nos quedaremos con las variables ing_x_hrs, anios_esc, fac_tri, emp_ppal y las variables llave (excepto tipo y mes_cal).

- Nota:hay que cambiar el ingreso por hora (ing_x_hrs) a NA cuando este sea mayor a 999998 (`ing_x_hrs = na_if(ing_x_hrs, 999998)`)

Ejercicio ENOE

3. Juntaremos ambas bases de tal manera que solo nos quedemos con las observaciones que hagan match. (Noten como cambian los nombres de las variables)
4. Haremos una nueva variable que denote el tipo de transición entre el sector formal y el informal.

Ejercicio ENOE

5. Replicaremos las siguientes gráficas

- Dw = diferencia entre el logaritmo del ingreso por hora en cada trimestre.
- ling.q2 = logaritmo del ingreso por hora en el segundo trimestre.