Introducción al análisis de datos con R (y chuches!)

Dan Chitwood, Donald Danforth Plant Science Center (St. Louis, MO). Traducción y adaptación: Antonio J Matas Arroyo. Universidad de Málaga (España)

24 de febrero de 2016

Introducción

El objetivo de este documentos es servir de guía e introducción al análisis computacional de datos de fenotipo con R. Se elaboró para el seminario "", enmarcado en THE MENTORAC PROJECT - UMA en la Universidad de Málaga, a partir del un repositorio de Dan Chitwood and Chris Topp (Donald Danforth Plant Science Center, St. Louis). Es el primer bloque de ejerciocios prácticas del seminario que incluirá, más adelante, toma y análisis de imagen digital, y un reto para los alumnos.

Como ejemplo se usaran datos de chucherías (candy, empleando el set de datos original del citado repositorio.

Sobre las chucherías

Efectivamente, chuches serán analizadas desde un punto de vista fenotípico durante este seminario. En el set de datos aparecen chuches que van desde chocolate hasta mantequilla de cacahuete a osos de gominola. Como prueba, se adjunta una imagen de los 75 tipos de chuches que serán analizados:

¿Qué se entiende como fenotipo? Lo usamos para hace referencia a todos los atributos de un organismo (o chuches en este caso). El fenotipo puede ser infinito: no solo es la suma de los atributos desde el nivel molecular al celular al organimos completo, pero que varía de forma plástica en respuesta a numerosos estímulos ambientales que el organismo puede recibir, y se manifiesta de forma dinámica a lo largo de la vida y el desarrollo de un organimos. Nunca obtendremos el fenotipo exacto de un organimos, ni de las chuces) pero intentaremos obtener tantos atributos como podamos (en este caso de los 75 tipos de chuches). Para este ejercicio de introducción se incluyen las etiquetas de valor alimencicio y composición de estas chuches, pero se podrían completar con más datos como color o forma.

How does R work?

Si estás leyendo este documento en R, ya cuentas con una instalación de R. R es una plataforma libre (gratuíta y de código abierto) que permite a los usuarios usar "paquetes" (conjuntos de funciones y programas) para análisis especializados o para escribir tus propios programas y compartirlos con otros. Tiene términos y gramática propios que pueden resultar confusos y difícil al principio, pero una vez que los hayas dominado, te permitirá hacer análisis complejos y tareas repetitivas de una forma fácil y automatizada.

Primer paso, importar los datos a R

Como no disponemos de mucho tiempo, iremos directos al análisis sin profundizar en lo que significa o implica cada detalle del código, pero os llevaréis una idea del potencial y la filosofía del análisis de datos con esta herramienta.

En cualquier caso real, lo primero que se hace es cargar (leer) tus datos en R. Estos datos pueden estar organizados en muchos formatos (vectores, matrices, listas...) pero en nuestro ejemplo tienen en formato de

Figure 1: Caption

una tabla (dataframe) Cada fila es una observación y cada columna una variable. Las filas y columns tienen nombres a los que podemos hacer referencia para incluirlos en el análisis.

En primer lugar debemos decirle a R dónde encontrar los datos y qué hacer con ellos.

En este caso, usando RStudio como ayuda, habremos creado un proyecto y en la raíz de ese proyecto encontraremos este documento y la tabla con los datos "candy_nutrition.txt". Si en la pestaña de archivos puedes ver ese documento, perfecto.

Este archivo con datos está salvado como un documento delimitado por tabulaciones (cada columna está separada de las anterior y siguiente por una tabulación). Para leer los datos usaremos la función `read.table()` Uno de los argumentos, `header`, le diche a R que se va a encontrar nombres de columnas, que están presentes en la primera fila de nuestros datos, por lo que este argumentos lo estableceremos como verdadero, `TRUE`, como se ve a continuación:

```
data <- read.table("./candy_nutrition.txt", header=TRUE)
```

Acabamos de crear un objetivo `data`, al asignar los datos que hemos leído mediante el símbolo `<-`. Significa que "para la salida de la función `read.table()`, crea un objeto con ese nombre y pon dentro los datos que acabamos de leer". Ahora, los datos están accesibles en R dentor del objetio `data`. Podríamos haber llamado a este objeto de cualquier forma, `pepe_data`, `ana_data`, `datos_chuches`...

Genial! Ya tenemos datos, ahora veamos cómo podemos trabajar con ellos.

Comprobando los datos

Lo primero que debes hacer tras leer los datos en R es comprobar que se han importado correctamente.

Usaremos la función `names()` para ver los nombres de la columnas (variables) de nuestros datos.

```
names(data)
```

```
##  [1] "id"                "name"              "company"
##  [4] "class"             "serving_size_g"    "calories"
##  [7] "calories_fat"      "total_fat_g"       "saturated_fat_g"
## [10] "cholesterol_mg"    "sodium_mg"         "total_carb_g"
## [13] "dietary_fiber_g"   "sugars_g"          "protein_g"
## [16] "primary_ingredient"
```

Nos dice que la primera columna es `id`, la segunda `name`, la tercera `company`, etc.

He usado el nombre original en inglés. A continuación una breve descripción de lo que es cada variable:

- `id`: un identificador único de cada tipo de cuche
- `name`: el nombre comercial de la chuche
- `company`: la compañía que fabrica la chuche
- `class`: el tipo o clase general
- `serving_size_g`: el tamaño de la ración en gramos
- `calories`: calorías por ración
- `calories_fat`: calorías procedentes de la grasa
- `total_fat_g`: el total de grasas en gramos
- `saturated_fat_g`: las grasas saturadas en gramos
- `cholesterol_mg`: colesterol en miligramos
- `sodium_mg`: sal (NaCl) en miligramos
- `total_carb_g`: carbohidratos totales en gramos
- `dietary fiber`: fibra alimentaria
- `sugars`: azucar en gramos
- `primary ingredient`: el primer ingrediente (el más abundante) de la etiqueta

Usaremos ahora la función `head()` para ver las primeras 10 líneas de nuestros datos:

```
head(data, n=10)
```

```
##       id                     name     company      class serving_size_g
## 1   id_1                mini_eggs     cadbury  chocolate             40
## 2   id_2    soft_eating_liquorice darrell_lea  liquorice             42
## 3   id_3              raspberries      haribo      sugar             39
## 4   id_4               candy_corn        nice      gummi             41
## 5   id_5           crawlers_minis      trolli       sour             40
## 6   id_6 strawberry_shortcake_mms        mars  chocolate             42
## 7   id_7           milk_chocolate     hershey  chocolate             43
## 8   id_8                milk_duds     hershey  chocolate             39
## 9   id_9       marshmallow_chicks   just_born      sugar             42
## 10 id_10    crazy_beans_starburst     wrigley jelly_bean             40
##    calories calories_fat total_fat_g saturated_fat_g cholesterol_mg
## 1       190           70           8             5.0              5
## 2       140           10           1             0.0              0
## 3       140            0           0             0.0              0
## 4       160          160           0             0.0              0
## 5       130            0           0             0.0              0
## 6       210          100          10             6.0              5
## 7       210          110          13             7.0              5
## 8       170           60           6             3.5              0
## 9       140            0           0             0.0              0
## 10      130            0           0             0.0              0
##    sodium_mg total_carb_g dietary_fiber_g sugars_g protein_g
## 1         30           28             0.5       27       2.0
## 2         40           30             0.0       16       1.0
## 3          0           36             0.0       29       1.0
## 4         75           39             0.0       32       0.0
## 5         35           31             0.0       24       1.0
## 6         40           29             0.0       28       2.0
## 7         35           26             2.0       22       3.0
## 8        105           27             0.0       20       0.5
## 9         15           36             0.0       34       1.0
## 10         0           33             0.0       26       0.0
##    primary_ingredient
## 1           chocolate
## 2               syrup
## 3               sugar
## 4               sugar
## 5               syrup
## 6           chocolate
## 7               sugar
## 8               syrup
## 9               sugar
## 10              sugar
```

Para ver las últimas líneas usaremos `tail()`:

```
tail(data)
```

```
##       id                          name   company         class
## 70 id_70             marshmallow_eggs just_born         sugar
## 71 id_71            tropical_starburst   wrigley    jelly_bean
```

```
## 72 id_72              reeses_miniatures   hershey peanut_butter
## 73 id_73 reese_peanut_butter_eggs_large   hershey peanut_butter
## 74 id_74             original_starburst   wrigley    jelly_bean
## 75 id_75             marshmallow_chicks just_born         sugar
##    serving_size_g calories calories_fat total_fat_g saturated_fat_g
## 70             32      110            0           0               0
## 71             40      140            0           0               0
## 72             44      220          110          13               5
## 73             34      170           90          10               3
## 74             40      140            0           0               0
## 75             42      140            0           0               0
##    cholesterol_mg sodium_mg total_carb_g dietary_fiber_g sugars_g
## 70            0.0        10           28               0       26
## 71            0.0         0           34               0       27
## 72            2.5       130           26               1       23
## 73            0.0       135           18               1       16
## 74            0.0         0           34               0       27
## 75            0.0        15           34               0       32
##    protein_g primary_ingredient
## 70         1              sugar
## 71         0              sugar
## 72         4          chocolate
## 73         4            peanuts
## 74         0              sugar
## 75         1              sugar
```

Entre las funciones más interesantes tenemos `summary()`. Nos resume toda la tabla de datos:

```
summary(data)
```

```
##        id                         name         company             class
##  id_1   : 1   marshmallow_chicks   : 4   hershey  :17   chocolate    :26
##  id_10  : 1   marshmallow_bunnies  : 2   mars     :11   gummi        : 9
##  id_11  : 1   milk_chocolate       : 2   just_born: 8   jelly_bean   : 5
##  id_12  : 1   peanut_mms           : 2   wrigley  : 8   liquorice    : 6
##  id_13  : 1   soft_eating_liquorice: 2   nestle   : 6   peanut_butter: 7
##  id_14  : 1   baby_ruth            : 1   haribo   : 4   sour         : 7
##  (Other):69   (Other)              :62   (Other)  :21   sugar        :15
##  serving_size_g     calories      calories_fat  total_fat_g
##  Min.   : 7.00   Min.   : 25.0   Min.   :  0   Min.   : 0.000
##  1st Qu.:39.50   1st Qu.:140.0   1st Qu.:  0   1st Qu.: 0.000
##  Median :40.00   Median :150.0   Median : 15   Median : 1.500
##  Mean   :38.56   Mean   :158.2   Mean   : 39   Mean   : 4.167
##  3rd Qu.:42.00   3rd Qu.:190.0   3rd Qu.: 80   3rd Qu.: 8.500
##  Max.   :45.00   Max.   :220.0   Max.   :160   Max.   :13.000
##
##  saturated_fat_g cholesterol_mg     sodium_mg       total_carb_g
##  Min.   :0.000   Min.   : 0.000   Min.   :  0.00   Min.   : 6.00
##  1st Qu.:0.000   1st Qu.: 0.000   1st Qu.: 10.00   1st Qu.:26.00
##  Median :1.000   Median : 0.000   Median : 30.00   Median :31.00
##  Mean   :2.347   Mean   : 1.567   Mean   : 37.73   Mean   :29.63
##  3rd Qu.:5.000   3rd Qu.: 2.500   3rd Qu.: 50.00   3rd Qu.:34.00
##  Max.   :8.000   Max.   :10.000   Max.   :180.00   Max.   :39.00
##
##  dietary_fiber_g     sugars_g       protein_g    primary_ingredient
```

```
##  Min.   :0.0000   Min.   : 6.00   Min.   :0.00   chocolate:21
##  1st Qu.:0.0000   1st Qu.:20.50   1st Qu.:0.00   dextrose : 2
##  Median :0.0000   Median :24.00   Median :1.00   peanuts  : 1
##  Mean   :0.3667   Mean   :23.63   Mean   :1.42   sugar    :37
##  3rd Qu.:1.0000   3rd Qu.:27.00   3rd Qu.:2.00   syrup    :14
##  Max.   :2.0000   Max.   :35.00   Max.   :5.00
##
```

Algunas columnas, como `id`, `name`, `company`, `class`, y `primary_ingredient`, indican cuántos términos diferentes hay. Otras, como `calories` y `total_fat_g`, muestran el mínimo, máximo, media, mediana y los cuartiles. Las variables en R serán o bien "factores" o bien "numericas". Básicamente, estas son variables categoricas o numéricas contínuas.

Visualización de datos con ggplot2

Primero debemos cargar el paquete ggplot2.

```
library(ggplot2)
```

Una vez cargado podemos usar las funciones que están definidas en este paquete.

Para saber más sobre `ggplot2`, puedes consultar la magnífica documentación.

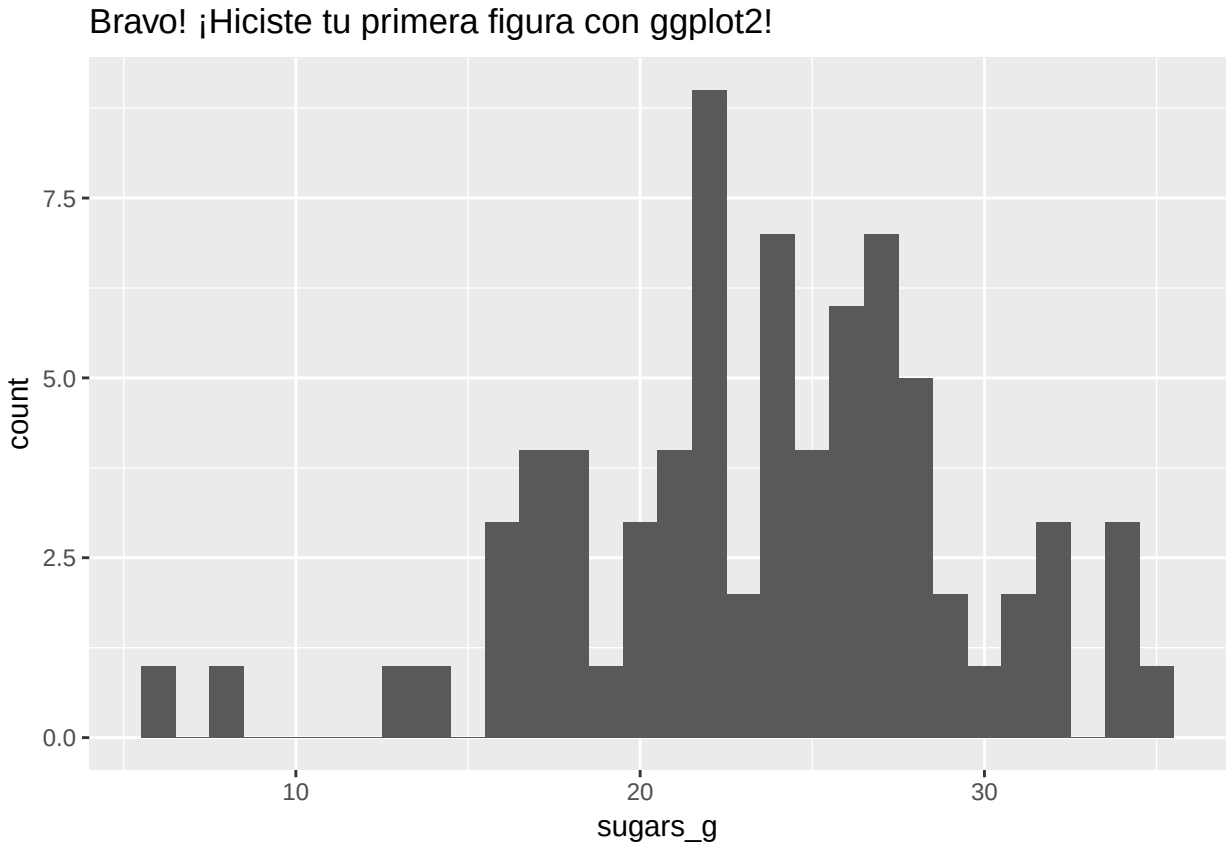
¡Hagamos nuestra primera gráfica!

```
p <- ggplot(data=data, aes(x=sugars_g))

p + geom_histogram() +

ggtitle(label="Bravo! ¡Hiciste tu primera figura con ggplot2!")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



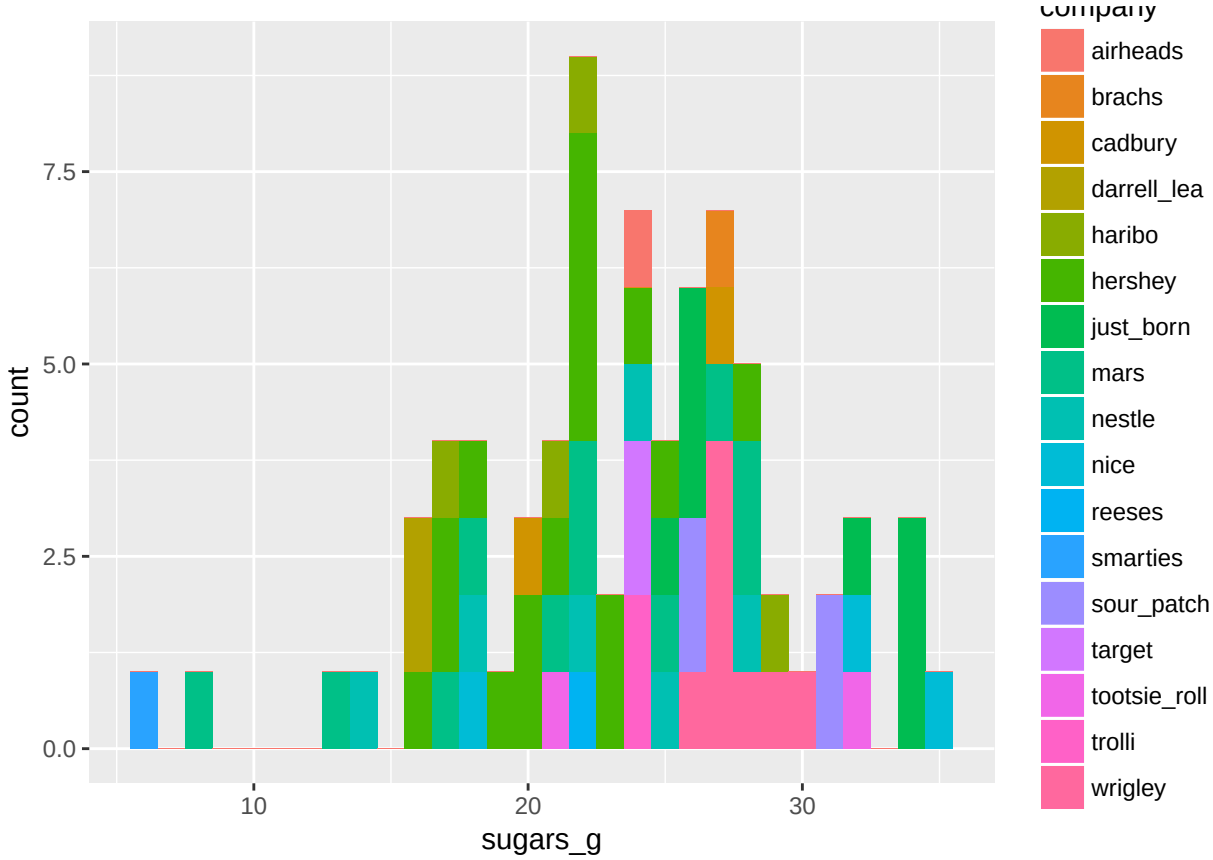
Podemos jugar un poco con las funcions de `ggplot2` y explorar sus posibilidades. La figura anterior es un histograma y mustra la distribución de valores para un carácter, en este caso `sugars_g`. La función `ggplot()` tiene una disposición por capas: en un objetio llamado `p` creado por la función `ggplot()` se especifican los datos, y después las variables usadas para la figura mediante otra función, `aes()`, que proviene de "aesthetics". En el caso anterior especificamos que x sea la variable `sugars_g`. Una vez creado el objeto `p`, podemos añadir "geom"s" que son básicamente tipos de gráficas. `geom_histogram()` crea un histograma. Hay muchas otras opciones com `ggtitle()` que añade un título.

Podmeos usar los atributos `fill` y `colour` en la función `aes()`. Por ejemplo, añadamos `fill=company` a la figura anterior.

```
p <- ggplot(data=data, aes(x=sugars_g, fill=company))

p + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Podemos controlar el esquema de color usando las funciones `scale_fill` o `scale_colour`. Reagamos nuestro histograma por el tipo de chuche, para las que hay 7 tipos diferentes. Podmeos especificar los colores por nombre "name". El orden de los niveles establecerá el orden de los colores. En la clase caracter el orden es alfabético:

```
summary(data$class)
```

```
##     chocolate         gummi    jelly_bean     liquorice peanut_butter
##            26             9             5             6             7
##          sour         sugar
##             7            15
```

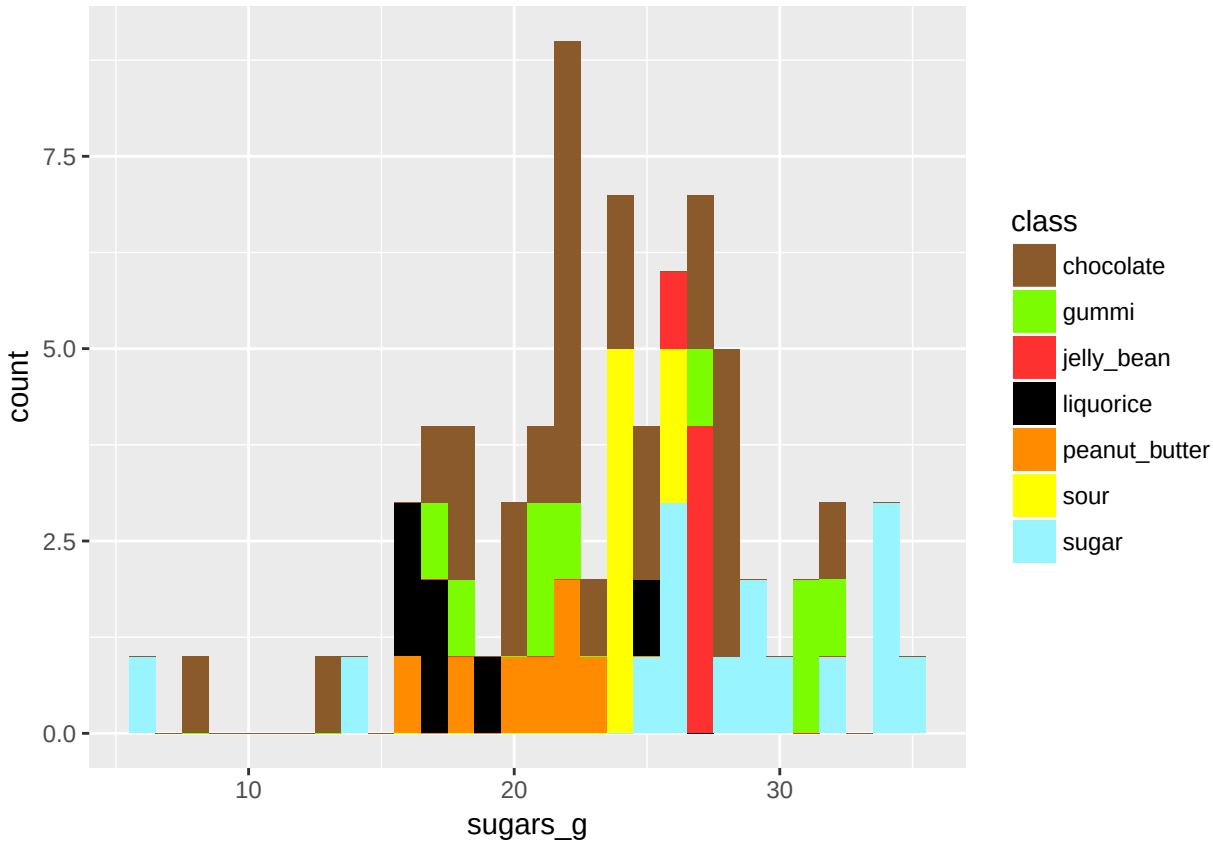
Tratemos de asignarle el color según el tipo:

```
p <- ggplot(data=data, aes(x=sugars_g, fill=class))

p + geom_histogram() +

scale_fill_manual(values=c("tan4","lawngreen","firebrick1","black","darkorange","yellow","cadetblue1"))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



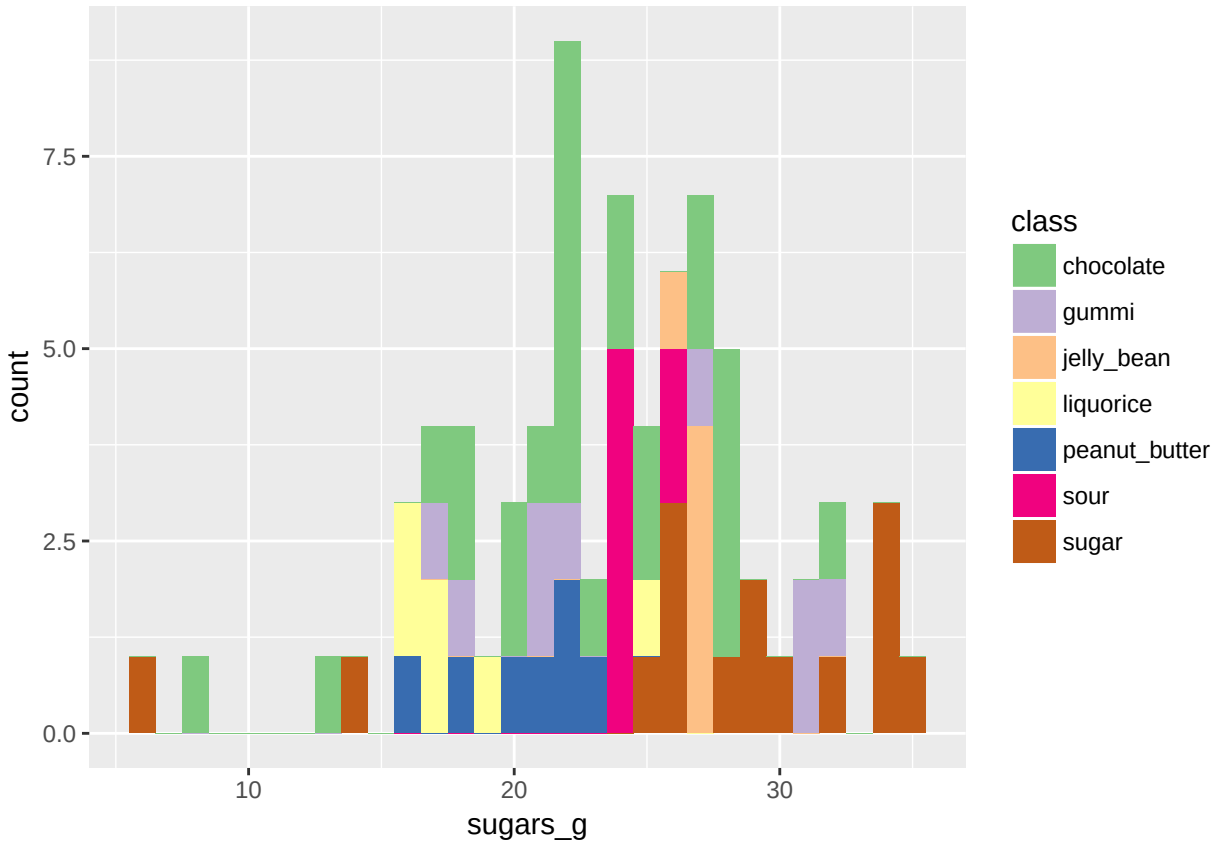
Si no nos gusta la paleta d ecolores, podemos usar paletas más elaboradas, como las incluídas en la web color brewer, usando la función `scale_fill_brewer()`.

```
p <- ggplot(data=data, aes(x=sugars_g, fill=class))

p + geom_histogram() +

scale_fill_brewer(type="qual", palette=1)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



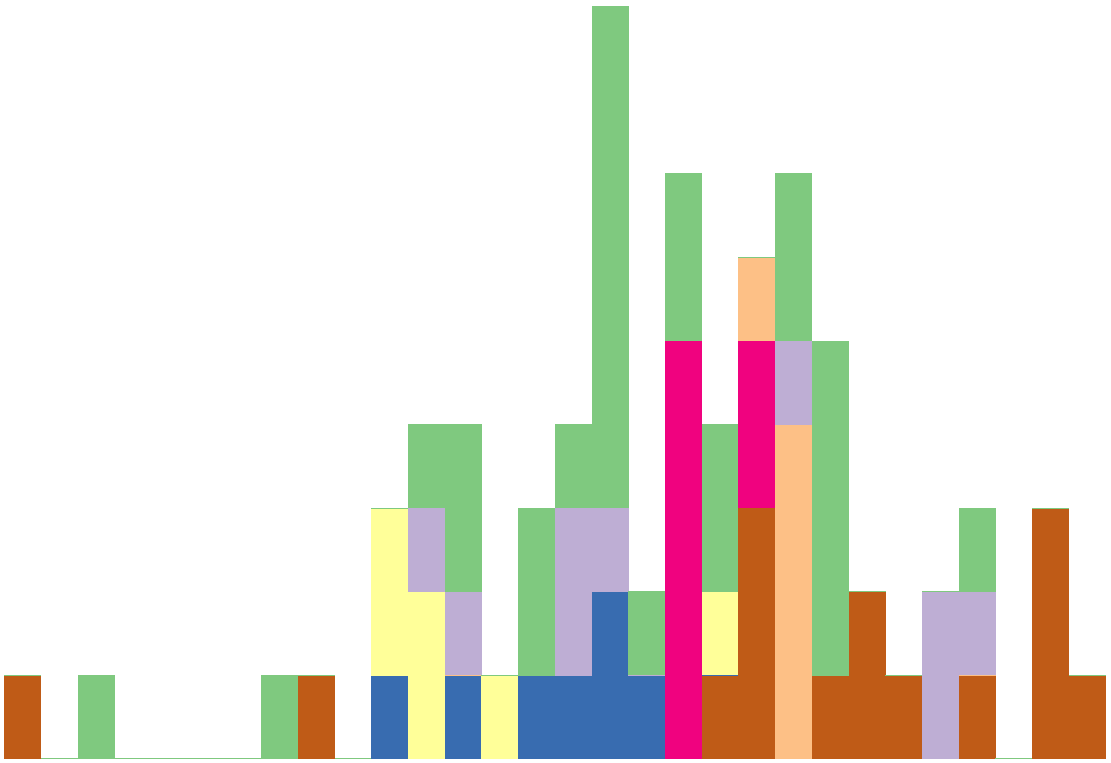
Podemos manipular otros aspectos de la gráfica usando la función `theme()`. Para empezar, definiremos todas estas características como vacías:

```
p <- ggplot(data=data, aes(x=sugars_g, fill=class))

p + geom_histogram() +

scale_fill_brewer(type="qual", palette=1) +
theme(axis.line=element_blank(),
      axis.text.x=element_blank(),
      axis.text.y=element_blank(),
      axis.ticks=element_blank(),
      axis.title.x=element_blank(),
      axis.title.y=element_blank(),
      legend.position="none",
      panel.background=element_blank(),
      panel.border=element_blank(),
      panel.grid.major=element_blank(),
      panel.grid.minor=element_blank(),
      plot.background=element_blank())
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



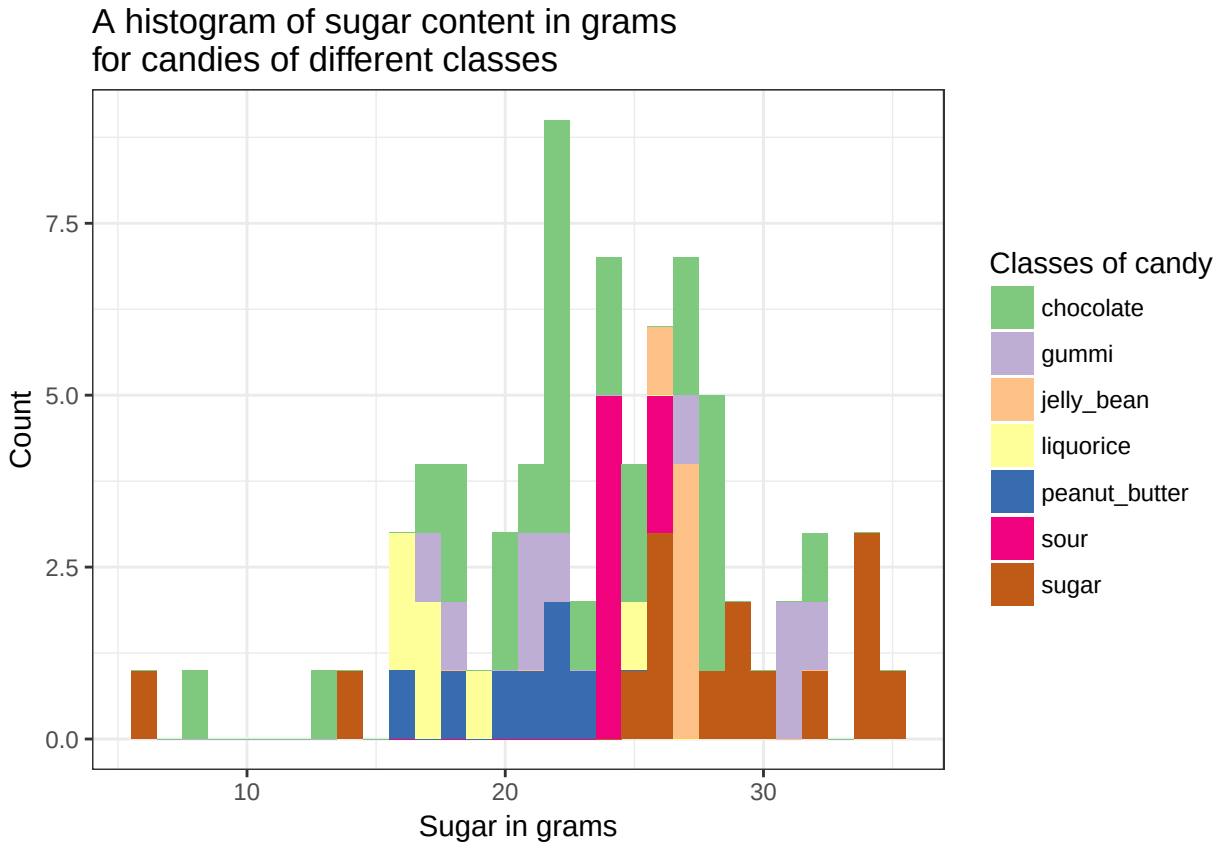
¡Nadie quiere *nothing* en su gráfica! Vamos a añadir tan solo títulos para los ejes usando `xlab()` y `ylab()` y vamos a quitar el fondo gris con `theme_bw()`. Fijáos también en el cambioi del título de la leyenda y que hemos puesto el título en dos líneas con la función `ggtitle` y `\n`, que crea una nueva línea.

```
p <- ggplot(data=data, aes(x=sugars_g, fill=class))

p + geom_histogram() +

scale_fill_brewer(type="qual", palette=1, name="Classes of candy") +
xlab(label="Sugar in grams") +
ylab(label="Count") +
ggtitle("A histogram of sugar content in grams\nfor candies of different classes") +
theme_bw()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

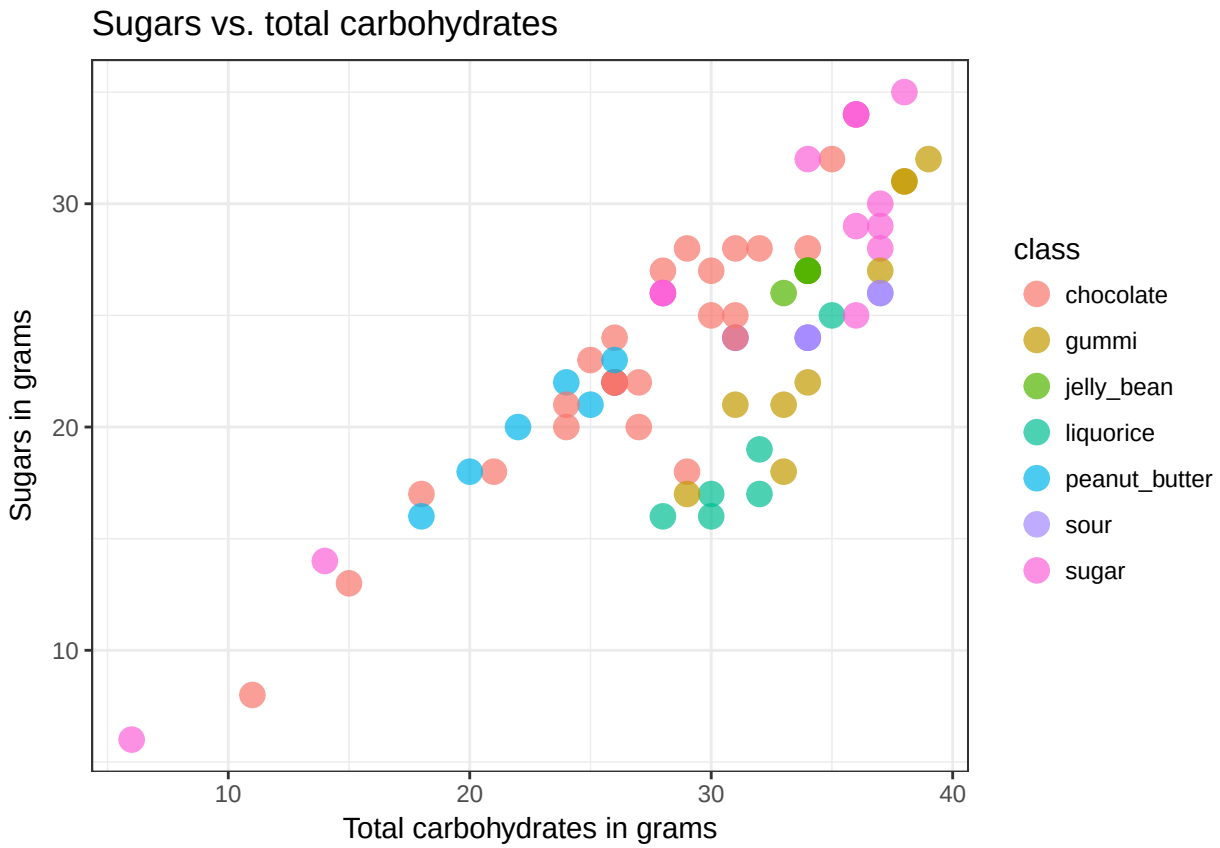


Ahora vamos a probar otros tipos de figuras, como los diagramas de dispersión. Fijáos en el uso de `size` y `alpha` para los puntos (`alpha` indica transparencia, donde "1" es opaco).

```
p <- ggplot(data=data, aes(x=total_carb_g, y=sugars_g, colour=class))

p + geom_point(size=4, alpha=0.7) +

scale_fill_brewer(type="qual", palette=1, name="Classes of candy") +
xlab(label="Total carbohydrates in grams") +
ylab(label="Sugars in grams") +
ggtitle("Sugars vs. total carbohydrates") +
theme_bw()
```



Estaría bien conocer la identida de cada chuche en esta figura. Y si dos chuches se superponen, ¿no sería mejor poder ver ambos nombres? Instalaremos el paquete `ggrepel` para poder usar la función `geom_repel_text`

```
install.packages("ggrepel")
```

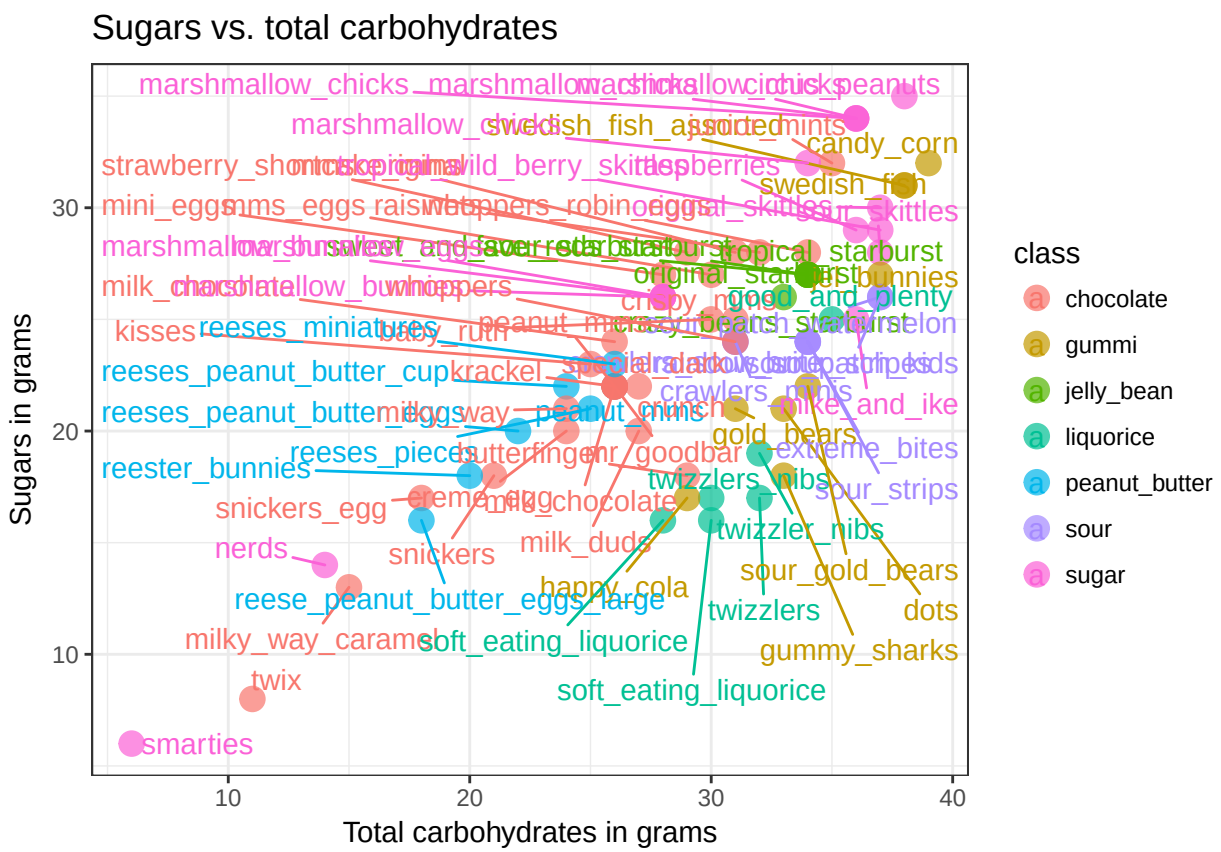
```
library(ggrepel)
```

Y volvemos a hacer la figura usando la función `geom_text_repel`:

```
p <- ggplot(data=data, aes(x=total_carb_g, y=sugars_g, colour=class))

p + geom_point(size=4, alpha=0.7) +

geom_text_repel(data=data, aes(x=total_carb_g, y=sugars_g, label=name)) +
scale_fill_brewer(type="qual", palette=1, name="Classes of candy") +
xlab(label="Total carbohydrates in grams") +
ylab(label="Sugars in grams") +
ggtitle("Sugars vs. total carbohydrates") +
theme_bw()
```

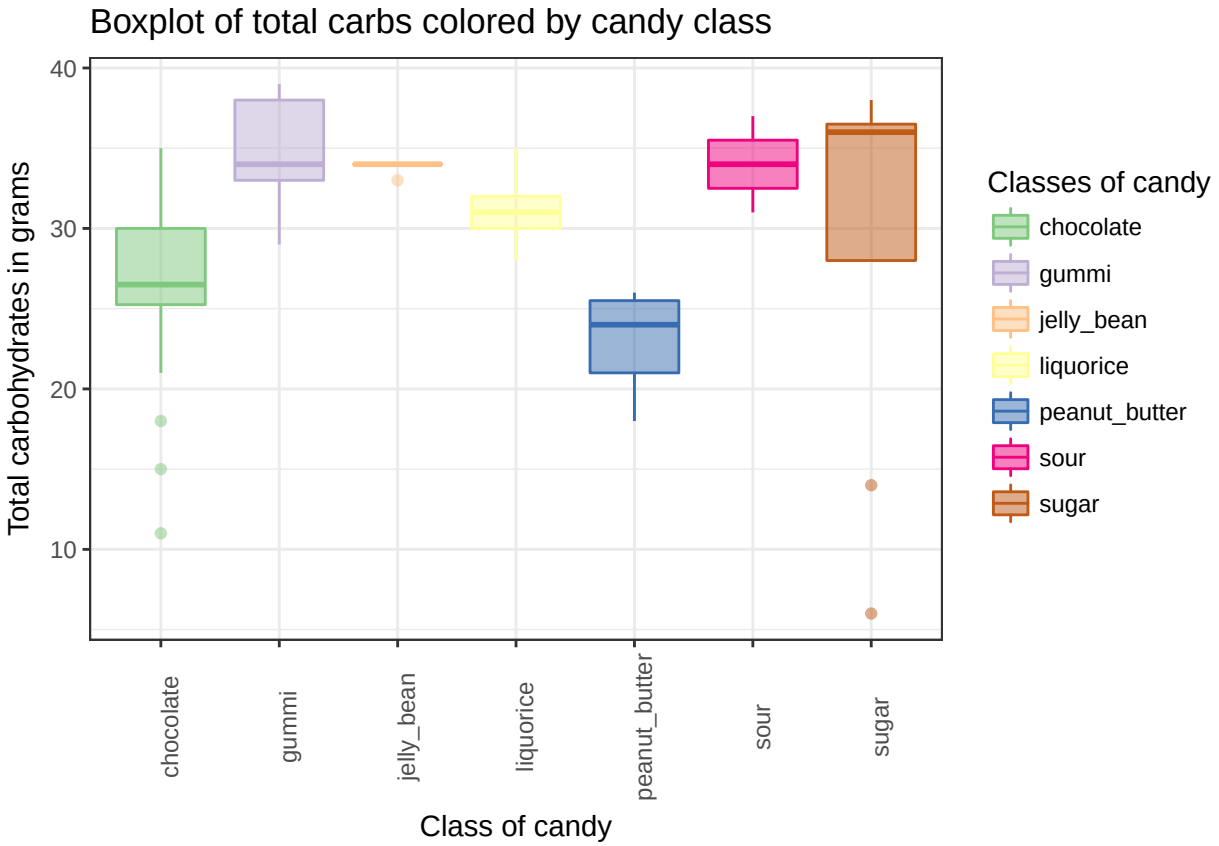


Un último tipo de gráfica, la caja y bigotes o boxplot.

```
p <- ggplot(data=data, aes(x=class, y=total_carb_g, colour=class, fill=class))

p + geom_boxplot(alpha=0.5) +

scale_fill_brewer(type="qual", palette=1, name="Classes of candy") +
scale_colour_brewer(type="qual", palette=1, name="Classes of candy") +
xlab(label="Class of candy") +
ylab(label="Total carbohydrates in grams") +
ggtitle("Boxplot of total carbs colored by candy class") +
theme_bw() +
theme(axis.text.x=element_text(angle=90))
```



La habilidad crear buenas visualizaciones está en el corazón de la exploración de datos y de poder conocer tus datos. Las habilidades que habéis puesto en práctica con este módulo se podrán usar en el siguiente módulo de este seminario, análisis de imágen, o en cualquier otro estudio que requiera manejar datos.