Análisis LEGO con R

Sonia Remacha

2025-03-13

1. Instalación y carga de paquetes avanzados

Se utilizarán readr para la importación del archivo .csv, tidytable y data.table para el procesamiento de datos, y ggplot2 para la visualización.

Para la instalación de tidytable, instalamos primero devtools, ya que tidytable no está en CRAN.

```
# Lista de librerías de CRAN necesarias
cran_libraries <- c("readr", "data.table", "ggplot2", "rmarkdown")

# Comprobar e instalar librerías de CRAN
for (lib in cran_libraries) {
  if (!requireNamespace(lib, quietly = TRUE)) {
    install.packages(lib)
  }
}

# Comprobar si devtools está instalado, si no, instalarlo
if (!requireNamespace("devtools", quietly = TRUE)) {
  install.packages("devtools")
}

# Instalar la librería desde GitHub si no está instalada
if (!requireNamespace("tidytable", quietly = TRUE)) {
  devtools::install_github("markfairbanks/tidytable")
}

# Cargar todas las librerías
library(readr)
library(data.table)
library(ggplot2)
library(rmarkdown)
library(tidytable)
```

2. Carga de datos

Cargamos “lego_population_cof.csv” y exploramos.

```
lego_data<- read.csv("lego_population_cof.csv")
str(lego_data)
```

```
## 'data.frame':    2307 obs. of  16 variables:
##  $ X             : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Item_Number   : chr  "-" "-" "-" "-" ...
##  $ Set_Name      : chr  NA NA NA NA ...
##  $ Amazon_Price  : chr  "" "" "-" "-" ...
##  $ Year          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ Pages         : logi  NA NA NA NA NA NA ...
##  $ unique_pieces: int  NA NA NA NA NA NA NA NA NA NA ...
##  $ Theme         : chr  NA NA NA NA ...
##  $ Pieces        : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ Price         : logi  NA NA NA NA NA NA ...
##  $ Ages          : chr  "Ages_NA" "Ages_NA" "Ages_NA" "Ages_NA" ...
##  $ Minifigures   : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ Packaging     : chr  NA NA NA NA ...
##  $ Weight        : chr  NA NA NA NA ...
##  $ Availability : chr  NA NA NA NA ...
##  $ Size          : chr  NA NA NA NA ...
```

```
summary(lego_data)
```

```
##        X          Item_Number          Set_Name         Amazon_Price
##  Min.   :   1.0   Length:2307        Length:2307        Length:2307
##  1st Qu.: 577.5   Class :character   Class :character   Class
:character
##  Median :1154.0   Mode  :character   Mode  :character   Mode
:character
##  Mean   :1154.0
##  3rd Qu.:1730.5
##  Max.   :2307.0
##
##       Year       Pages         unique_pieces       Theme
##  Min.   :2018   Mode:logical   Min.   :   1.0   Length:2307
##  1st Qu.:2018   NA's:2307      1st Qu.:  39.0   Class :character
##  Median :2019                  Median :  83.0   Mode  :character
##  Mean   :2019                  Mean   : 118.6
##  3rd Qu.:2020                  3rd Qu.: 164.0
##  Max.   :2020                  Max.   :1067.0
##  NA's   :974                   NA's   :1008
##      Pieces      Price            Ages            Minifigures
##  Min.   :   1   Mode:logical   Length:2307        Min.   : 1.000
##  1st Qu.:  70   NA's:2307      Class :character   1st Qu.: 1.000
##  Median : 193                  Mode  :character   Median : 3.000
##  Mean   : 397                                     Mean   : 3.202
##  3rd Qu.: 456                                     3rd Qu.: 4.000
##  Max.   :9036                                     Max.   :28.000
##  NA's   :986                                      NA's   :1398
##   Packaging           Weight          Availability          Size
##  Length:2307        Length:2307        Length:2307        Length:2307
##  Class :character   Class :character   Class :character   Class
:character
```

```
##  Mode   :character   Mode   :character   Mode   :character   Mode
:character
##
##
##
##
```

Observamos que este archivo tiene datos con valores nulos y erróneos, además de tipos de variable incorrectos, columnas innecesarias y falta de normalización. Por lo tanto el siguiente paso será hacer una limpieza de estos datos.

3. Limpieza de datos

Columna por columna, vemos los siguientes problemas y peculiaridades:

- 1- Column1 (X): No sirve para nada, se puede obviar.
- 2- Item_Number: En algunas filas tiene "-", en otras un número identificatorio, y en otras un precio. Debido a su ambigüedad, puede obviarse.
- 3- Set_Name: Muchas filas con NA, pero el resto parece que se refiere al nombre del set de LEGO. Tratamos NA como nombre desconocido.
- 4- Amazon_price: El formato no está normalizado y no permite estudiarlo bien. Sobran el "€" y ocasional "*". Tratamos"-" como NA, precio de Amazon desconocido. Además, Hay muy pocos datos relevantes así que no nos sirve para el estudio.
- 5- Year: Año de salida del set.
- 6- Pages: Solo contiene NA, puede obviarse.
- 7- unique_pieces: Número de piezas únicas en cada set.
- 8- Theme: Tema del set.
- 9- Pieces: Número de piezas de cada set.
- 10- Price: Solo contiene NA, puede obviarse.
- 11- Ages: Edad recomendada del set, los desconocidos en vez de nombrarlos NA, los nombra "Ages_NA".
- 12- Minifigures: Número de minifiguras de cada se.
- 13- Packaging: Material de la caja donde viene el set.
- 14- Weight: El formato no está normalizado y no permite estudiarlo bien. Incluye una conversión a libras, que obviaremos.
- 15- Availability: Dónde se puede conseguir el set.
- 16- Size: Tamaño del set.

Conversión tipos de variable y eliminación de columnas:

- 1- Column1 (X): Eliminar
- 2- Item_number: Eliminar

- 3- Set_name: Categórica
- 4- Amazon_price: Eliminar
- 5- Year: Categórica
- 6- Pages: Eliminar
- 7- unique_pieces: Numérica
- 8- Theme: Categórica
- 9- Pieces: Numérica
- 10- Price: Eliminar
- 11- Ages: Categórica
- 12- Minifigures: Numérica
- 13- Packaging: Categórica
- 14- Weight: Numérica
- 15- Availability: Categórica
- 16- Size: Categórica

```
# Conversión a tidytable

lego_data_clean <- as_tidytable(lego_data)

# Eliminamos columnas irrelevantes o con muy poca información

lego_data_clean <- tidytable(
  lego_data_clean %>% select (-c("X", "Item_Number", "Pages", "Price",
"Amazon_Price"))
)

head(lego_data_clean)
```

```
## # A tidytable: 6 × 11
##   Set_Name  Year unique_pieces Theme Pieces Ages    Minifigures
Packaging Weight
##   <chr>    <int>         <int> <chr>  <int> <chr>         <int> <chr>
<chr>
## 1 <NA>        NA            NA <NA>      NA Ages_NA          NA <NA>
<NA>
## 2 <NA>        NA            NA <NA>      NA Ages_NA          NA <NA>
<NA>
## 3 <NA>        NA            NA <NA>      NA Ages_NA          NA <NA>
<NA>
## 4 <NA>        NA            NA <NA>      NA Ages_NA          NA <NA>
<NA>
## 5 <NA>        NA            NA <NA>      NA Ages_NA          NA <NA>
<NA>
## 6 <NA>        NA            NA <NA>      NA Ages_NA          NA <NA>
<NA>
## # ℹ 2 more variables: Availability <chr>, Size <chr>
```

```r
# Antes de cambiar el tipo de variable de Weight, quitamos lo sobrante de
sus datos (kg, lb...)
lego_data_clean <- lego_data_clean %>%
  mutate(Weight = parse_number(Weight))

# Conversión tipos de variable
lego_data_clean <- lego_data_clean %>%
  mutate(
    Set_Name = factor(Set_Name),
    Year = factor(Year),
    Theme = factor(Theme),
    Ages = factor(Ages),
    Packaging = factor(Packaging),
    Availability = factor(Availability),
    Size = factor(Size),

    unique_pieces = as.numeric(unique_pieces),
    Pieces = as.numeric(Pieces),
    Minifigures = as.numeric(Minifigures),
    Weight = as.numeric(Weight)
  )

str(lego_data_clean)
```

```
## Classes 'tidytable', 'tbl', 'data.table' and 'data.frame':   2307 obs.
of  11 variables:
##  $ Set_Name     : Factor w/ 1297 levels "123 Sesame Street",..: NA NA
NA NA NA NA NA NA NA NA ...
##  $ Year         : Factor w/ 3 levels "2018","2019",..: NA NA NA NA NA
NA NA NA NA NA ...
##  $ unique_pieces: num  NA NA NA NA NA NA NA NA NA NA ...
##  $ Theme        : Factor w/ 47 levels "Architecture",..: NA NA NA NA
NA NA NA NA NA NA ...
##  $ Pieces       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ Ages         : Factor w/ 35 levels "Ages_1 - 3","Ages_1+",..: 35 35
35 35 35 35 35 35 35 35 ...
##  $ Minifigures  : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ Packaging    : Factor w/ 12 levels "Blister pack",..: NA NA NA NA
NA NA NA NA NA NA ...
##  $ Weight       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ Availability : Factor w/ 7 levels "Educational",..: NA NA NA NA NA
NA NA NA NA NA ...
##  $ Size         : Factor w/ 2 levels "Large","Small": NA NA NA NA NA
NA NA NA NA NA ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

4. Análisis de datos

Una vez limpia nuestra fuente de datos, podemos empezar a analizar.

En este análisis, no podremos estudiar el precio de los sets, ya que las filas que contenían precio (tanto en la columna Item_Number como la Amazon_Price) no iban asociadas a nada más, el resto de columnas no contenían datos relevantes.Por lo tanto, el análisis se hará resolviendo a otro tipo de preguntas no monetarias:

- Distribución de sets por año
- Cantidad de sets por temática
- Tipos de empaques más comunes
- Número de piezas por set
- Relación número de piezas con número de minifiguras
- Edad del público de LEGO y su relación con el número de piezas
- Relación peso con tamaño
- Disponibilidad de los sets

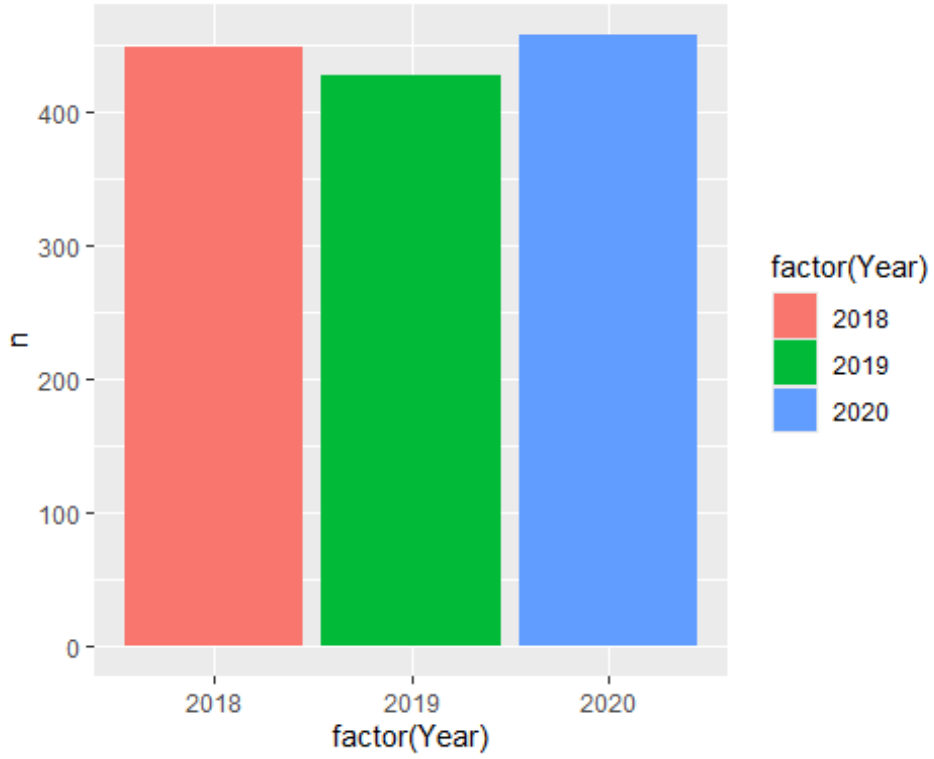
Distribución de sets por año

- ¿Cuántos sets se lanzaron por año? ¿Se han lanzado más sets en años recientes?

```
# Conteo de sets por año
lego_count_year <- lego_data_clean |>
  filter(!is.na(Year)) |>
  count(Year, sort = TRUE) |>
  arrange(Year) |>
  mutate(freq_rel = n / sum(n)*100)

lego_count_year
```

```
## # A tidytable: 3 × 3
##   Year      n freq_rel
##   <fct> <int>    <dbl>
## 1 2018    448     33.6
## 2 2019    427     32.0
## 3 2020    458     34.4
```

```
# Diagrama de barras
ggplot(lego_count_year, aes(x = factor(Year), y = n, fill =
factor(Year))) +
  geom_bar(stat = "identity")
```




```
  labs(title = "Distribución de Sets por Año",
       x = "Año",
       y = "Frecuencia")
```

```
## $x
## [1] "Año"
##
## $y
## [1] "Frecuencia"
##
## $title
## [1] "Distribución de Sets por Año"
##
## attr(,"class")
## [1] "labels"
```

Respondiendo a las preguntas, el número de sets lanzados por año suele ser similar. En el último año se han lanzado más sets que años anteriores, pero no podemos asegurar un crecimiento anual.

Cantidad de sets por temática

- ¿Cuáles son las líneas de LEGO más populares?

```
# Conteo de líneas
lego_count_theme <- lego_data_clean |>
  filter(!is.na(Theme)) |>
  count(Theme, sort = TRUE) |>
```

```r
  mutate(freq_rel = n / sum(n)*100)

lego_count_theme
```

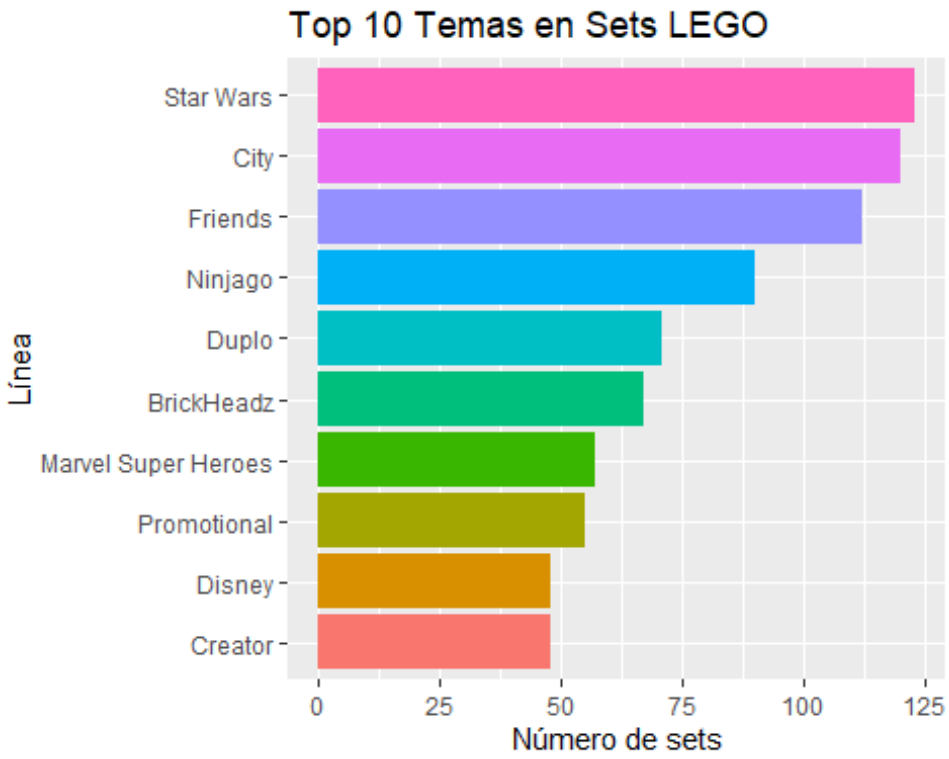
```
## # A tidytable: 47 × 3
##    Theme                  n freq_rel
##    <fct>              <int>    <dbl>
##  1 Star Wars            123     9.25
##  2 City                 120     9.02
##  3 Friends              112     8.42
##  4 Ninjago               90     6.77
##  5 Duplo                 71     5.34
##  6 BrickHeadz            67     5.04
##  7 Marvel Super Heroes   57     4.29
##  8 Promotional           55     4.14
##  9 Creator               48     3.61
## 10 Disney                48     3.61
## # ℹ 37 more rows
```

```r
# Top 10 líneas
lego_count_theme_top10 <- lego_count_theme |>
  top_n(10, wt = n) |>
  arrange(desc(n))

lego_count_theme_top10
```

```
## # A tidytable: 10 × 3
##    Theme                  n freq_rel
##    <fct>              <int>    <dbl>
##  1 Star Wars            123     9.25
##  2 City                 120     9.02
##  3 Friends              112     8.42
##  4 Ninjago               90     6.77
##  5 Duplo                 71     5.34
##  6 BrickHeadz            67     5.04
##  7 Marvel Super Heroes   57     4.29
##  8 Promotional           55     4.14
##  9 Creator               48     3.61
## 10 Disney                48     3.61
```

```r
lego_count_theme_top10$Theme <- factor(
  lego_count_theme_top10$Theme,
  levels = lego_count_theme_top10$Theme[order(lego_count_theme_top10$n)]
)

# Diagrama de barras
ggplot(lego_count_theme_top10, aes(x = Theme, y = n, fill = Theme)) +
  geom_bar(stat = "identity") +
  coord_flip()+
  labs(title = "Top 10 Temas en Sets LEGO",
       x = "Línea",
```

```
       y = "Número de sets") +
  theme(legend.position = "none")
```



La línea de sets más popular de LEGO es Star Wars, seguida de City y Friends.

Tipos de empaques más comunes

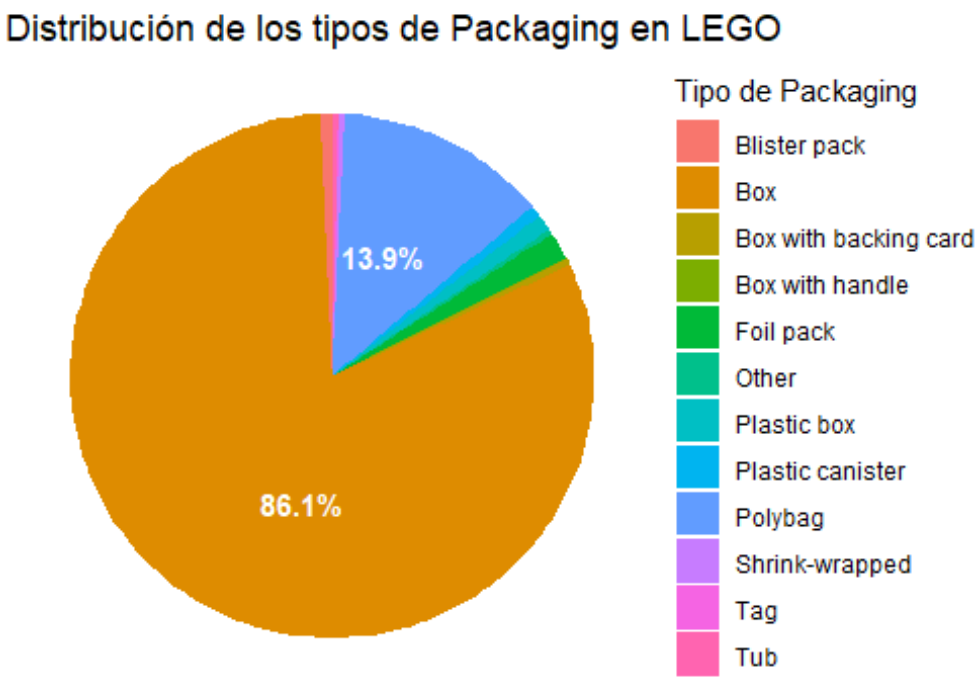
- ¿Los sets vienen mayormente en cajas o en bolsas?

```
# Conteo de tipo de empaque
lego_count_packaging <- lego_data_clean |>
  filter(!is.na(Packaging)) |>
  count(Packaging, sort = TRUE) |>
  mutate(freq_rel = n / sum(n)*100) |>
  arrange(desc(n))

lego_count_packaging
```

```
## # A tidytable: 12 × 3
##    Packaging                n freq_rel
##    <fct>                <int>    <dbl>
##  1 Box                   1051  81.3
##  2 Polybag                169  13.1
##  3 Foil pack               22   1.70
##  4 Plastic box             13   1.01
##  5 Blister pack             9   0.696
##  6 Plastic canister         9   0.696
##  7 Box with backing card    5   0.387
```

```
##  8 Tub                       5   0.387
##  9 Other                     4   0.309
## 10 Shrink-wrapped            4   0.309
## 11 Box with handle           1   0.0773
## 12 Tag                       1   0.0773
```

```r
# Mostrar sólo el número de los 2 mayores tipos de empaque en el diagrama
etiquetas_seleccionadas <- lego_count_packaging |>
  slice(1:2)

# Diagrama de sectores
ggplot(lego_count_packaging, aes(x = "", y = n, fill = Packaging)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar("y", start = 0) +
  theme_void() +
  labs(title = "Distribución de los tipos de Packaging en LEGO",
       fill = "Tipo de Packaging") +
  geom_text(data = etiquetas_seleccionadas,
            aes(label = paste0(round(n/sum(n) * 100, 1), "%")),
            position = position_stack(vjust = 0.5),
            size = 4, color = "white", fontface = "bold")
```



La gran mayoría de sets utiliza cajas (81,3%), seguido de bolsas (13,9%). El resto de empaques son minoritarios.

Número de piezas por set

- ¿Cuál es el set con mayor cantidad de piezas?

```
# Máximo de piezas
lego_max_pieces <- lego_data_clean %>%
  select(Set_Name, Year, Pieces) %>%
  filter(!is.na(Pieces)) %>%
  filter(Pieces == max(Pieces))

lego_max_pieces
```

```
## # A tidytable: 1 × 3
##   Set_Name  Year  Pieces
##   <fct>     <fct>  <dbl>
## 1 Colosseum 2020    9036
```

El set con más piezas es “Colosseum”, del 2020 con 9036 piezas.

Relación número de piezas con número de minifiguras

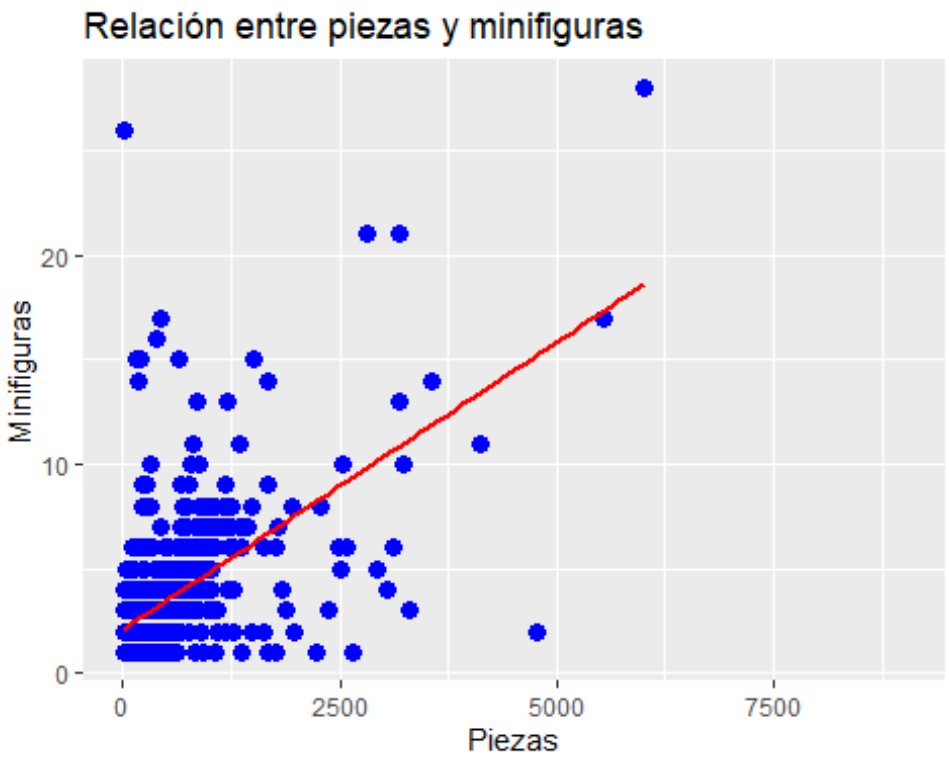
¿Hay correlación entre el número de piezas de un set con el número de minifiguras que pueda tener?

```
# Correlación piezas y minifiguras
correlation_pieces_minifigures <- lego_data_clean %>%
      summarise(correlacion = cor(lego_data_clean$Pieces,
lego_data_clean$Minifigures, use="complete.obs"))

correlation_pieces_minifigures
```

```
## # A tidytable: 1 × 1
##   correlacion
##         <dbl>
## 1       0.574
```

```
# Diagrama de puntos
ggplot(lego_data_clean, aes(x = Pieces, y = Minifigures)) +
  geom_point(color = "blue", size = 3) +
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  labs(title = "Relación entre piezas y minifiguras",
       x = "Piezas",
       y = "Minifiguras")
```

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



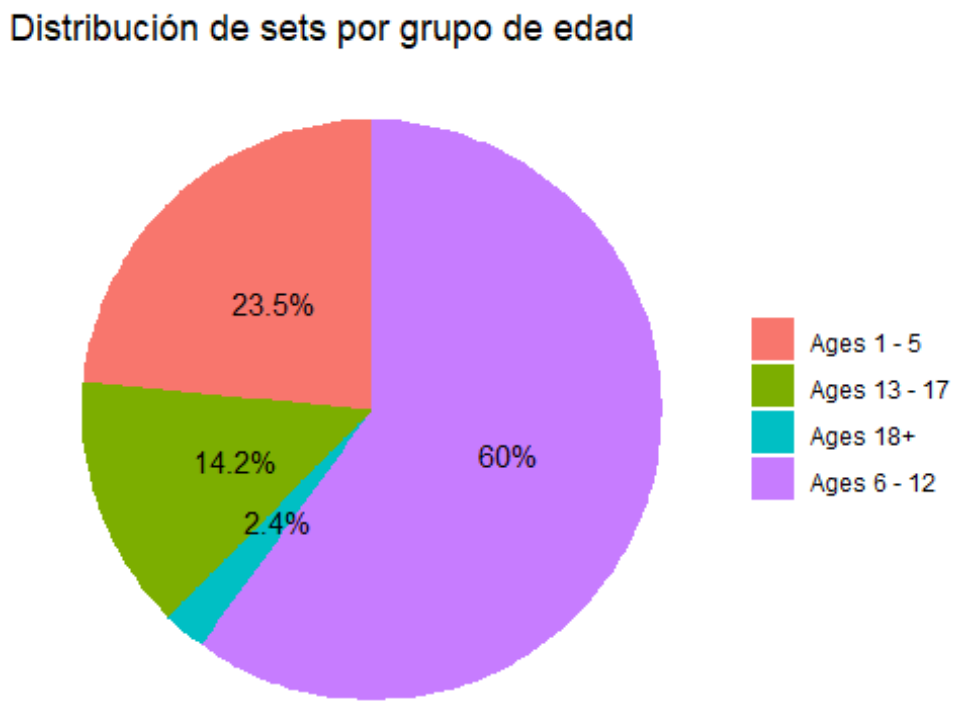
Existe una correlación moderada entre piezas y minifiguras (0.574). Por lo general podemos esperar más minifiguras cuantas más piezas tenga el set.

Edad del público de LEGO y su relación con el número de piezas

- ¿Cuál es el público al que más se dirige LEGO con sus sets? ¿Se relaciona con el número de piezas?

```
setDT(lego_data_clean)
lego_data_clean[, Ages := as.character(Ages)]

# Eliminación de NA y "Ages_NA"
lego_data_clean <- lego_data_clean[!is.na(Ages) & Ages != "Ages_NA"]

# Reagrupación para facilitar el estudio
lego_data_clean[, new_age_group := fifelse(
  Ages %in% c("Ages_1+", "Ages_2+", "Ages_1 - 3", "Ages_2 - 5", "Ages_4 -
99", "Ages_2 - 6",
              "Ages_5 - 99", "Ages_4+", "Ages_3+", "Ages_5 - 12",
"Ages_5+", "Ages_3 - 6",
              "Ages_4 - 6", "Ages_5 - 10"),
  "Ages 1 - 5",
  fifelse(
    Ages %in% c("Ages_6+", "Ages_7+", "Ages_6 - 12", "Ages_4 - 7",
"Ages_6 - 14", "Ages_11+",
                "Ages_7 - 14", "Ages_6 - 10", "Ages_8+", "Ages_9+",
"Ages_7 - 12", "Ages_8 - 12",
```

```
                "Ages_9 - 14", "Ages_9 - 12", "Ages_8 - 14"),
    "Ages 6 - 12",
    fifelse(
      Ages %in% c("Ages_13+", "Ages_12+", "Ages_14+", "Ages_16+",
"Ages_10+"),
      "Ages 13 - 17",
      fifelse(Ages %in% c("Ages_18+"), "Ages 18+", "Otro")
    )
  )
)]

# Conteo y media
lego_count_ages <- lego_data_clean[, .(
  mean_pieces = mean(Pieces, na.rm = TRUE),
  set_count = .N
), by = new_age_group]

# Frecuencia relativa
lego_count_ages[, set_count_freq := (set_count / sum(set_count)) * 100]

# Ordenar por media de piezas en orden descendiente
setorder(lego_count_ages, -mean_pieces)

lego_count_ages

##    new_age_group mean_pieces set_count set_count_freq
##            <char>       <num>     <int>          <num>
## 1:      Ages 18+   2370.4783        23       2.358974
## 2:  Ages 13 - 17    912.6449       138      14.153846
## 3:   Ages 6 - 12    367.2483       585      60.000000
## 4:    Ages 1 - 5    147.0349       229      23.487179

# Diagrama de sectores

ggplot(lego_count_ages, aes(x = "", y = set_count_freq, fill =
new_age_group)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  theme_void() +
  labs(title = "Distribución de sets por grupo de edad") +
  theme(legend.title = element_blank())+
  geom_text(aes(label = paste0(round(set_count_freq, 1), "%")),
            position = position_stack(vjust = 0.5))
```



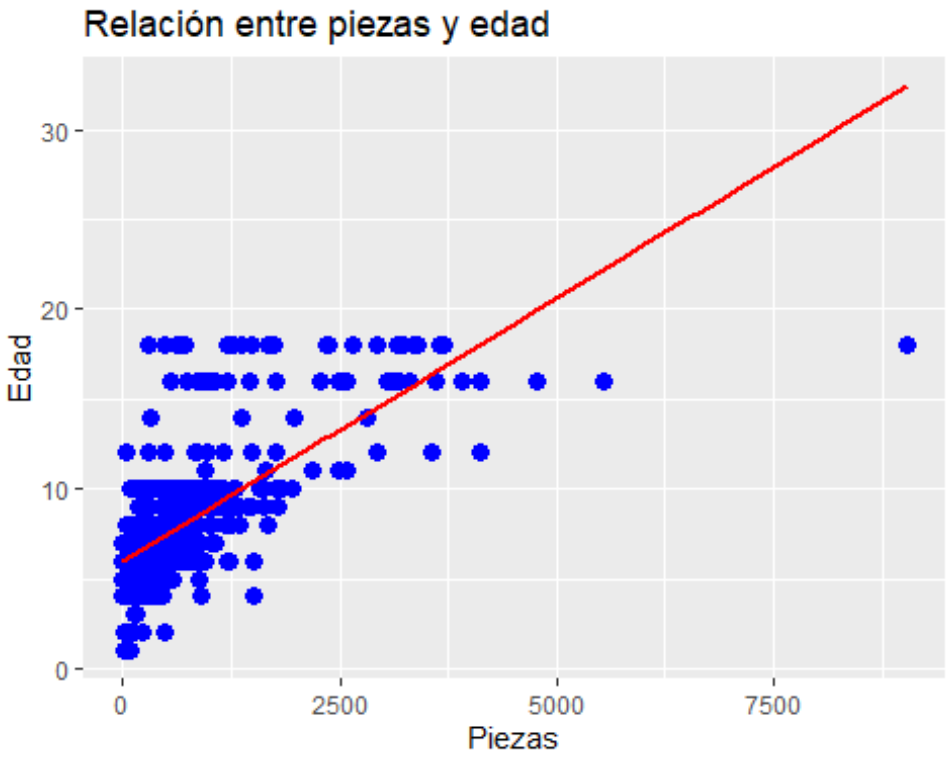
Según el diagrama, vemos que LEGO se enfoca más a un público infantil (edades de 6 a 12 años).

```
# Conversión Ages a números
lego_data_clean[, Age_numeric := as.numeric(gsub("\\D", "", substr(Ages,
6, 7)))]


# Calcular la correlación entre la edad mínima y Pieces, excluyendo NA
correlation_ages_pieces <- cor(lego_data_clean$Age_numeric,
lego_data_clean$Pieces, use = "complete.obs")

correlation_ages_pieces
```

```
## [1] 0.6628583
```

```
# Diagrama de puntos
ggplot(lego_data_clean, aes(x = Pieces, y = Age_numeric)) +
  geom_point(color = "blue", size = 3) +
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  labs(title = "Relación entre piezas y edad",
       x = "Piezas",
       y = "Edad")
```

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

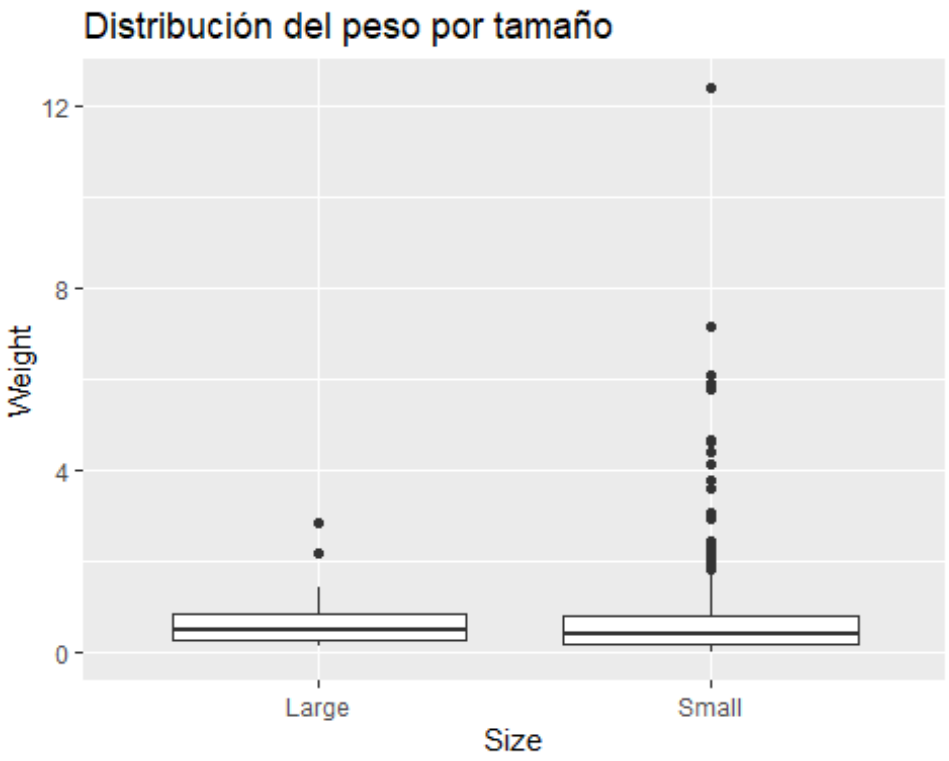


Vemos que la relación entre el número de piezas y la edad mínima de los sets tienen una correlación de 0.66, lo cual es moderadamente positiva. En la tabla anterior, también podemos observar el crecimiento de número de piezas según los grupos de edad.

Relación peso con tamaño

- ¿Cómo se relaciona el peso del set con el tamaño que tiene?

```
# Correlación

lego_data_clean <- lego_data_clean %>%
  mutate(Size_numeric = ifelse(Size == "Small", 1, ifelse(Size ==
"Large", 2, NA)))

correlation_weight_size <- cor(lego_data_clean$Size_numeric,
lego_data_clean$Weight, use = "complete.obs")
correlation_weight_size

## [1] -0.003713084

# Diagrama de caja
ggplot(lego_data_clean, aes(x = Size, y = Weight)) +
  geom_boxplot() +
  labs(title = "Distribución del peso por tamaño", x = "Size", y =
"Weight")
```



De estas dos variables podíamos esperar que el tamaño grande significara más peso, pero no es así. La correlación es prácticamente 0, por lo que podemos obviar estos datos.

Distribución de sets por año

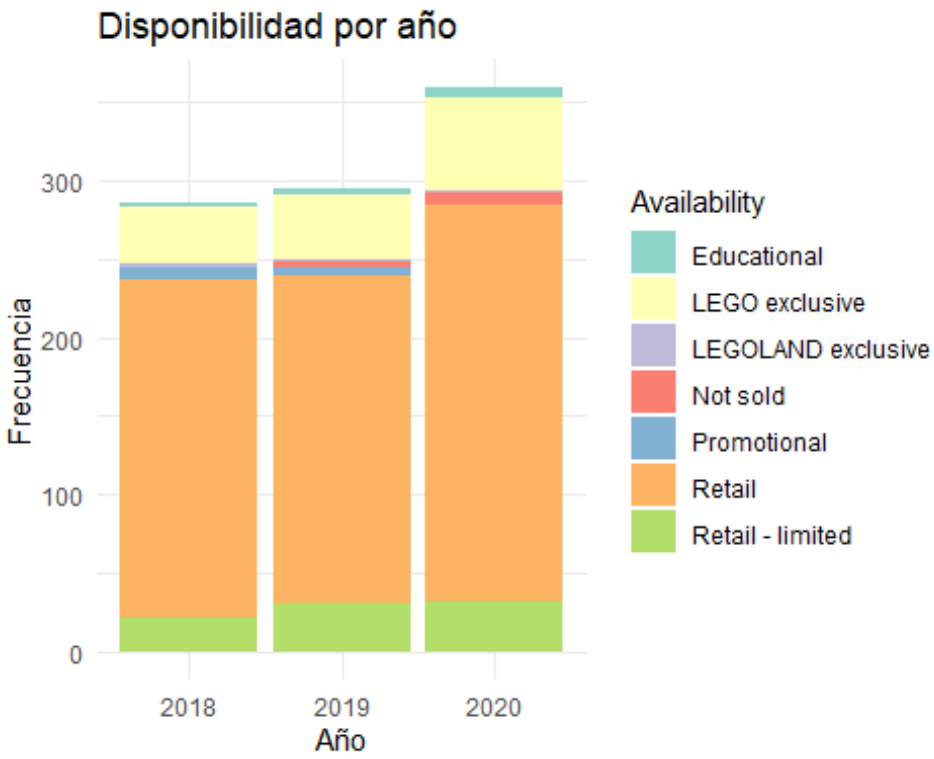
¿Dónde es más probable que nos encontremos un set de LEGO?

```
setDT(lego_data_clean)

# Conteo y eliminación de NA
availability_count <- lego_data_clean[!is.na(Availability), .N, by =
.(Availability)]
availability_count[, freq_relative := N / sum(N)*100]

availability_by_year <- lego_data_clean[!is.na(Availability), .N, by =
.(Year, Availability)]

availability_count
```

```
##          Availability     N freq_relative
##                 <fctr> <int>         <num>
## 1:     LEGO exclusive   135    14.3617021
## 2:             Retail   676    71.9148936
## 3:        Educational    14     1.4893617
## 4:   Retail - limited    85     9.0425532
## 5:        Promotional    12     1.2765957
```

```
## 6:           Not sold    12     1.2765957
## 7: LEGOLAND exclusive     6     0.6382979
```

```r
# Diagrama de barras apiladas
ggplot(availability_by_year, aes(x = Year, y = N, fill = Availability)) +
  geom_bar(stat = "identity") +
  labs(title = "Disponibilidad por año", x = "Año", y = "Frecuencia") +
  theme_minimal() +
  scale_fill_brewer(palette = "Set3")
```



Vemos que la disponibilidad en el último año de los datos es mayor que la de años anteriores, y que la gran mayoría de sets de LEGO se pueden comprar en tiendas (un 71,91%)