

Fastbook 06

Visual Analytics

Gráficos de barras



06. Gráficos de barras

En este fastbook vamos a recorrer una serie de visualizaciones muy utilizadas en los medios de comunicación. Hablamos de los **gráficos de barras** y sus derivados, que permiten realizar comparaciones numéricas entre variables categóricas (y sus niveles). Comenzaremos por el protagonista de este episodio, pero veremos alternativas modernas como el **gráfico de piruleta** e, incluso, haremos una parada en los gráficos de tartas.

De nuevo utilizaremos R. Te recuerdo que lo idóneo es que piques el código y pruebes todo lo posible. ¡Equivocarse y experimentar es parte del aprendizaje también!

Autor: Daniel Pegalajar Luque

[Gráfico de barras: descripción](#)

[Gráfico de barras en R](#)

[Conclusiones](#)

[Bibliografía](#)

Gráfico de barras: descripción

X Edix Educación

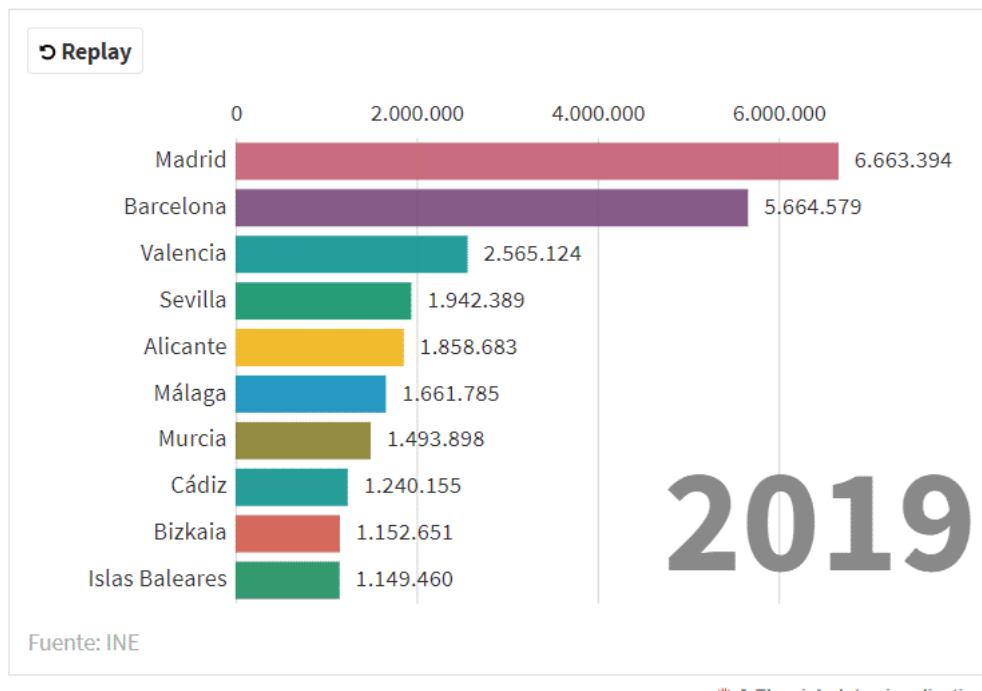
Un **gráfico de barras** (o de columnas) es una de las opciones más clásicas en el mundo de la visualización.

Ya sea con las barras en posición horizontal o vertical (lo habitual), se utiliza con el objetivo de realizar **comparaciones numéricas a lo largo de diferentes categorías** de una variable dada. Por tanto, uno de los ejes mostrará las distintas categorías o niveles y el otro eje representará el conteo de los casos de cada opción.

La principal diferencia con los **histogramas** es que el tipo de gráfico protagonista de este tema no sirve para mostrar el desarrollo de una variable continua, ya que aquí no especificamos un ancho para la barra, sino que **las barras representan subgrupos de observaciones** (categorías), y en este caso, tampoco se altera el ancho de banda (¿recuerdas los bins y su importancia?).

Una de sus mayores flaquezas, como viene siendo habitual en los gráficos clásicos, se manifiesta a la hora de representar un gran número de categorías, ya que la legibilidad de la visualización se ve afectada. No obstante, veremos alternativas para lidiar con este problema.

Para profundizar en este concepto veamos un ejemplo de este gráfico.



Fuente: [Verne](#)

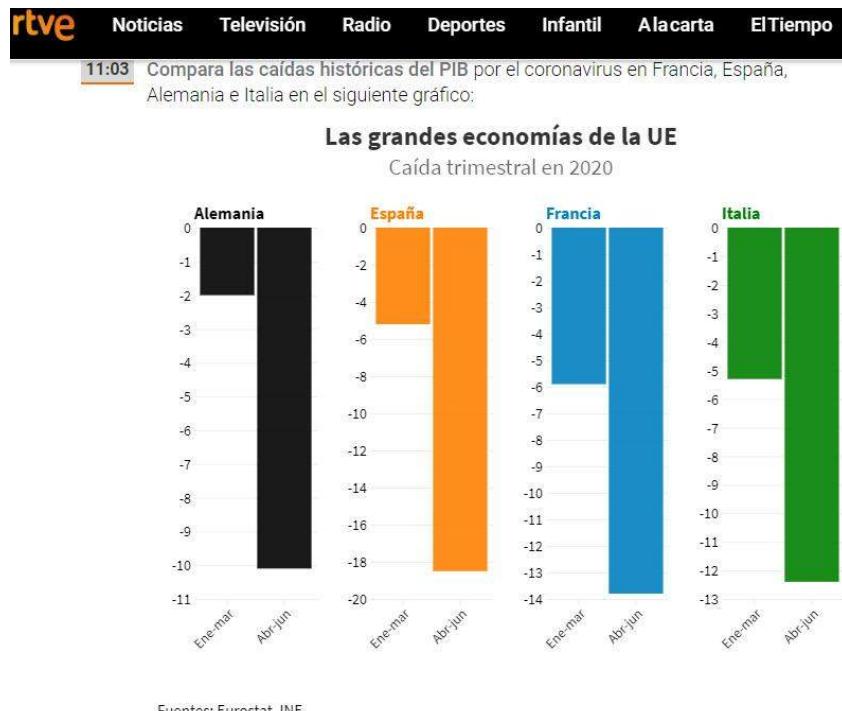
En el ejemplo superior tienes la lista de las **10 provincias más pobladas en España en 2019**. Cada barra en el **eje Y** (eje categórico en posición horizontal) representa una categoría (provincias en este caso); y el **eje X** representa el valor numérico de la variable población que estamos utilizando para comparar. La lectura es muy sencilla: Madrid y Barcelona son las dos provincias más pobladas con diferencia sobre la tercera, Valencia.

Algunos consejos sobre estas visualizaciones:

- Asegúrate de entender bien la **diferencia entre histograma y gráfico de barras**.
- Si las **etiquetas de tus categorías son muy largas**, utiliza la posición horizontal para una mejor lectura.
- Ordenar de forma descendente los resultados** suele facilitar las comparaciones y mejora el resultado visual.

- Asegúrate de que tu eje numérico o de conteo empiece en 0 (a no ser que tengas buenos motivos para truncarlo). De no ser así, resaltarás las diferencias no significativas y se te podría acusar de intentar ‘manipular los aprendizajes’, y no queremos eso, ¿verdad?
- Dotar de **efecto 3D** a las barras es igual a un suspenso en la asignatura. Como recordarás de los primeros fastbooks, buscamos en esencia visualizaciones sencillas de leer y el efecto 3D no contribuye a ello.

Por último, grábate en la cabeza que con este gráfico puedes visualizar rápidamente qué grupos son los más comunes o destacados, además de cuantificar cómo son las diferencias entre dichos grupos. Es un **gráfico muy popular y utilizado en los medios** (eso no significa que se use bien), por lo que ahora que lo conoces con mayor detalle comenzarás a identificarlo en ocasiones, quizás hasta encuentres casos en los que se usan erróneamente o de forma ‘malintencionada’.



Fuente: [Twitter](#) (ya corregido).

Es hora de ponerse manos a la obra y descubrir cómo generar estos gráficos en R y Python. ¡Vamos a ello!

Gráfico de barras en R

X Edix Educación

Como ya es tradición con R, cargamos algunas opciones para trabajar más ágilmente. Volvemos a las opciones que arrastrábamos de los temas anteriores ya que buscamos realizar comparaciones numéricas:

```
# Desactivamos la notación científica. ¿A quién le gusta ver en sus gráficos números como
# 1e25?
options(scipen = 999)

# Cargamos las librerías necesarias para pintar
library(ggplot2) # Nuestra biblia a partir de ahora
library(scales) # Nos ayudará a mejorar el aspecto de nuestros gráficos

library(tidyverse) # Necesario si queremos realizar algún tratamiento en los datos

# Establecemos un tema por defecto para nuestros gráficos
# Personalmente soy fanático de theme_bw(), es el tema clásico 'dark-on-light'
# ggplot ofrece un listado de temas completos que puedes aprovechar. Echa un vistazo:
# https://ggplot2.tidyverse.org/reference/ggtheme.html

# Hay gente que realiza sus propios temas, generando auténticas obras de arte, ¿te atreves?
theme_set(theme_bw())

# Cargamos los datos que vamos a utilizar

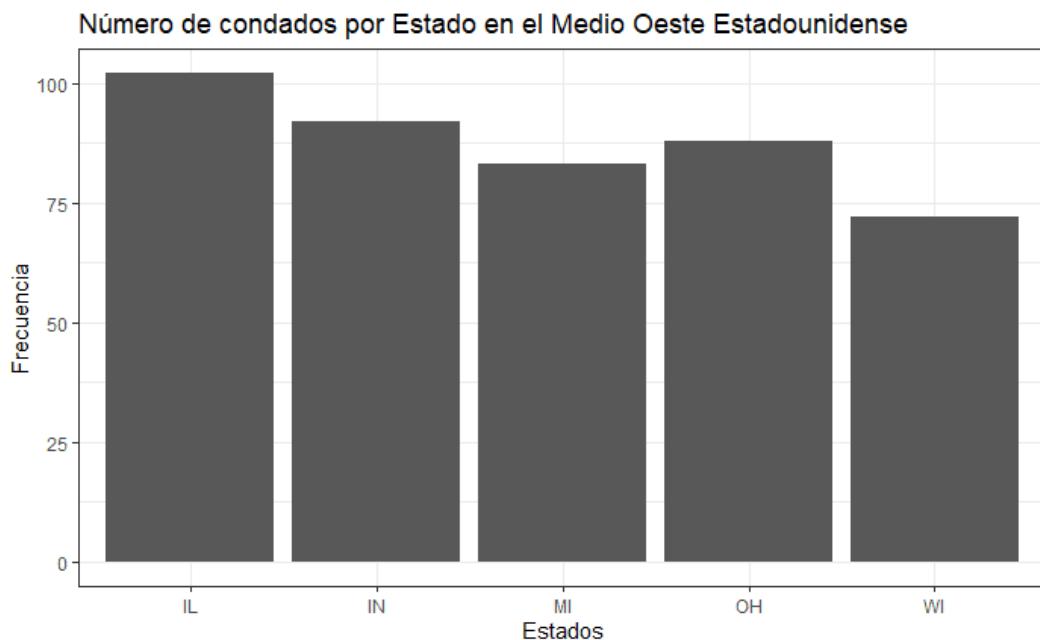
data("diamonds", package = "ggplot2")
data("txhousing", package = "ggplot2")
data("midwest", package = "ggplot2")

## Librerías con datasets para series temporales
library(ukbabynames)
library(babynames)|
```

Vamos a centrarnos en utilizar un conjunto de datos de [midwest](#) que recoge información sobre diversos condados del Medio Oeste estadounidense.

Este tipo de visualización solo necesita la especificación de una **variable categórica** para mapear el conteo de los diferentes niveles (en este caso, estados) en los datos. Por defecto, el geom utiliza la función `stat_count()` para ofrecernos en el eje Y la **frecuencia o conteo** de observaciones registradas en cada estado.

Los estados con más condados son Illinois e Indiana; Wisconsin el que menos.



Podemos jugar con esta opción y mostrar los resultados mediante una **proporción**:

```
## Tu primer gráfico de barras (proporciones)

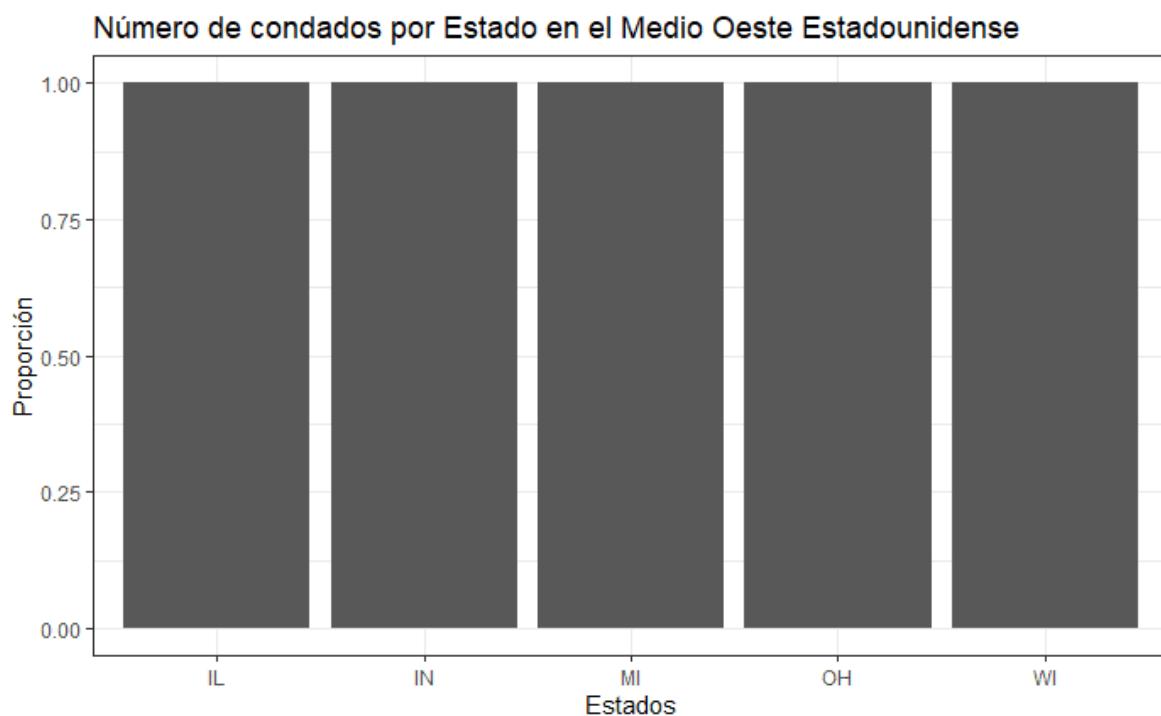
gg <- ggplot(midwest, aes(x = state)) +
  geom_bar(aes( y = ..prop.., group = 1)) +
  scale_y_continuous(labels = percent) +
  labs(title = "Número de condados por Estado en el Medio Oeste Estadounidense",
       x = "Estados",
       y = "Proporción")

gg
```

Si deseamos mostrar frecuencias relativas (muy útil cuando los valores absolutos no son lo importante de la historia que queremos contar) usaremos el argumento `..prop..` dentro del `geom`. Un poco extraño respecto a lo visto hasta ahora, ¿no?

La explicación es sencilla: la función `stat_count()` mencionada anteriormente genera variables temporales para representar los datos y ahorrarte trabajo extra. Para evitar problemas utiliza dos puntos antes del argumento. El **objetivo** no es otro que no confundir este método con variables que puedan existir en nuestro conjunto de datos con ese nombre.

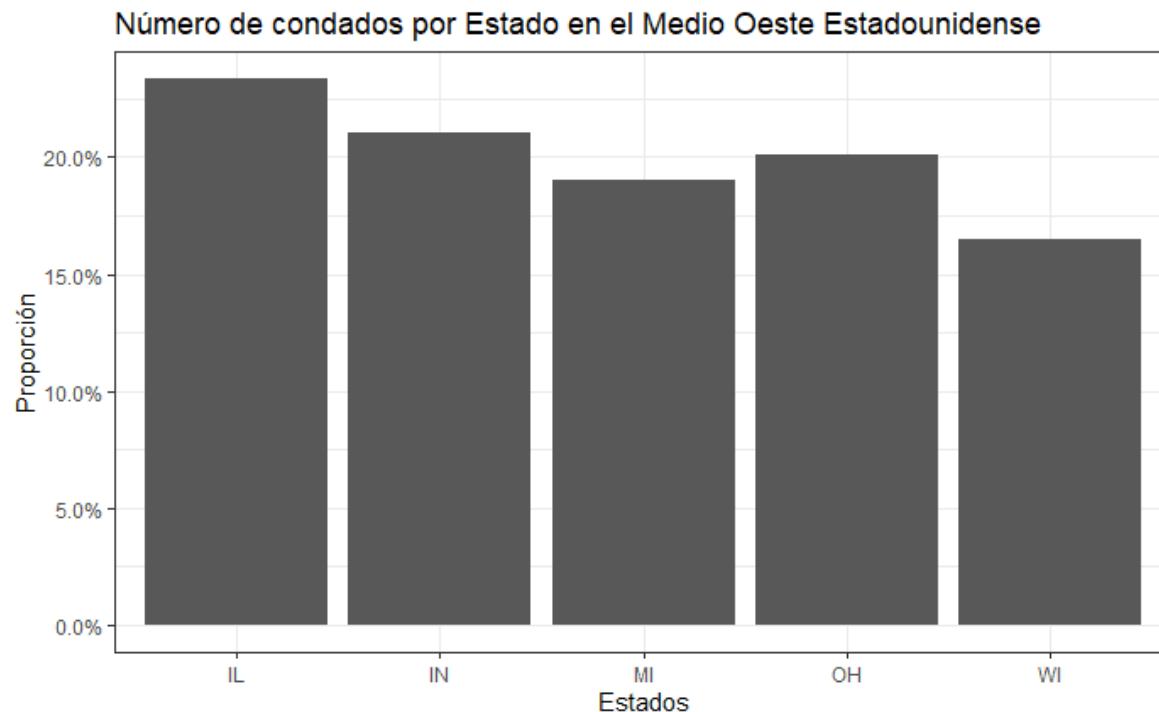
De todas formas, con esto el resultado aún no es definitivo:



Si te fijas, todos los estados ahora se expresan en porcentaje, pero todos suman 1 (algo poco útil). Esto es debido a que la función de ggplot está calculando estas frecuencias relativas para cada estado; por tanto, es necesario especificarle que haga los cálculos con todo el conjunto de datos. Para ello utiliza el siguiente código:

```
## Dando vida con un poco de color
gg ← ggplot(midwest, aes(x = state, fill = state)) +
  geom_bar() +
  labs(title = "Número de condados por Estado en el Medio Oeste Estadounidense",
       x = "Estados",
       y = "Frecuencia") +
  theme(legend.position = "none")
gg
```

Basta con incluir el argumento **group** dentro del geom con un “1”. Esto es un valor *dummy* que indica a ggplot que no hay otro grupo que todo el dataset:

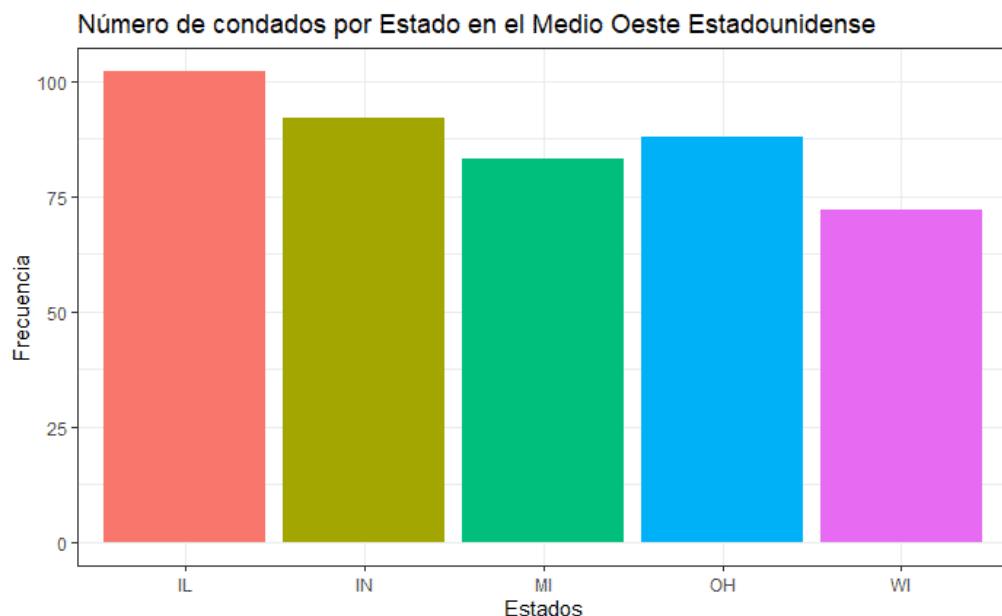


Ahora es más útil obtener aprendizajes de forma relativa: Illinois posee un 23% de los condados frente al 17% de Wisconsin.

Por otro lado, recuerda que puedes dotar de mejor aspecto, ‘mayor atractivo’, a tu visualización mediante una paleta de colores. Para ello utiliza el argumento **fill** con la variable que dará el color, en este caso, a los diferentes estados.

```
## Dando vida con un poco de color
gg <- ggplot(midwest, aes(x = state, fill = state)) +
  geom_bar() +
  labs(title = "Número de condados por Estado en el Medio Oeste Estadounidense",
       x = "Estados",
       y = "Frecuencia") +
  theme(legend.position = "none")
gg
```

No te confundas y utilices el argumento **col/colour**, o lo único que colorearás será el borde de cada barra...

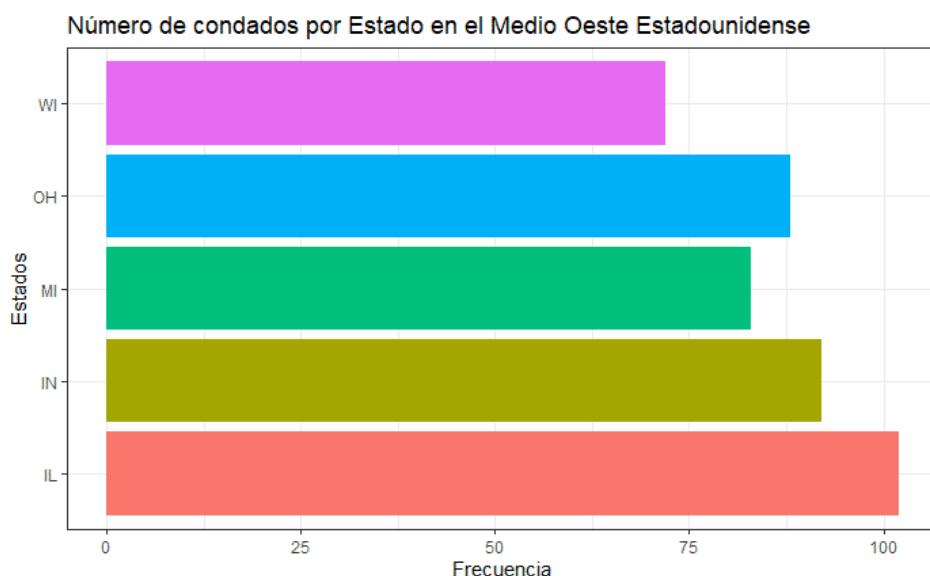


Como verás en la imagen, desactivamos la aparición de una leyenda, dado que va a aportar poca información, puesto que ya tiene los estados en el eje X.

Otro apunte: al igual que en muchos geoms estudiados en la asignatura, **tú decides la orientación del gráfico en base al eje utilizado**, y si quieres voltear las barras, basta con que cambies al eje Y:

```
## Dando vida con un poco de color
gg <- ggplot(midwest, aes(y = state, fill = state)) +
  geom_bar() +
  labs(title = "Número de condados por Estado en el Medio Oeste Estadounidense",
       y = "Estados",
       x = "Frecuencia") +
  theme(legend.position = "none")
gg
```

En este tipo de gráficos duelen los ojos al ver **que las barras no están ordenadas de manera descendente**.



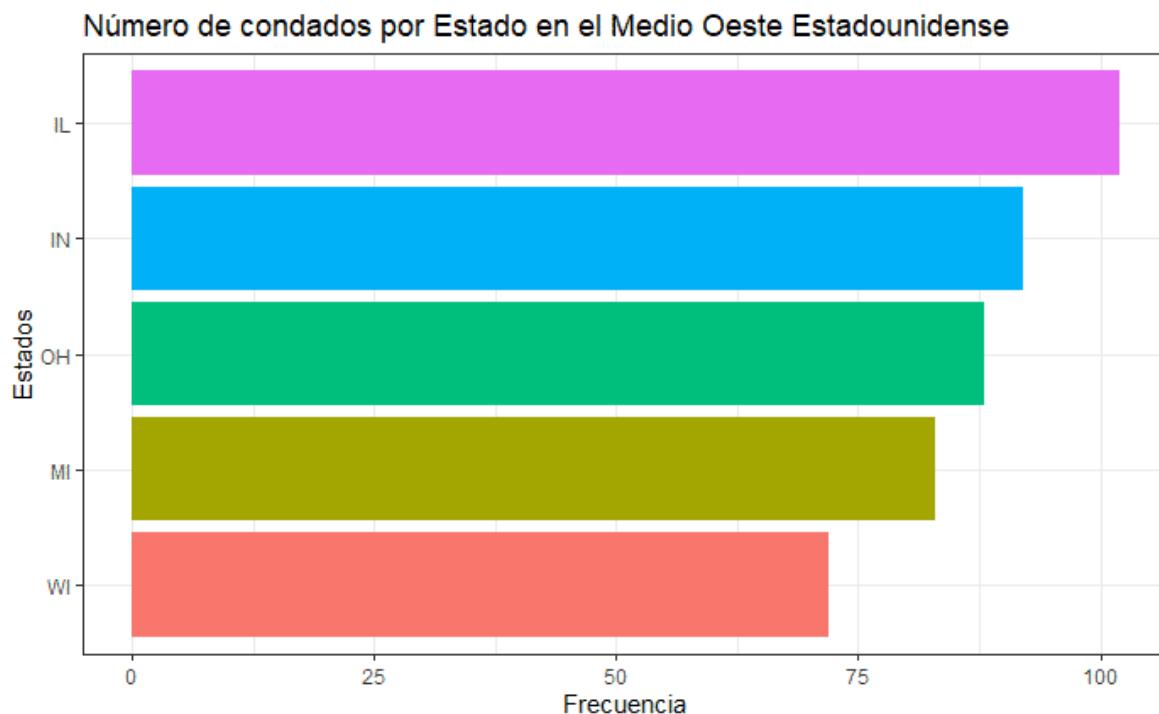
Para solventar este problema, puedes utilizar lo siguiente (también te vale para la otra visión):

```
## Generando un orden adecuado
orden <- midwest %>% count(state) %>% arrange(n) %>% pull(state)

gg <- midwest %>%
  mutate(state = factor(state, orden)) %>%
  ggplot(aes(y = state, fill = state)) +
  geom_bar() +
  labs(title = "Número de condados por Estado en el Medio Oeste Estadounidense",
       y = "Estados",
       x = "Frecuencia") +
  theme(legend.position = "none")

gg
```

Podemos generar un orden utilizando **dplyr**. Basta con hacer un conteo, ordenar por ese valor y extraer las etiquetas en el orden obtenido. Tras este proceso, convertiremos la variable state en un factor cuyos niveles serán los determinados en ese ordenamiento. A continuación tienes el resultado:



Otra posibilidad es que hagamos uso de la maravillosa librería [forcats](#) incluida en el [tidyverse](#):

```
## Generando un orden adecuado
library(forcats) ## Librería específica para el tratamiento de variables categóricas
gg <- midwest %>%
  mutate(state = fct_infreq(state)) %>%
  ggplot(aes(y = (state), fill = state)) +
  geom_bar() +
  scale_y_discrete(limits = rev) +
  labs(title = "Número de condados por Estado en el Medio Oeste Estadounidense",
       y = "Estados",
       x = "Frecuencia") +
  theme(legend.position = "none")
gg
```

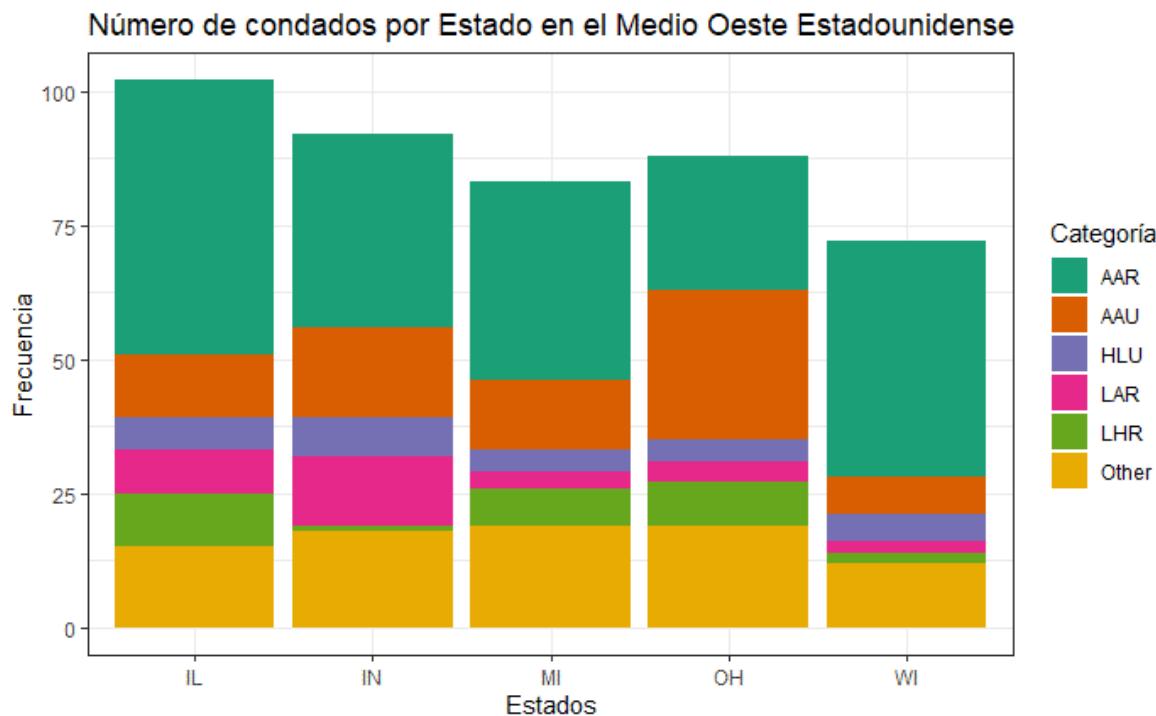
Basta con utilizar su función `fct_infreq()` para generar la variable factor `state` con el orden correcto. En esta versión, tenemos que especificar el orden reverso en la escala para no obtener un efecto contrario al que esperamos (desactiva la línea de `scale_y_discrete()` en el anterior bloque de código para entender a qué me refiero).

Una forma de sacar mayor potencial al coloreado en este geom consiste en **cruzar la información de dos variables categóricas**. Esto es el equivalente a una tabla de frecuencias o proporciones que puede permitir la detección de patrones muy fácilmente.

```
## Potenciando el uso del color en geom_bar
midwest <- midwest %>%
  mutate(category_new = fct_lump_min(category, 25))
gg <- ggplot(midwest, aes(x = state, fill = category_new)) +
  geom_bar() +
  scale_fill_brewer(palette = "Dark2") +
  labs(title = "Número de condados por Estado en el Medio Oeste Estadounidense",
       x = "Estados",
       y = "Frecuencia",
       fill = "Categoría")
gg
```

Comenzamos utilizando la función `fct_lump_min()` que sirve para colapsar variables categóricas. En nuestro caso, la variable `category` posee 16 niveles, muchos de ellos con un solo registro.

Vamos a utilizar un umbral y todos aquellos niveles que no lo superen se irán a una categoría 'Other'.



El resultado permite comparar no solo el conteo de los condados a nivel de estado, sino también analizar los patrones de la nueva variable entre estados. Por ejemplo, la categoría AAU abunda en Ohio mientras que la categoría LAR es más común en Illinois e Indiana.

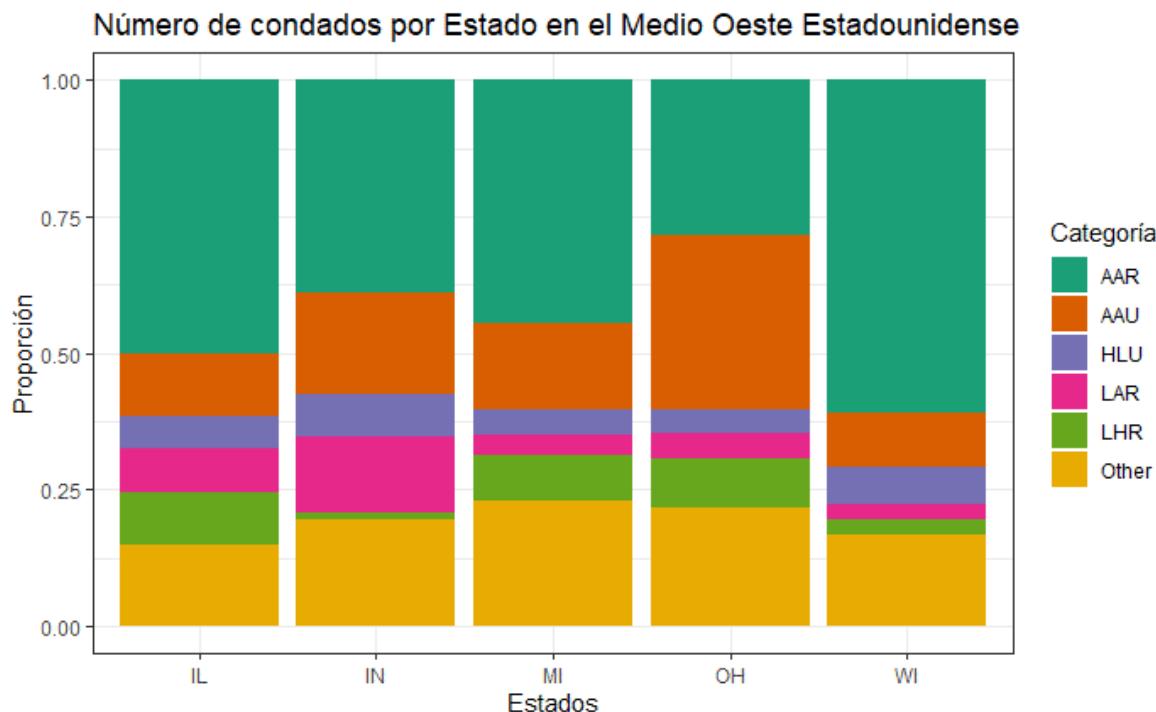
Por defecto, al utilizar esta modalidad ggplot genera una visualización con **barras apiladas** (stacked) con conteo interno. Si recuerdas, para algunos lectores puede ser complicado comparar cuando la escala está desajustada. Para solucionar este problema, podemos recurrir al argumento **position** del propio geom:

```
## Igualando las escalas
midwest <- midwest %>%
  mutate(category_new = fct_lump_min(category, 25))

gg <- ggplot(midwest, aes(x = state, fill = category_new)) +
  geom_bar(position = "fill") +
  scale_fill_brewer(palette = "Dark2") +
  labs(title = "Número de condados por Estado en el Medio Oeste Estadounidense",
       x = "Estados",
       y = "Proporción",
       fill = "Categoría")

gg
```

Ahora todas las barras son de la misma altura, facilitando la comparación total a través de las diferentes combinaciones de grupos.



Por contra, perdemos la capacidad de ver el tamaño relativo de cada corte o estado sobre el total del conjunto de datos (puede ser una categoría importante en un estado, ¿pero cómo de importante es esa conclusión sobre el total del conjunto de datos?).

Para esos casos, tenemos la siguiente opción:

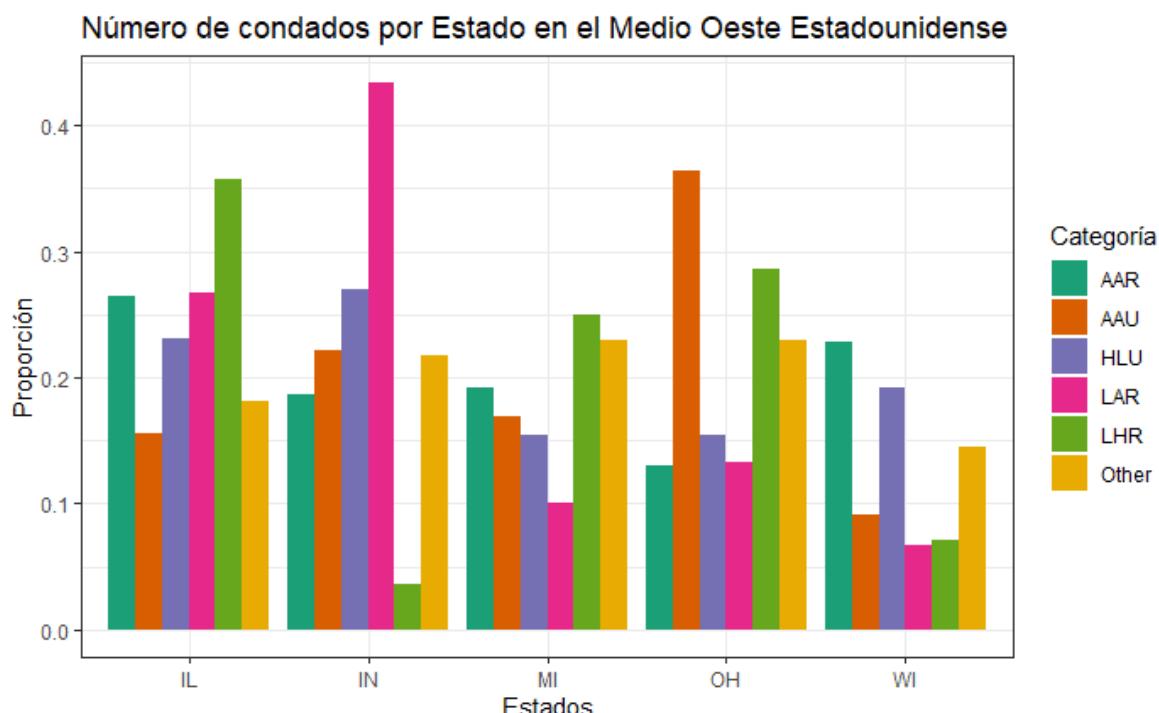
```
## Cuando no quieras perder las referencias totales

gg <- ggplot(midwest, aes(x = state, fill = category_new)) +
  geom_bar(position = "dodge", aes(y = ..prop.., group = category_new)) +
  scale_fill_brewer(palette = "Dark2") +
  labs(title = "Número de condados por Estado en el Medio Oeste Estadounidense",
       x = "Estados",
       y = "Proporción",
       fill = "Categoría")

gg
```

Si lo intentas comprobar, en cada estado las barras no sumarán uno, ya que la proporción está agrupada a nivel de categoría. Por tanto, para una **categoría en particular la suma de uno se obtiene a lo largo de todos los estados.**

Esta visión nos permite obtener aprendizajes interesantes y más fácilmente como que “Buena parte de la categoría LAR se concentra en Indiana” o que “ocurre lo mismo con la categoría AAU en Ohio”.



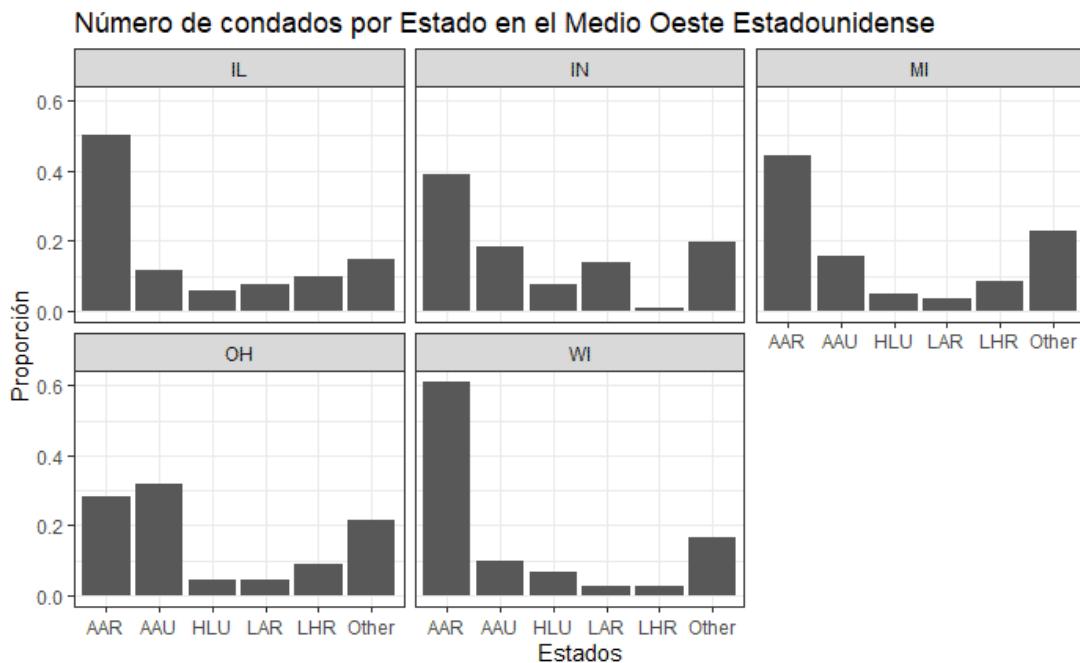
Por último, recuerda que siempre dispones de los **facets** si quieras ‘aclarar’ la lectura de un gráfico cuando se dificulta:

```
## Nunca olvides los facets

gg <- ggplot(midwest, aes(x = category_new, fill = category_new)) +
  geom_bar(position = "dodge", aes(y = ..prop.., group = state)) +
  scale_fill_brewer(palette = "Dark2") +
  facet_wrap(~state) +
  labs(title = "Número de condados por Estado en el Medio Oeste Estadounidense",
       x = "Estados",
       y = "Proporción",
       fill = "Categoría")

gg
```

Ahora, las proporciones suman uno en cada cuadro, pero la separación de esta forma permite no producir demasiadas barras juntas en cada estado.



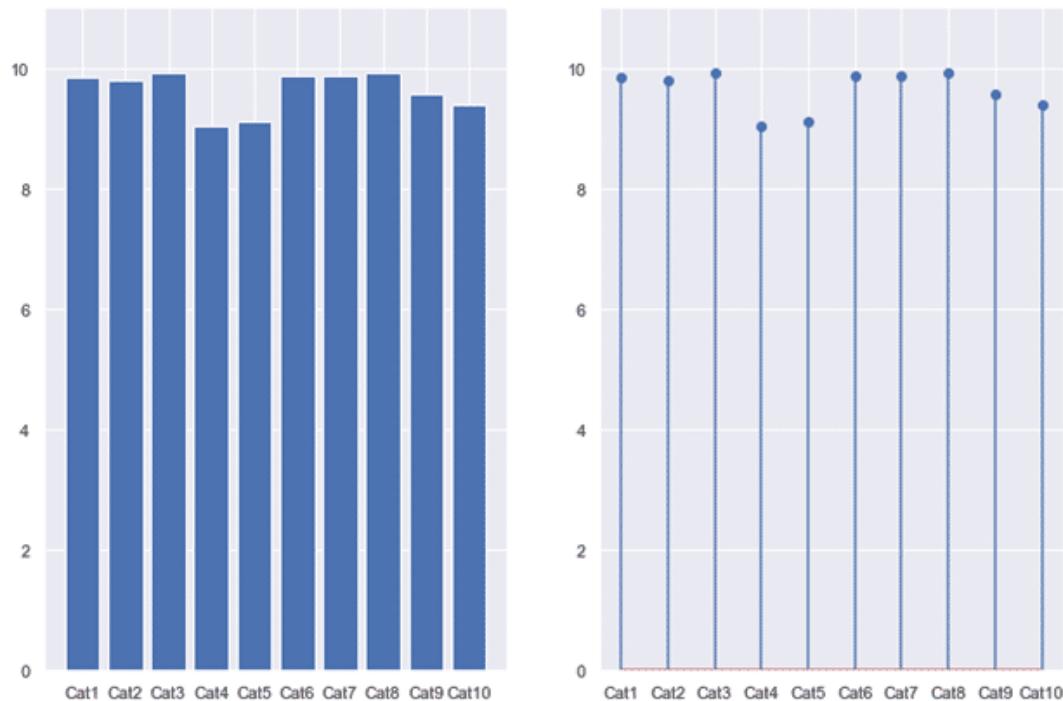
Por último, recuerda que no hay una opción ‘mejor’, cada una te puede venir bien en diferentes momentos. Utiliza tu criterio e intenta reproducir lo que quieras transmitir.

Lollipop chart o gráfico piruleta: tu opción con numerosas categorías

Los gráficos de **piruleta** (cualquier queja por los nombres de los gráficos dedícasela a los anglosajones, ¿eh?) son variaciones muy interesantes para los gráficos de barras tradicionales.

Cuando queremos **visualizar variables con un gran número de categorías**, este tipo de gráfico puede ser más conveniente, ya que es más atractivo a la hora de leer la información y permite evitar la aglomeración de barras en tu visualización (la superposición de numerosas barras de tamaño similar puede producir un patrón de muaré), evitando así ‘daños visuales’ colaterales a los lectores.

¿Un imprescindible en esta visualización? Que las categorías estén ordenadas; y si no vas a ordenarla, vuelve al gráfico de barras, pues un gráfico piruleta resulta muy complicado de leer sin cumplir ese requisito.



Fuente: [TowardsDataScience](#).

Para conseguir esta bonita visualización en R hay que trabajar un poco más, pero el resultado merece la pena.

En este caso concreto, vamos a crear un conjunto de datos específico para llegar a la visualización objetivo. Cogeremos prestado los nombres de los fabricantes de coches que aparecen en el conjunto de datos “mpg” de ggplot.

```
## Creamos los datos
# Establecemos la semilla aleatoria para que a todos nos salga lo mismo
set.seed(7)

# Utilizamos
data <- data.frame(
  fabricante=mpg %>% pull(manufacturer) %>%
    unique() %>% sort() %>% toupper,
  y=abs(rnorm(15))
)

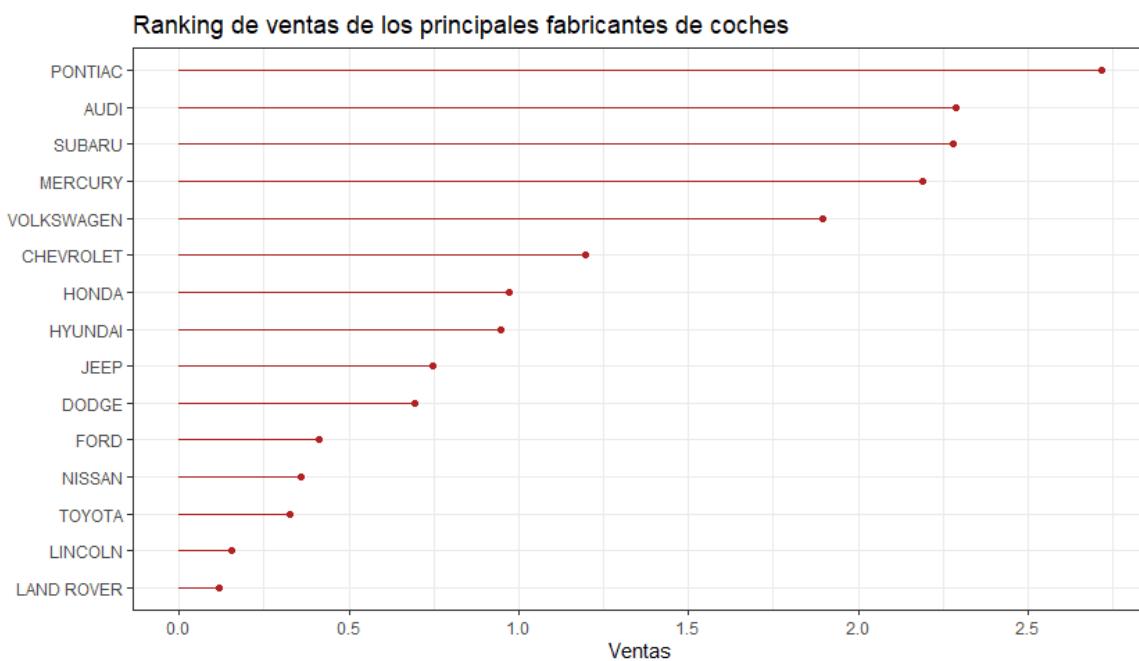
# Reordenamos la información (importante para este tipo de visualización)
data <- data %>%
  arrange(y) %>%
  mutate(fabricante=fct_inorder(fabricante))

# Ploteamos la información
p <- ggplot(data, aes(x=fabricante, y=y)) +
  geom_segment( aes(x=fabricante, xend=fabricante, y=0, yend=y), color = "firebrick") +
  geom_point(color= "firebrick") +
  coord_flip() +
  scale_y_continuous(breaks = pretty_breaks(n = 5)) +
  theme(
    legend.position="none"
  ) +
  xlab("") +
  ylab("Ventas") +
  ggtitle("Ranking de ventas de los principales fabricantes de coches")

p
```

Ordenaremos los resultados, un requisito clave en este tipo de visualización antes de comenzar a pintar. Para llegar al objetivo final, es necesario combinar dos geoms, ya que no existe uno específico para esta visualización: se trata de **geom_segment**, encargado de dibujar el trazo de la piruleta y **geom_point**. Por último, la función **coord_flip()** permite voltear la orientación de tu visualización.

El resultado debería ser el siguiente:



Bonito, ¿no? En este ejemplo ficticio, Pontiac sería el fabricante líder seguido de otros tres competidores muy parejos: Audi, Subary y Mercury.

Ahora imagina que trabajas para el fabricante que posee las marcas Chevrolet y Dodge y quieres enriquecer aún más tu gráfico de cara a una reunión importante. Puedes hacerlo de la siguiente forma:

```
## Creamos los datos
# Establecemos la semilla aleatoria para que a todos nos salga lo mismo
set.seed(7)

# Utilizamos
data <- data.frame(
  fabricante=mpg %>% pull(manufacturer) %>%
  unique() %>% sort() %>% toupper,
  y=abs(rnorm(15))
)

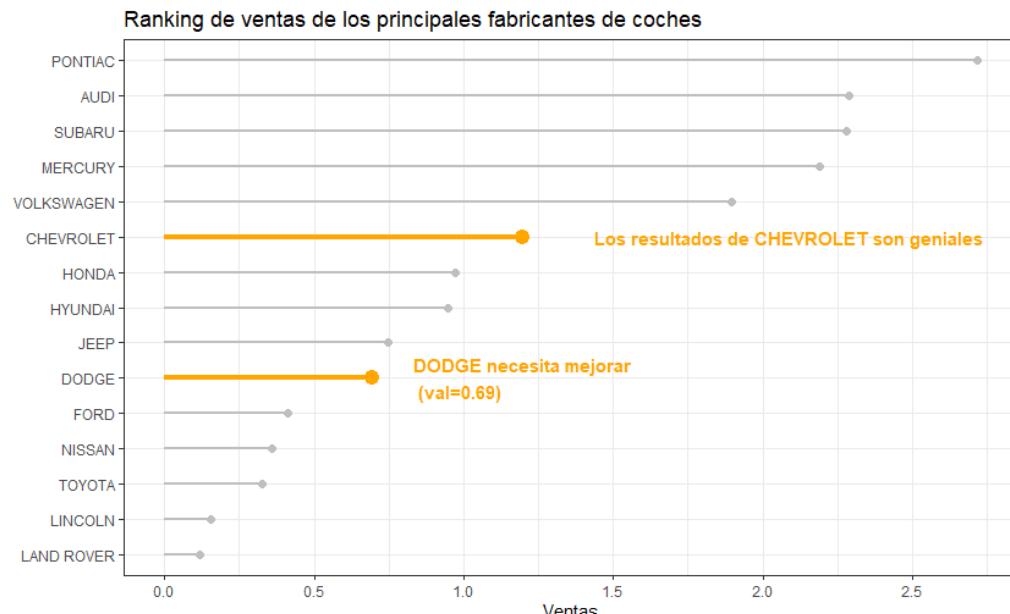
# Reordenamos la información (importante para este tipo de visualización)
data <- data %>%
  arrange(y) %>%
  mutate(fabricante=fct_inorder(fabricante))

# Ploteamos la información
p <- ggplot(data, aes(x=fabricante, y=y)) +
  geom_segment(aes(x=fabricante, xend=fabricante, yend=y),
               color = ifelse(data$fabricante %in% c("CHEVROLET", "DODGE"), "orange", "grey"),
               size = ifelse(data$fabricante %in% c("CHEVROLET", "DODGE"), 1.5, 1)) +
  geom_point(color = ifelse(data$fabricante %in% c("CHEVROLET", "DODGE"), "orange", "grey"),
             size = ifelse(data$fabricante %in% c("CHEVROLET", "DODGE"), 4, 2)) +
  coord_flip() +
  scale_y_continuous(breaks = pretty_breaks(n = 5)) +
  theme(
    legend.position="none"
  ) +
  xlab("") +
  ylab("Ventas") +
  ggtitle("Ranking de ventas de los principales fabricantes de coches")

# Añadimos anotaciones
p + annotate("text", x = grep("CHEVROLET", data$fabricante), y = data$y[which(data$fabricante == "CHEVROLET")]*1.2,
             label="Los resultados de CHEVROLET son geniales",
             color="orange", size=4, angle=0, fontface="bold", hjust=0) +
  annotate("text", x = grep("DODGE", data$fabricante), y = data$y[which(data$fabricante == "DODGE")]*1.2,
             label = paste("DODGE necesita mejorar\n(val=", data$y[which(data$fabricante == "DODGE")]->%
               round(2),")", sep=""),
             color="orange", size=4, angle=0, fontface="bold", hjust=0)
```

p

Intenta leer el código de lo que hemos realizado arriba: es sencillo de entender y puedes probar a modificar utilizando otros fabricantes para comprobar que lo has entendido. El resultado de nuestra visualización definitiva sería el siguiente:



Existe otra variación que potencia aún más si cabe estas visualizaciones, reciben el nombre de **gráficos de puntos de Cleveland** y permiten comparar el valor de dos variables numéricas por grupo.

Para cada entidad o grupo principal se dibuja un punto para cada variable, con colores diferentes. La diferencia entre ellos se resalta utilizando un segmento que los une. Piensa, por ejemplo, en romper una variable y añadirle información sobre el sexo para entender si existen diferencias reseñables respecto al género. Vamos a ver un ejemplo y así entenderemos su potencial:

```
# Creamos el conjunto de datos (Source: EIA 2016)
data = tibble::tribble(
  ~Country, ~Y1990, ~Y2015,
  'Russia', 71.5, 101.4,
  'Canada', 74.4, 102.9,
  'Other non-OECD Europe/Eurasia', 60.9, 135.2,
  'South Korea', 127, 136.2,
  'China', 58.5, 137.1,
  'Middle East', 170.9, 158.8,
  'United States', 106.8, 169,
  'Australia/New Zealand', 123.6, 170.9,
  'Brazil', 208.5, 199.8,
  'Japan', 181, 216.7,
  'Africa', 185.4, 222,
  'Other non-OECD Asia', 202.7, 236,
  'OECD Europe', 173.8, 239.9,
  'Other non-OECD Americas', 193.1, 242.3,
  'India', 173.8, 260.6,
  'Mexico/Chile', 221.1, 269.8
)

data <- data %>%
  rowwise() %>%
  mutate(media = mean(c_across(Y1990:Y2015))) %>%
  arrange(media) %>%
  mutate(Country=factor(Country, Country))

# Plot
ggplot(data) +
  geom_segment( aes(x=Country, xend=Country, y=Y1990, yend=Y2015), color="grey", size = 2) +
  geom_point( aes(x=Country, y=Y1990), color="firebrick", size=3 ) +
  geom_point( aes(x=Country, y=Y2015), color="limegreen", size=3 ) +
  coord_flip()+
  labs(title = 'Productividad energética en países seleccionados y regiones',
       subtitle = 'Billones de $ GDP por cuatrillón BTU. Rojo: 1990, Verde:2015',
       caption = 'Source: EIA, 2016',
       x = NULL, y = NULL)
```

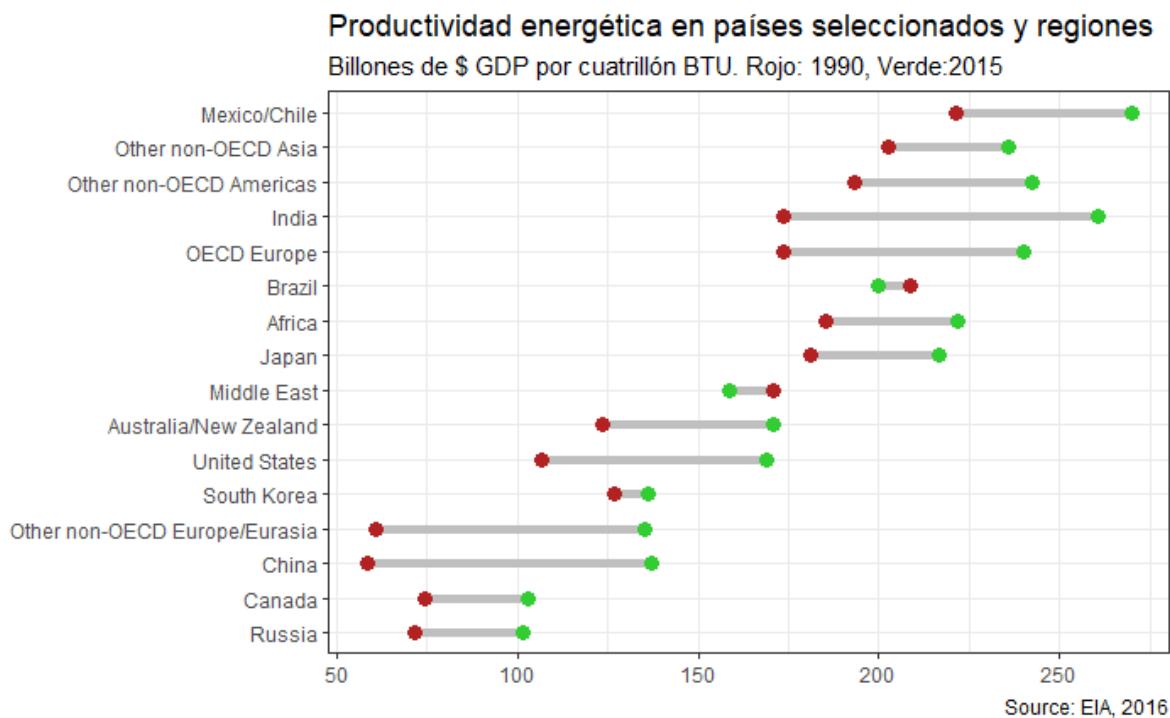
Te dejo el bloque de datos para que no tengas que picarlo a mano:

```
data = tibble::tribble(~Country, ~Y1990, ~Y2015,  
  'Russia', 71.5, 101.4,  
  'Canada', 74.4, 102.9,  
  'Other non-OECD Europe/Eurasia', 60.9, 135.2,  
  'South Korea', 127, 136.2,  
  'China', 58.5, 137.1,  
  'Middle East', 170.9, 158.8,  
  'United States', 106.8, 169,  
  'Australia/New Zealand', 123.6, 170.9,  
  'Brazil', 208.5, 199.8,  
  'Japan', 181, 216.7,  
  'Africa', 185.4, 222,  
  'Other non-OECD Asia', 202.7, 236,  
  'OECD Europe', 173.8, 239.9,  
  'Other non-OECD Americas', 193.1, 242.3,  
  'India', 173.8, 260.6,  
  'Mexico/Chile', 221.1, 269.8)
```

Una vez introducidos los datos obtendremos el orden para cada país. ¿Cómo? Haremos la media entre los dos valores y utilizaremos ese ranking para el orden final.

Respecto a los *lollipop chart*, únicamente necesitamos agregar otro `geom_line()` indicando los datos de la segunda variable numérica.

El resultado es el siguiente:



Esta visualización puede requerir más trabajo al no disponer de un geom propio, pero el resultado merece la pena. Permite apreciar un ranking global y, además, permite extraer conclusiones específicas con un solo vistazo.

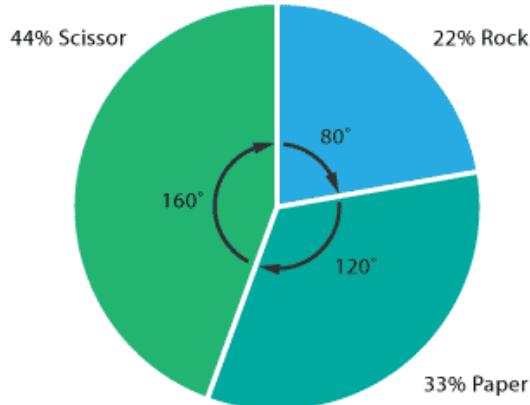
En este caso, por ejemplo, India y China son las regiones que más evolución han experimentado, mientras que otros países como Brasil o el Medio Este son los únicos que han sufrido un retroceso.

Gráfico de tartas en R: la eterna crítica

Ampliamente utilizado en los medios y en el mundo de los negocios, el **gráfico de tarta** es posiblemente la **visualización más denostada por los expertos** (y no sin razones de peso...).

Este tipo de gráficos ayuda a **comparar proporciones y porcentajes entre los niveles de una variable categórica** (justo la misma misión que los gráficos de barras), dividiendo un círculo en segmentos proporcionales. La extensión de cada uno de ellos representará el peso de cada nivel y la suma total será igual a 100%.

Veamos con más detalle la anatomía de este gráfico:



Data			
Rock	Paper	Scissor	TOTAL
2	3	4	9
To calculate percentages			
$2/9=22\%$	$3/9=33\%$	$4/9=44\%$	100%
Degrees for each "pie slice"			
$(2/9) \times 360 = 80^\circ$	$(3/9) \times 360 = 120^\circ$	$(4/9) \times 360 = 160^\circ$	360°

Fuente: [DataViz](#).

Mi recomendación es que **intentes evitar su uso en lo posible**. Te explico alguno de sus principales problemas:

- Los lectores suelen ser malos **trasladando ángulos a valores o proporciones**. Este requisito mental hace que los aprendizajes puedan ser muy subjetivos según el lector que interprete el gráfico.
- Solo son **útiles cuando deben mostrar pocos valores** (esto es un hándicap de muchos gráficos en realidad), ya que conforme aumenta el número de cortes o porciones, la interpretación se dificulta cada vez más por el tamaño de estos. Imposible usarlos con gran cantidad de información.
- Si tratamos de **generar varios para realizar comparaciones** el desastre puede ser legendario. La comparativa entre tamaños de porciones de un gráfico a otro puede ser una tarea que conlleve muchos errores.
- Si los usas**, que sea para mostrar una sola variable, con pocos valores. No muestres la leyenda y anota los niveles sobre las porciones y, lo más importante: no uses efecto 3D.

Para esta función, **utiliza alternativas como los gráficos de barras o piruleta**, con los que obtendrás mejores resultados. ¡Vamos a verlos en R!

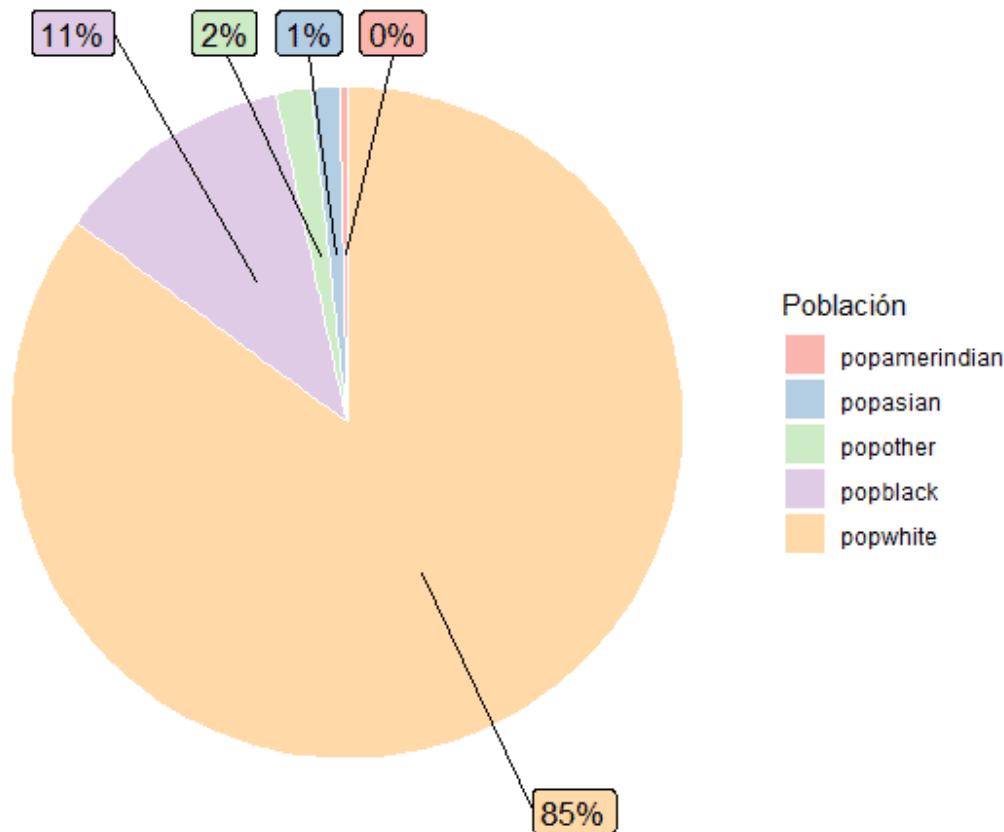
Como verás, R tiene una bella forma de mostrar cuándo un gráfico no es adecuado. En este caso, queremos agregar la población por raza en los condados del Medio Oeste y mostrarlos mediante un gráfico de tartas.

```
## Pie chart
library(ggrepel)

## Generamos datos agregados para facilitar la visualización
midwest2 <- midwest %>%
  ## Seleccionamos las variables de interés y pivotamos para obtener una tabla en formato long
  select(popwhite:popother) %>%
  pivot_longer(everything(),
              names_to = "poblacion",
              values_to = "valor") %>%
  group_by(poblacion) %>%
  summarise(valor = sum(valor), .groups = "drop") %>%
  # Ordenamos los valores para facilitar la visualización en orden
  arrange(-valor) %>%
  # Generamos las proporciones y la posición de las etiquetas
  mutate(prop = valor / sum(valor),
        csum = rev(cumsum(rev(prop))),
        pos = prop/2 + lead(csum, 1),
        pos = if_else(is.na(pos), prop/2, pos))

# Pintamos el gráfico
ggplot(midwest2, aes(x = "", y=prop, fill=fct_inorder(poblacion))) +
  geom_bar(width = 1, stat = "identity", color="white") +
  coord_polar(theta = "y") +
  scale_fill_brewer(palette="Pastel1") +
  # Nos apoyamos en la función de la librería de ggrepel para que no haya superposición
  geom_label_repel(data = midwest2,
                   aes(y = pos, label = paste0(round(prop*100, 0), "%")),
                   size = 4.5, nudge_x = 1, show.legend = FALSE) +
  guides(fill = guide_legend(title = "Población")) +
  theme_void()
```

El resultado de dicho código es el siguiente:



Todo se basa en el `geom_bar()` y la combinación con la función `coord_polar()` que altera el eje de coordenadas cartesiana habitual. A partir de ahí, utilizamos la librería `ggrepel` para evitar el solapamiento de las etiquetas en casos como este, donde existen fragmentos pequeños de población, por ejemplo, la asiática o la nativa.

Te dejo a ti que intentes eliminar la leyenda incluyéndola en el interior de la tarta.

Por cierto, existe una variación estética de este gráfico llamado el **gráfico de donut**:

Fuente: [DataViz](#).



Las características y críticas son exactamente las mismas que las del gráfico de tarta, por lo que no vamos a entrar en su desarrollo en R.

Conclusiones

X Edix Educación

En el octavo fastbook hemos analizado el nicho de visualizaciones enfocado a las **comparaciones numéricas**, empezando por el archiconocido **gráfico de barras o columnas** — ampliamente utilizado en los medios (no siempre con intenciones informativas) —, hasta otras alternativas, como los **gráficos piruleta o de puntos de Cleveland** y, algunos menos queridos, como los **gráficos de tarta**. Todos ellos nos ayudan a realizar comparaciones numéricas entre diferentes variables categóricas o entre niveles de una sola de estas variables. Elige sabiamente entre ellos en función de lo que deseas transmitir.

Como siempre, **te recomiendo encarecidamente que pruebes los resultados en cada opción y no te quedes solo con la lectura de este documento**. Esta ciencia se aprende con práctica, así que, adelante, intenta picar el código para afianzar lo aprendido.

¿Te atreves a utilizar esta visualización con algún conjunto de datos propio? Prueba y extrae conclusiones.

Lesson 4 of 4

Bibliografía

X Edix Educación

- [Ggplot2 – Página principal.](#)
- [Data Visualization, de Kieran Healy.](#)

¡Enhorabuena! Fastbook superado

edix

Creamos Digital Workers