ggplot

Rodrigo Negrete Pérez

January 29, 2022

1. Merge
2. Intro a ggplot
3. Graficar datos agrupados
4. Infinita personalización
5. ggsave()

Section 1

Merge

Merge

La sesión anterior hablamos un poco de los merge, pero no hicimos ninguno.

- Utilizamos la base de minsitros de Nyrup y Bramwell. . . bueno una de ellas
- Descarguemos la otra y hagamos un merge
- La pueden encontra en Github en la carpeta de ggplot.
- Descarguen de nuevo la base de minsitros pasada ¿Cómo?

```
setwd('C:/Users/rodri/OneDrive - INSTITUTO TECNOLOGICO AUTONOM

library(tidyverse)

min_within<-readxl::read_excel('WhoGov_within_V1.1.xlsx',
                               guess_max=999999999)
min_cross<-readxl::read_excel('WhoGov_crosssectional_V1.1.xlsx
                              guess_max=9999999)
```

- ¿En qué se diferencian?
- ¿Cómo podemos ver las diferencias?
- Si queremos añadir los datos del gobierno a la base de ministros, ¿qué tipo de merge tenemos que hacer?

```
min<-left_join(min_within,
               min_cross,
               by=c('year'='year',      'country_name'='countr
```

- Sale un error, frecuente con los merge, es algo fácilmente corregible.

```
min_cross$year<-as.numeric(min_cross$year)
min_within$year<-as.numeric(min_within$year)
```

- ¿Cómo lo harías con dplyr?

- Reintentemos el merge

```
min<-left_join(min_within,
               min_cross,
               by=c('year'='year', 'country_name'='country_nam
```

Section 2

Intro a ggplot

- R tiene funciones para graficar incorporadas.
- Sin embargo, ggplot es el paquete más utilizado.
- Tiene una sintaxis estándar, amplia personalización y muchos otros paquetes adicionales.
- Su principal virtud es que podemos añadir elementos fácilmente

ggplot: sintaxis

- Primero debemos especificar:
  - Base de datos
  - aes: variables en los ejes
- Segundo: Geometría->Linea, barras, puntos, etc.
- Tercero: Estética
  - Título
  - Ejes
  - Labels

- Dentro de ggplot() ponemos el df y aes
  - En aes() especificamos nuestras dos variables
- Añadimos un + y ponemos después la geometría

```
ggplot(data= ,
       aes(x= , y= ))+
  geom_line() # geom_
```

Ejemplo

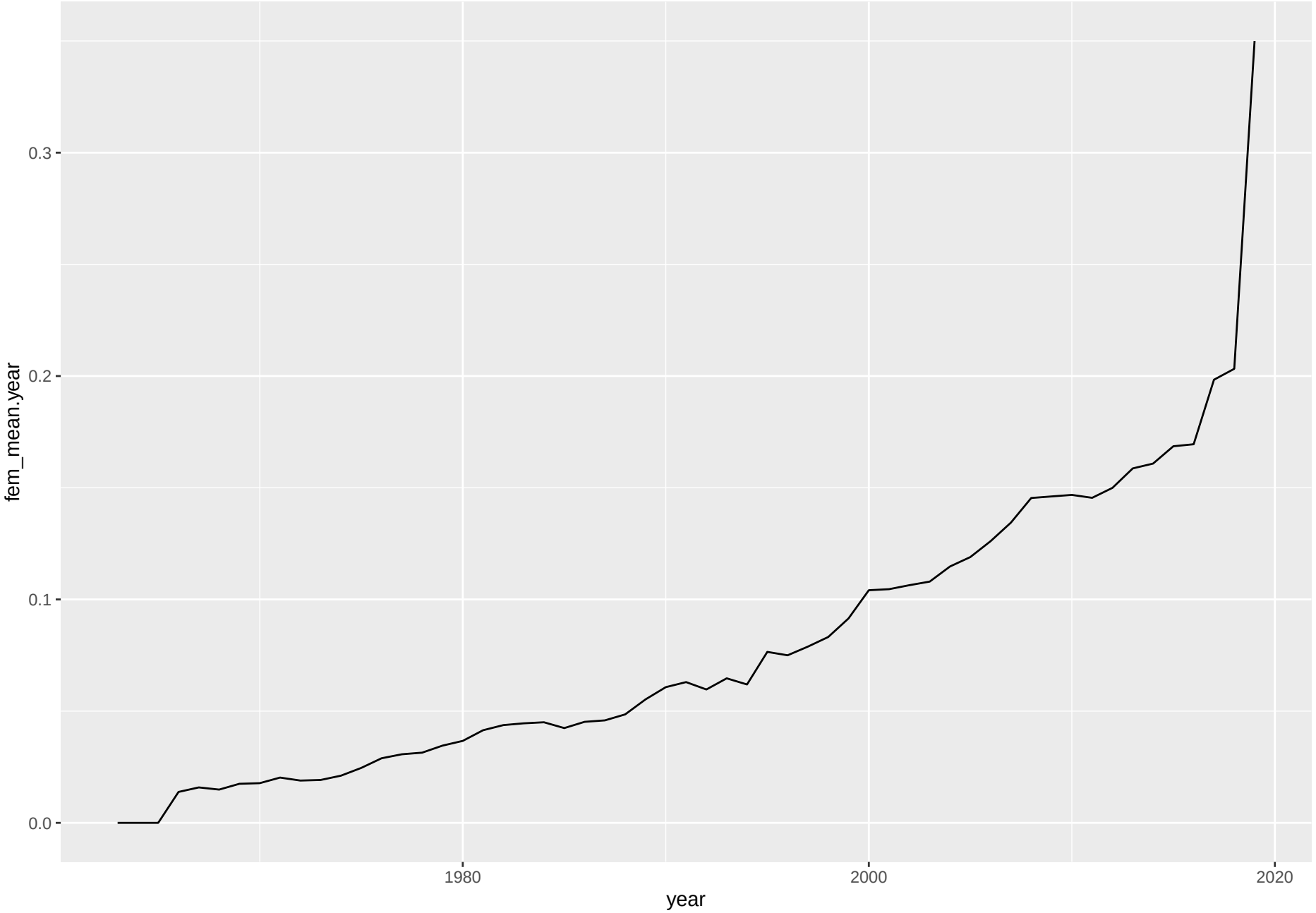
- La sesión pasada obtuvimos el promedio de mujeres en el gabinete por año

```
min<- min %>% mutate(
    female=ifelse(gender=='Female',1,0))

min_grouped<- min %>%
  group_by(year) %>%
  summarise(
    fem_mean.year=mean(female,
                       na.rm=T) )
```

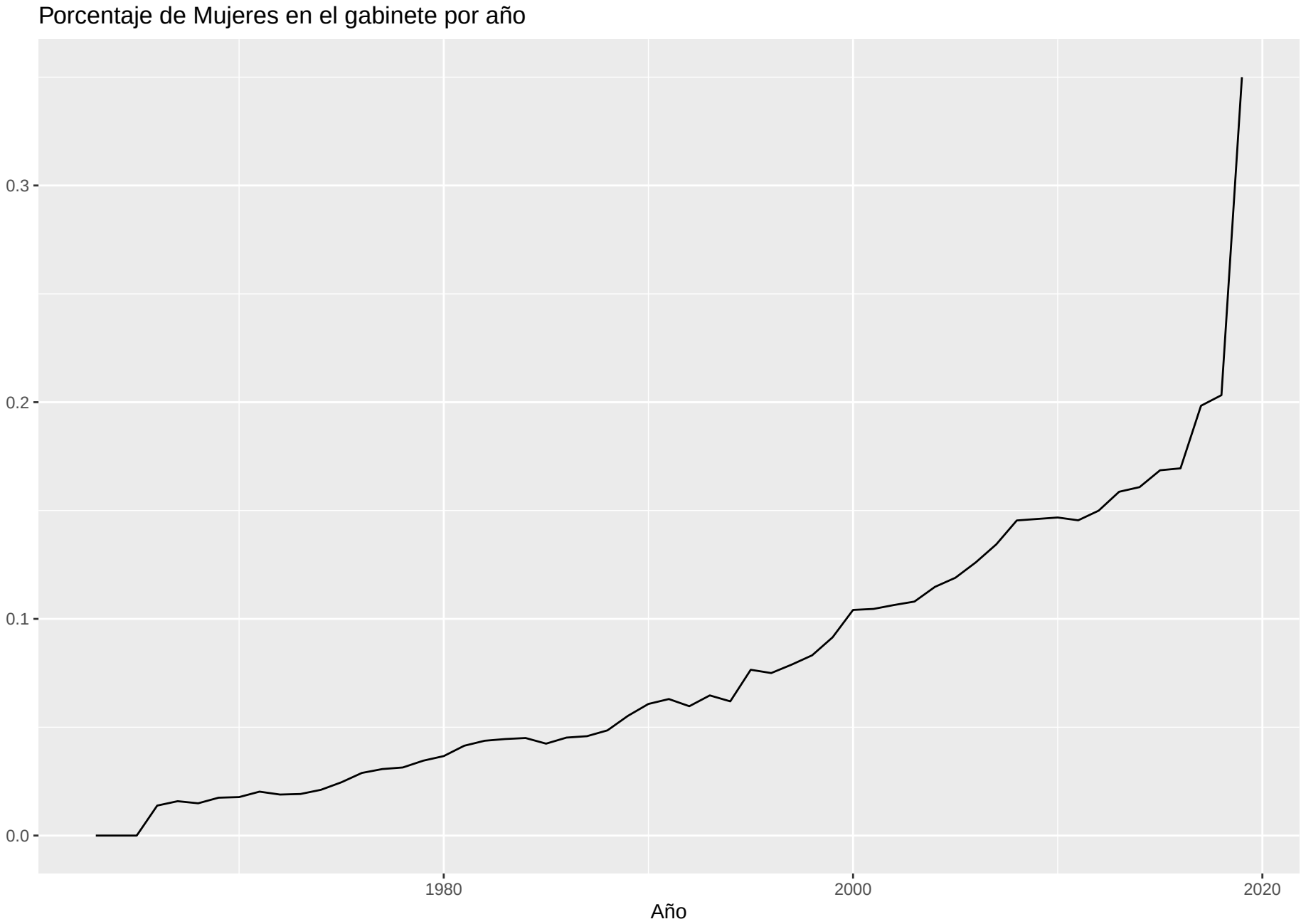
- Intentemos graficar con ggplot

```
ggplot(data=min_grouped,
       aes(x=year, y=fem_mean.year))+
  geom_line()
```


0.3
0.2
0.1
0.0
fem_mean.year
1980
2000
2020
year

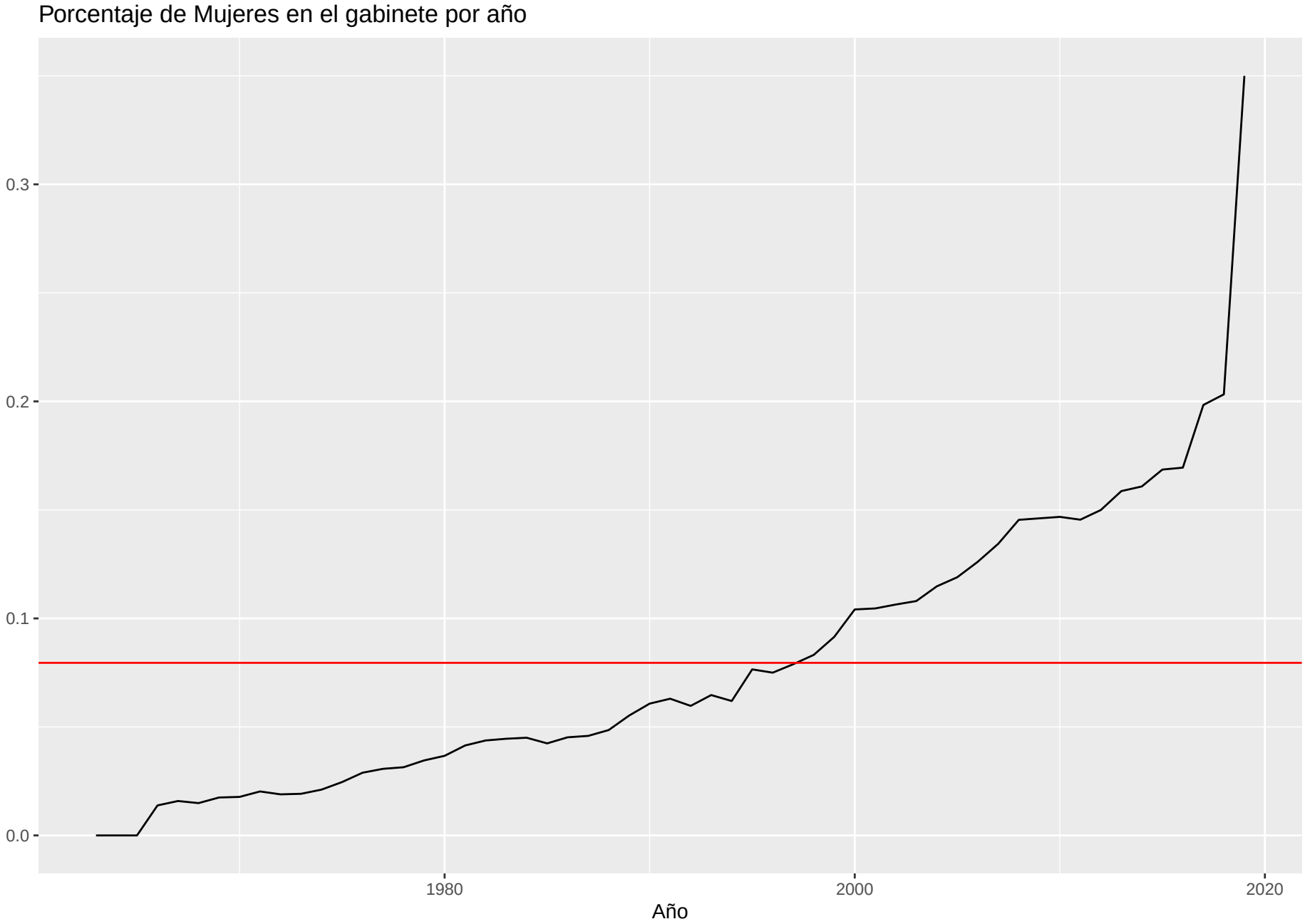
##

```
ggplot(data=min_grouped,
       aes(x=year, y=fem_mean.year))+
  geom_line()+
  ggtitle('Porcentaje de Mujeres en el gabinete por año')+
  xlab('Año')+
  ylab('')
```


Porcentaje de Mujeres en el gabinete por año
0.3
0.2
0.1
0.0
1980
2000
2020
Año

- Podemos añadir una figura adicional
- Por ejemplo, intentemos poner el promedio de porcentaje a lo largo de todo el periodo como una línea horizontal

```
ggplot(data=min_grouped,
       aes(x=year, y=fem_mean.year))+
  geom_line()+
  ggtitle('Porcentaje de Mujeres en el gabinete por año')+
  xlab('Año')+
  ylab('')+
  geom_hline(yintercept = mean(fem_mean.year), color='r')
```


Porcentaje de Mujeres en el gabinete por año
0.3
0.2
0.1
0.0
1980
2000
2020
Año

Section 3

Graficar datos agrupados

- Una característica útil de ggplot es que puede graficar datos agrupados fácilmente
- ggplot asigna un color a cada categoría.

Línea

- Supongamos que queremos ver la proporción de mujeres en el gabinete, pero ahora por régimen político.
- Primero tenemos que crear el df
  - De hecho lo podemos crear directamente en la función de ggplot.

```
min_regime<-min %>% group_by(system_category) %>% summarise(
  fem.mean_regime=mean(female,
                       na.rm=T)
)
```

```
## # A tibble: 11 x 2
##    system_category              fem.mean_regime
##    <chr>                                  <dbl>
##  1 Civilian dictatorship                0.0628
##  2 Crown Colony                         0
##  3 French Overseas Territory            0
##  4 Military dictatorship                0.0485
##  5 Mixed democratic                     0.125
##  6 Parliamentary democracy              0.130
##  7 Part of Yugoslavia                   0.0280
##  8 Presidential                         0.118
##  9 Presidential democracy               0.124
## 10 Royal dictatorship                   0.0304
## 11 <NA>                                 0.00963
```

Gráfico de barras

Supongamos que queremos hacer un gráfico de barras

- Queremos que haya una barra para cada régimen
- También queremos colores distintos.

Tenemos que decirle a ggplot que agrupe por la categoría y que coloreé por la categoría

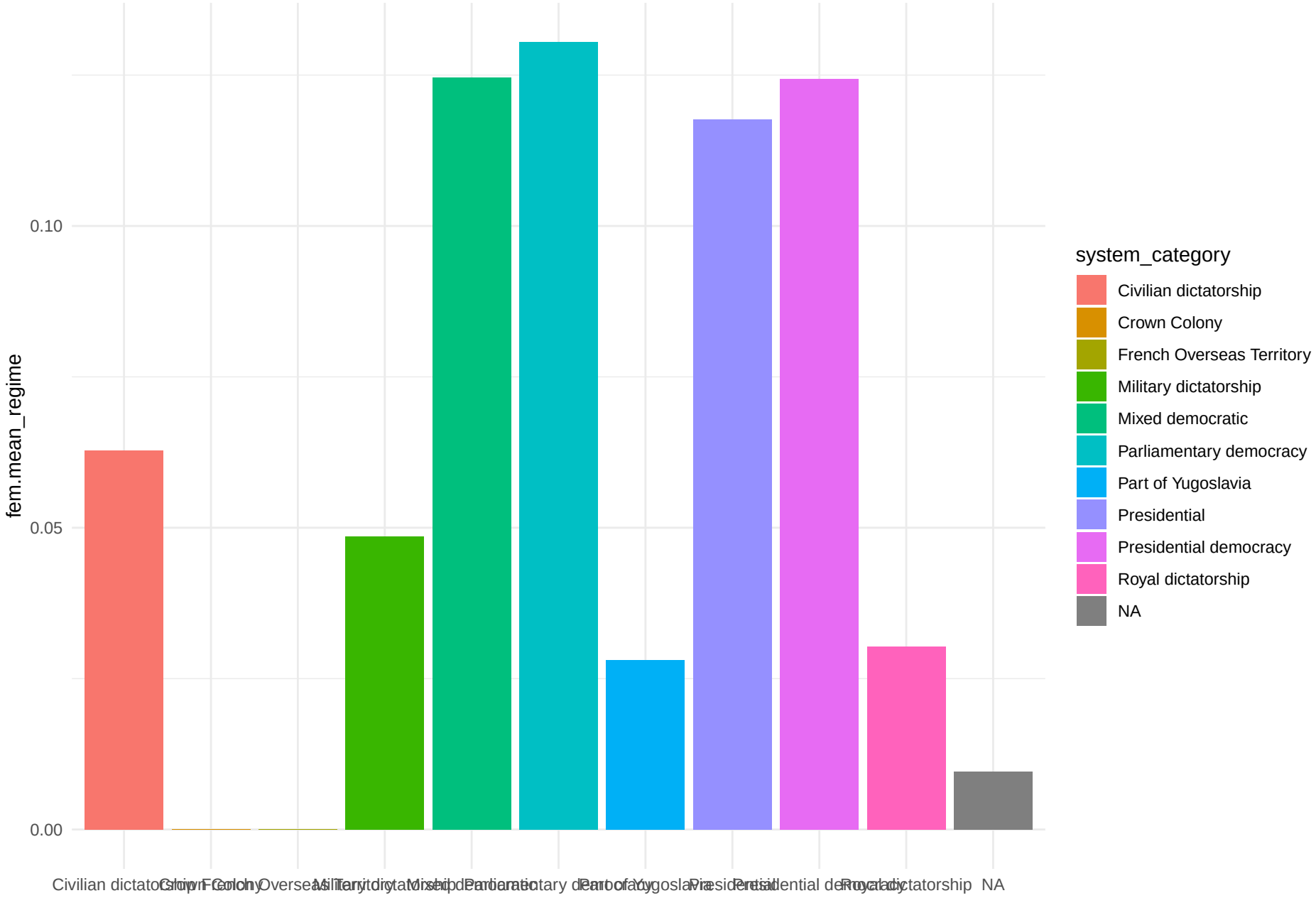
Un par de consideraciones:

- color es para líneas; fill, para el relleno.
  - En el caso de una gráfica de barras, queremos fill.
- Podemos poner fill/color dentro o fuera de aes(), pero los resultados son distintos:
  - Dentro-> ggplot colorea la figura agrupando por la variable (lo que queremos en este caso)
  - Fuera-> ggplot interpreta que quieres ese color específico o paleta de colores: 'red','black'.
- ggplot automáticamente inserta una leyenda con un código de colores para las categorías

- Entonces, nosotros queremos el color dentro de aes()

```
ggplot(min_regime,
       aes(system_category, fem.mean_regime,
           fill=system_category))+
  geom_bar(stat = 'identity')+
  theme_minimal()
```

- Las gráficas de barras pueden apilarse o ponerse una al lado de la otra: de ahí que especifiquemos stat=‘identity’
- con theme() podemos cambiar el estilo


fem.mean_regime
0.10
0.05
0.00
Civilian dictatorship
Crown Colony
French Overseas Territory
Military dictatorship
Mixed democratic
Parliamentary democracy
Part of Yugoslavia
Presidential
Presidential democracy
Royal dictatorship
NA
system_category
Civilian dictatorship
Crown Colony
French Overseas Territory
Military dictatorship
Mixed democratic
Parliamentary democracy
Part of Yugoslavia
Presidential
Presidential democracy
Royal dictatorship
NA

- Generemos algunos datos para los siguientes ejemplos

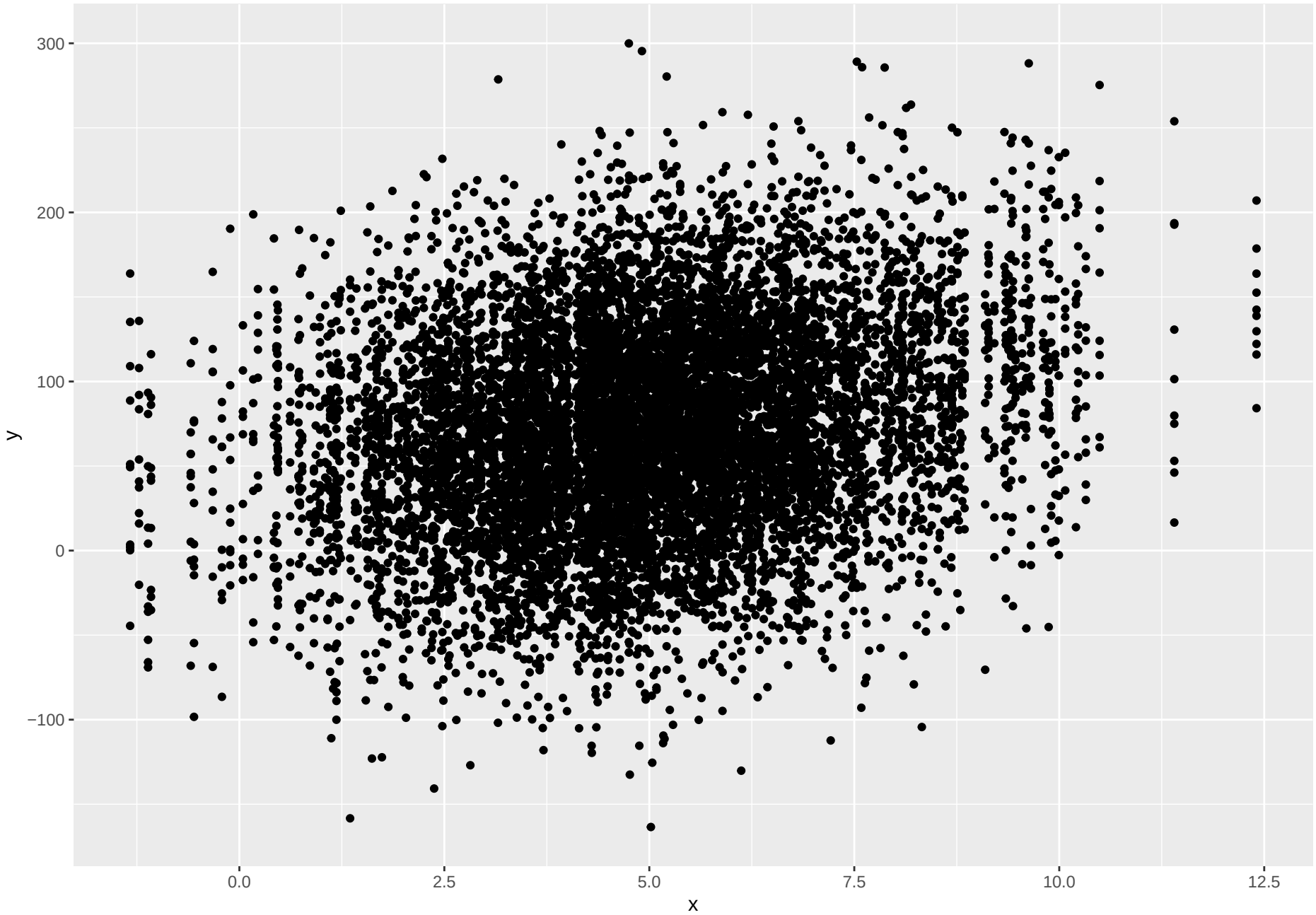
```
set.seed(2020)
x<- rnorm(1000, 5, 2)
y<-1.5*x+.5*x^2+ rnorm(10000,50,60)
df<-data.frame(x,y)
head(df)
```

```
##           x           y
## 1 5.7539442  75.4638782
## 2 5.6030967   0.3770096
## 3 2.8039537  49.7011299
## 4 2.7391882  45.4207398
## 5 -0.5930686  -6.0055724
## 6 6.4411470 102.0343620
```

Scatterplot

- Grafiquemos los puntos

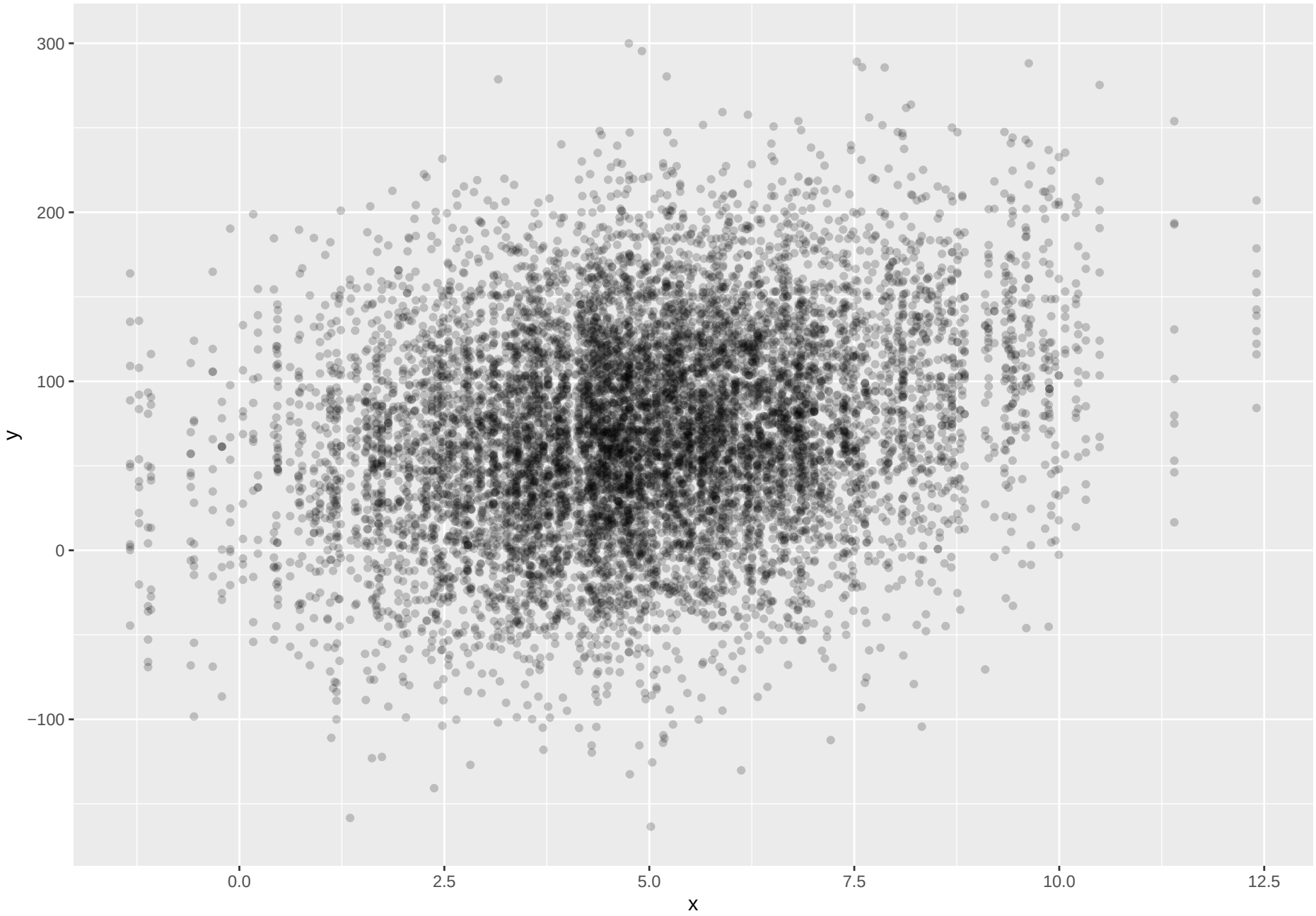
```
ggplot(df, aes(x,y))+
  geom_point()
```


300
200
100
0
−100
y
0.0
2.5
5.0
7.5
10.0
12.5
x

alpha=

- Al tener muchos datos, nuestro scatterplot se ve como una fea mancha negra
- Podemos especificar la transparencia, para que no se vea tan empalmado

```
ggplot(df, aes(x,y))+
  geom_point(alpha=.2)
```


300
200
100
0
−100
y
0.0
2.5
5.0
7.5
10.0
12.5
x

geom_jitter()

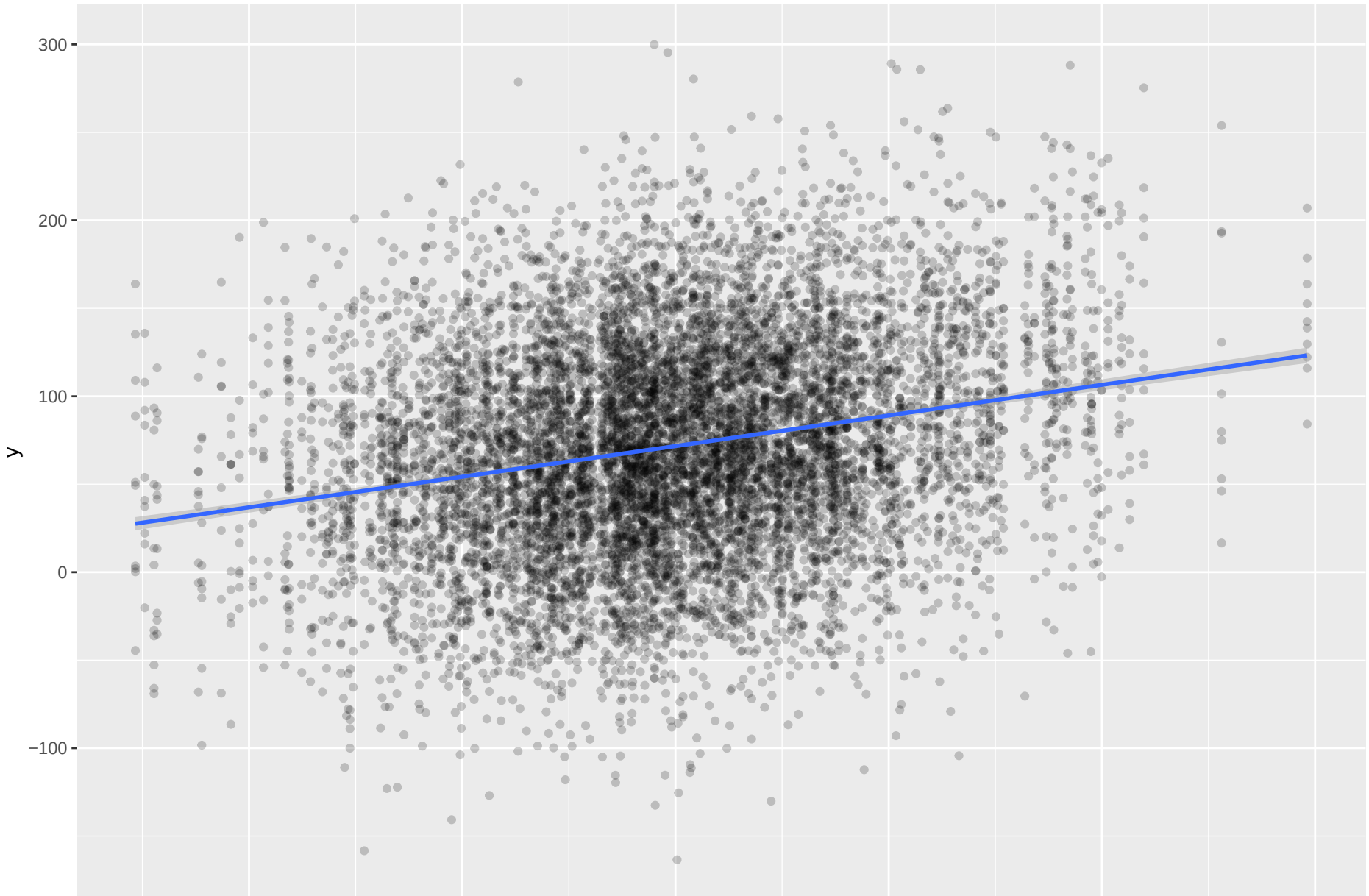
- Con el mismo propósito, podemos usar geom_jtter() cuando tenemos muchas observaciones juntas en una región pequeña de la gráfica. Muy útil para variables categóricas, aunque en este caso no es tan necesario

geom_smooth()

- Podemos añadir una línea de regresión fácilemente con **geom_smooth()**
- Automáticamente añade los intervalos de confianza
- Por default, interpreta que es la regresión de y sobre x
- Añadimos que el método es "lm" por linear model

```
ggplot(df, aes(x,y))+
  geom_point(alpha=.2)+
  geom_smooth(method='lm')
```

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

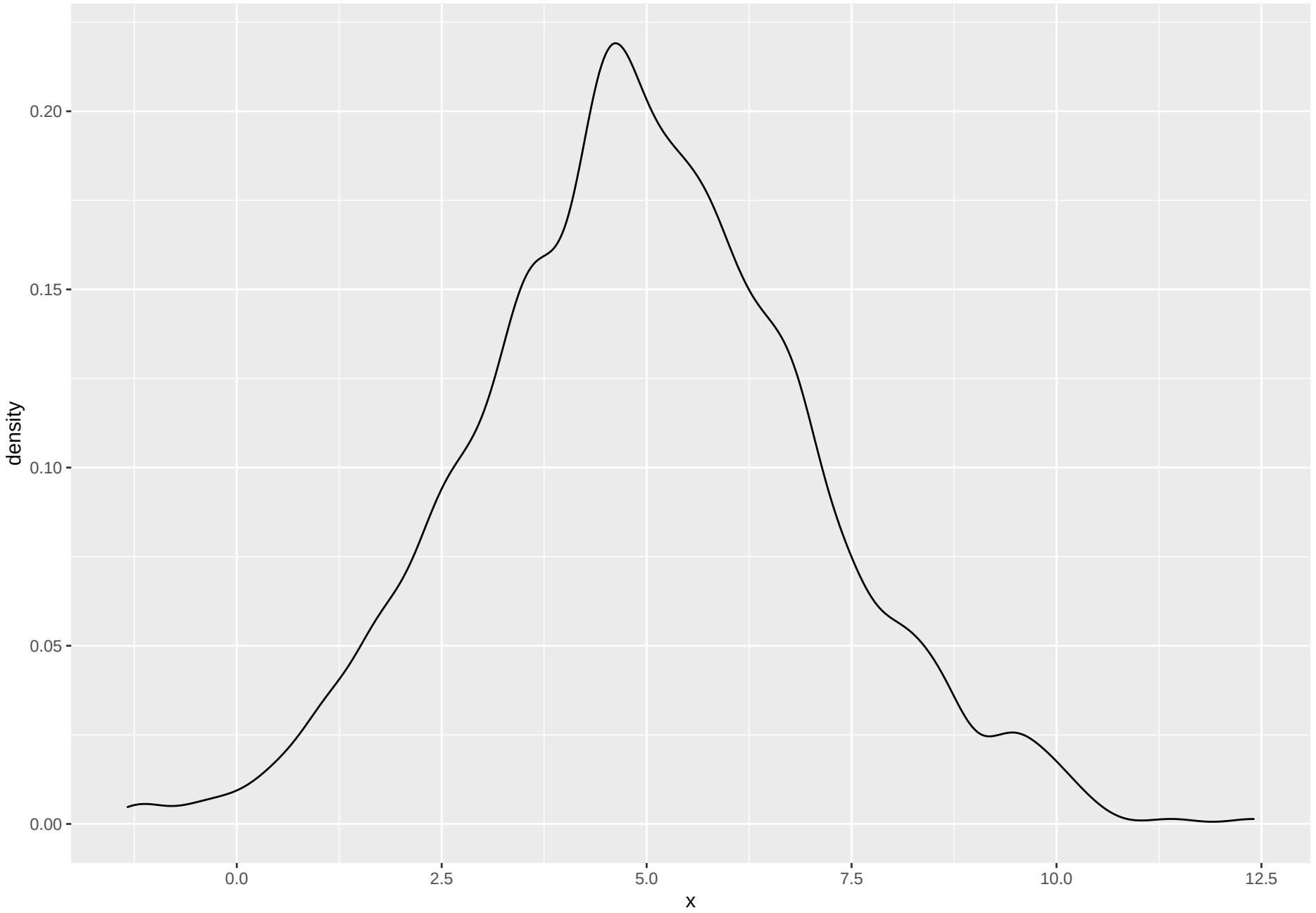


Los intervalos de confiana casi no se ven porque tenemos muchas observaciones

Densidad

- usemos geom_density() para visualizar la densidad
- Obvio, solo necesitamos una variable, pero que esté en un df

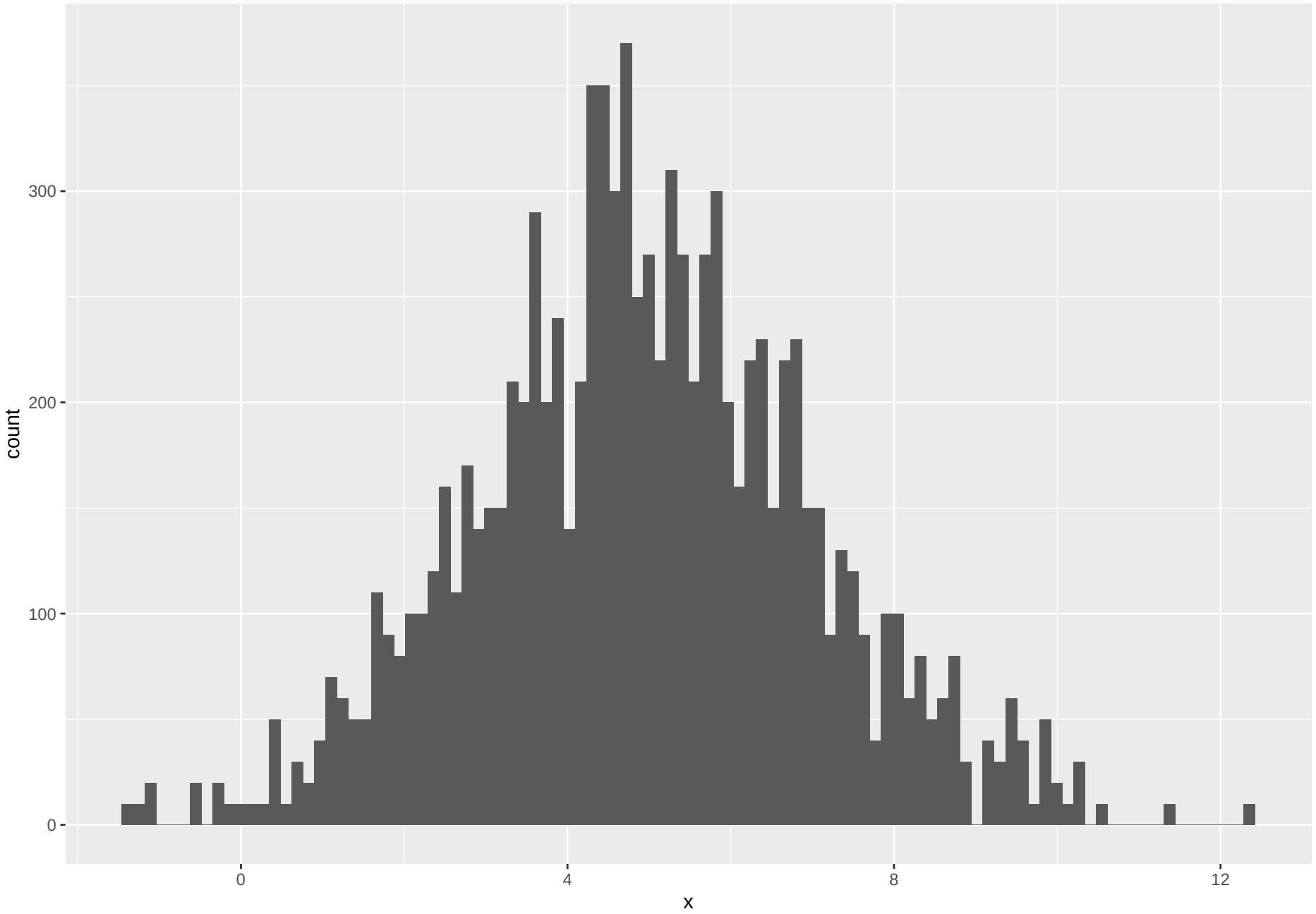
```
ggplot(df)+
  geom_density(aes(x))
```


0.20
0.15
0.10
0.05
0.00
density
0.0
2.5
5.0
7.5
10.0
12.5
x

Histograma

- Análogamente, podemos hacer un histograma
- con bins= podemos controlar las marcas de clases

```
ggplot(df)+
  geom_histogram(aes(x), bins=100)
```


300
200
100
0
count
0
4
8
12
x

Section 4

Infinita personalización

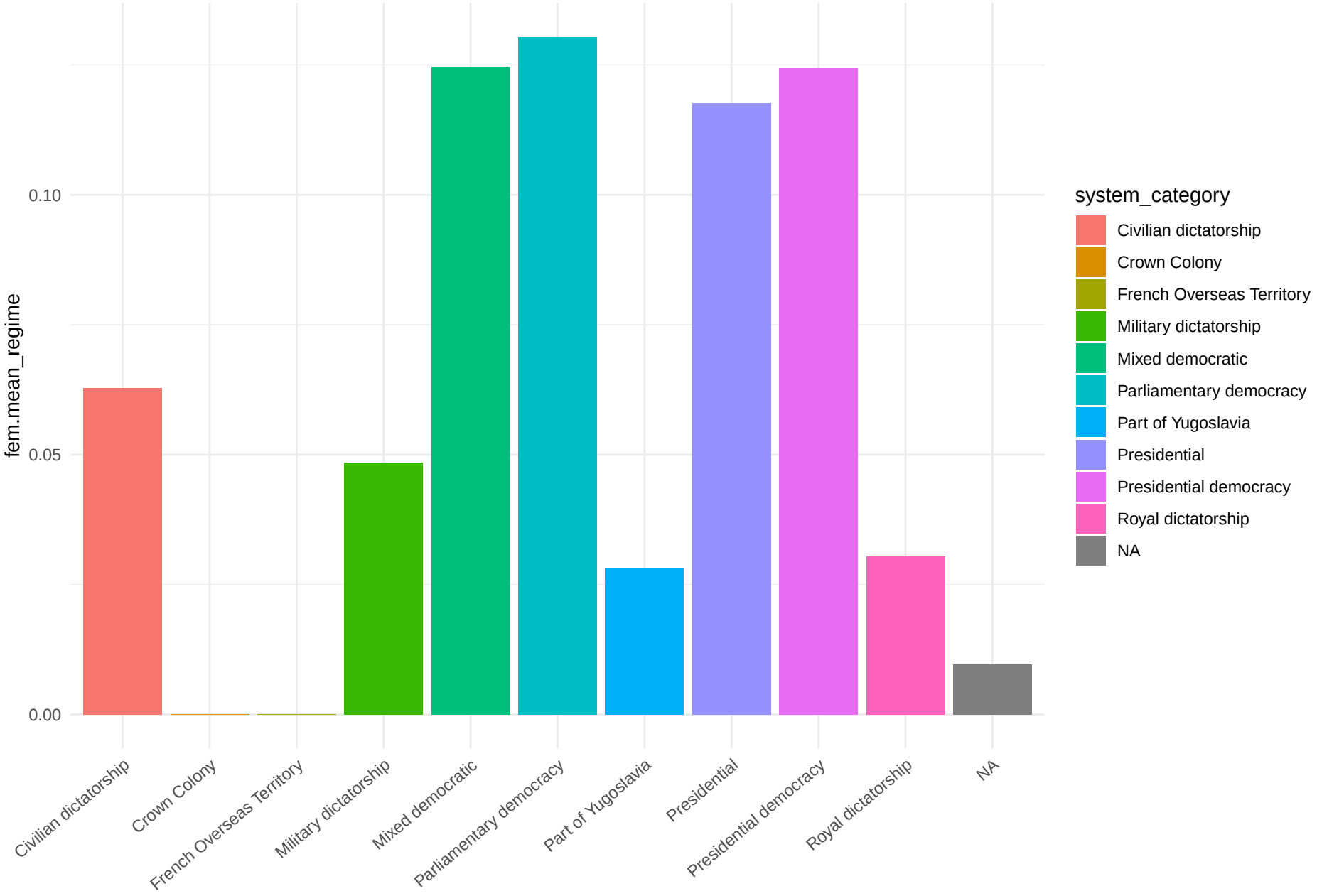
Lo básico

- Hasta aquí, lo básico de ggplot.
- Las posibilidades de personalización son infinitas
- Basta buscar en internet

- Por ejemplo, las letras están empalmadas.
- Busqué en internet cómo cambiar el ángulo
- salió que tenía que añadir

theme(axis.text.x = element_text(angle=40, hjust=1), panel.grid.major.y = element_line())

```
ggplot(min_regime,
       aes(system_category, fem.mean_regime,
           fill=system_category))+
  geom_bar(stat = 'identity')+
  xlab('')+
  theme_minimal()+
  theme(axis.text.x = element_text(angle=40, hjust=1),
        panel.grid.major.y = element_line())
```


fem.mean_regime
0.10
0.05
0.00
Civilian dictatorship
Crown Colony
French Overseas Territory
Military dictatorship
Mixed democratic
Parliamentary democracy
Part of Yugoslavia
Presidential
Presidential democracy
Royal dictatorship
NA
system_category
Civilian dictatorship
Crown Colony
French Overseas Territory
Military dictatorship
Mixed democratic
Parliamentary democracy
Part of Yugoslavia
Presidential
Presidential democracy
Royal dictatorship
NA

Section 5

ggsave()

ggsave()

- Los gráficos de ggplot() pueden guardarse como cualquier objeto
- Podemos guardar el nuestra gráfica generada como un archivo de formato de nuestra preferencia con ggsave()
- Ejecutarlo una y otra vez no genera un nuevo archivo, sino que lo sobreescribe, en caso de no haber cambiado
- Podemos modificar tamaño o escala

```
ggsave(
  filename, # nombre para guardar
  plot = last_plot(), # grafica que va a guardar (ultima pred
  device = NULL, # formato
  path = NULL, # donde lo va a guardar
)
```