# Review de Apps en Google Playstore

Fernando Grau Rivero

UCamp - Utel

# Programa

- Objetivos
- Generación del Data set
- Análisis General de Datos
- Preprocesado de Datos
- Limpieza de Datos
- Análisis Exploratorio de Datos (EDA)
- Preparación para el modelo
- Aplicación de Modelos
- Conclusión

# Objetivos

# Generación del Data Set

```
df = pd.read_csv('/content/drive/MyDrive/1. UCamp - Ciencias de Datos/Modulo 7 - Técnicas Avanzadas de Cs Datos/Proyecto M7/googleplaystore.csv')

df.head(20)
```

| | App | Category | Rating | Reviews | Size | Installs | Type | Price | Content Rating | Genres | Last Updated | Current Ver | Android Ver |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Photo Editor & Candy Camera & Grid & ScrapBook | ART_AND_DESIGN | 4.1 | 159 | 19M | 10,000+ | Free | 0 | Everyone | Art & Design | January 7, 2018 | 1.0.0 | 4.0.3 and up |
| 1 | Coloring book moana | ART_AND_DESIGN | 3.9 | 967 | 14M | 500,000+ | Free | 0 | Everyone | Art & Design;Pretend Play | January 15, 2018 | 2.0.0 | 4.0.3 and up |
| 2 | U Launcher Lite – FREE Live Cool Themes, Hide ... | ART_AND_DESIGN | 4.7 | 87510 | 8.7M | 5,000,000+ | Free | 0 | Everyone | Art & Design | August 1, 2018 | 1.2.4 | 4.0.3 and up |
| 3 | Sketch - Draw & Paint | ART_AND_DESIGN | 4.5 | 215644 | 25M | 50,000,000+ | Free | 0 | Teen | Art & Design | June 8, 2018 | Varies with device | 4.2 and up |
| 4 | Pixel Draw - Number Art Coloring Book | ART_AND_DESIGN | 4.3 | 967 | 2.8M | 100,000+ | Free | 0 | Everyone | Art & Design;Creativity | June 20, 2018 | 1.1 | 4.4 and up |
| 5 | Paper flowers instructions | ART_AND_DESIGN | 4.4 | 167 | 5.6M | 50,000+ | Free | 0 | Everyone | Art & Design | March 26, 2017 | 1.0 | 2.3 and up |
| 6 | Smoke Effect Photo Maker - Smoke Editor | ART_AND_DESIGN | 3.8 | 178 | 19M | 50,000+ | Free | 0 | Everyone | Art & Design | April 26, 2018 | 1.1 | 4.0.3 and up |
| 7 | Infinite Painter | ART_AND_DESIGN | 4.1 | 36815 | 29M | 1,000,000+ | Free | 0 | Everyone | Art & Design | June 14, 2018 | 6.1.61.1 | 4.2 and up |
| 8 | Garden Coloring Book | ART_AND_DESIGN | 4.4 | 13791 | 33M | 1,000,000+ | Free | 0 | Everyone | Art & Design | September 20, 2017 | 2.9.2 | 3.0 and up |
| 9 | Kids Paint Free - Drawing Fun | ART_AND_DESIGN | 4.7 | 121 | 3.1M | 10,000+ | Free | 0 | Everyone | Art & Design;Creativity | July 3, 2018 | 2.8 | 4.0.3 and up |
| 10 | Text on Photo - Fonteee | ART_AND_DESIGN | 4.4 | 13880 | 28M | 1,000,000+ | Free | 0 | Everyone | Art & Design | October 27, 2017 | 1.0.4 | 4.1 and up |

# Análisis General de Datos

Volver a la página de agenda

# Valores

## ¿Cuántos registros (o filas con datos) hay?

Como podemos ver tenemos un total de 10841 registros que constan de 13 atributos(columnas)

## Información de las Columnas

- **App:** El nombre de la aplicación
- **Category:** La categoría de la aplicación
- **Rating:** La calificación de la aplicación en la Play Store
- **Reviews:** El número de reseñas de la aplicación
- **Size:** El tamaño de la aplicación
- **Install:** El número de instalaciones de la aplicación
- **Type:** El tipo de la aplicación (Gratis/Pagado)
- **Price:** El precio de la aplicación (0 si es Gratis)
- **Content Rating:** El público objetivo apropiado de la aplicación
- **Genres:** El género de la aplicación
- **Last Updated:** La fecha en que la aplicación fue actualizada por última vez
- **Current Ver:** La versión actual de la aplicación
- **Android Ver:** La versión mínima de Android requerida para ejecutar la aplicació

Volver a la página de agenda

# Valores

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10841 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   App             10841 non-null  object
 1   Category        10841 non-null  object
 2   Rating          9367 non-null   float64
 3   Reviews         10841 non-null  object
 4   Size            10841 non-null  object
 5   Installs        10841 non-null  object
 6   Type            10840 non-null  object
 7   Price           10841 non-null  object
 8   Content Rating  10840 non-null  object
 9   Genres          10841 non-null  object
 10  Last Updated    10841 non-null  object
 11  Current Ver     10833 non-null  object
 12  Android Ver     10838 non-null  object
dtypes: float64(1), object(12)
memory usage: 1.1+ MB
```

Volver a la página de agenda

# Preprocesado de Datos

# 1. Columna “Reviews” (número de reseñas)

Verificamos si exite un valor no numerico en este columna

```
df[~df.Reviews.str.isnumeric()]
```

| | App | Category | Rating | Reviews | Size | Installs | Type | Price | Content Rating | Genres | Last Updated | Current Ver | Android Ver |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **10472** | Life Made WI-Fi Touchscreen Photo Frame | 1.9 | 19.0 | 3.0M | 1,000+ | Free | 0 | Everyone | NaN | February 11, 2018 | 1.0.19 | 4.0 and up | NaN |

- Eliminamos este registro
- Convertimos las variables en numeros enteros (int)

# 2. Columna “Size”(Peso)

```
df['Size'].unique()
```

```
array(['19M', '14M', '8.7M', '25M', '2.8M', '5.6M', '29M', '33M', '3.1M',
       '28M', '12M', '20M', '21M', '37M', '2.7M', '5.5M', '17M', '39M',
       '31M', '4.2M', '7.0M', '23M', '6.0M', '6.1M', '4.6M', '9.2M',
       '5.2M', '11M', '24M', 'Varies with device', '9.4M', '15M', '10M',
       '1.2M', '26M', '8.0M', '7.9M', '56M', '57M', '35M', '54M', '201k',
       '3.6M', '5.7M', '8.6M', '2.4M', '27M', '2.5M', '16M', '3.4M',
       '8.9M', '3.9M', '2.9M', '38M', '32M', '5.4M', '18M', '1.1M',
       '2.2M', '4.5M', '9.8M', '52M', '9.0M', '6.7M', '30M', '2.6M',
       '7.1M', '3.7M', '22M', '7.4M', '6.4M', '3.2M', '8.2M', '9.9M',
       '4.9M', '9.5M', '5.0M', '5.9M', '13M', '73M', '6.8M', '3.5M',
       '4.0M', '2.3M', '7.2M', '2.1M', '42M', '7.3M', '9.1M', '55M',
       '23k', '6.5M', '1.5M', '7.5M', '51M', '41M', '48M', '8.5M', '46M',
       '8.3M', '4.3M', '4.7M', '3.3M', '40M', '7.8M', '8.8M', '6.6M',
       '5.1M', '61M', '66M', '79k', '8.4M', '118k', '44M', '695k', '1.6M',
       '6.2M', '18k', '53M', '1.4M', '3.0M', '5.8M', '3.8M', '9.6M',
       '45M', '63M', '49M', '77M', '4.4M', '4.8M', '70M', '6.9M', '9.3M',
       '10.0M', '8.1M', '36M', '84M', '97M', '2.0M', '1.9M', '1.8M',
       '5.3M', '47M', '556k', '526k', '76M', '7.6M', '59M', '9.7M', '78M',
       '72M', '43M', '7.7M', '6.3M', '334k', '34M', '93M', '65M', '79M',
       '100M', '58M', '50M', '68M', '64M', '67M', '60M', '94M', '232k',
       '99M', '624k', '95M', '8.5k', '41k', '292k', '11k', '80M', '1.7M',
       '74M', '62M', '69M', '75M', '98M', '85M', '82M', '96M', '87M',
       '71M', '86M', '91M', '81M', '92M', '83M', '88M', '704k', '862k',
       '899k', '378k', '266k', '375k', '1.3M', '975k', '980k', '4.1M',
       '89M', '696k', '544k', '525k', '920k', '779k', '853k', '720k',
       '713k', '772k', '318k', '58k', '241k', '196k', '857k', '51k',
```

- Observamos que los datos contienen prefijos (kilo y Mega). Procederemos a reemplazar 'k' y 'M' con sus valores correspondientes para convertir los datos en valores numéricos.

- Los datos de la comlumna 'Size' debe ser de tipo flotante.

Al completar la eliminación de los prefijos procedimos a convertir todos los valores en megabytes

# 3. Columna “Install” y “Price” (N. Intslaciones y Precio)

```
df['Installs'].unique()
```

```
array(['10,000+', '500,000+', '5,000,000+', '50,000,000+', '100,000+',
       '50,000+', '1,000,000+', '10,000,000+', '5,000+', '100,000,000+',
       '1,000,000,000+', '1,000+', '500,000,000+', '50+', '100+', '500+',
       '10+', '1+', '5+', '0+', '0'], dtype=object)
```

```
[21] df['Price'].unique()
```

```
array(['0', '$4.99', '$3.99', '$6.99', '$1.49', '$2.99', '$7.99', '$5.99',
       '$3.49', '$1.99', '$9.99', '$7.49', '$0.99', '$9.00', '$5.49',
       '$10.00', '$24.99', '$11.99', '$79.99', '$16.99', '$14.99',
       '$1.00', '$29.99', '$12.99', '$2.49', '$10.99', '$1.50', '$19.99',
       '$15.99', '$33.99', '$74.99', '$39.99', '$3.95', '$4.49', '$1.70',
       '$8.99', '$2.00', '$3.88', '$25.99', '$399.99', '$17.99',
       '$400.00', '$3.02', '$1.76', '$4.84', '$4.77', '$1.61', '$2.50',
       '$1.59', '$6.49', '$1.29', '$5.00', '$13.99', '$299.99', '$379.99',
       '$37.99', '$18.99', '$389.99', '$19.90', '$8.49', '$1.75',
       '$14.00', '$4.85', '$46.99', '$109.99', '$154.99', '$3.08',
       '$2.59', '$4.80', '$1.96', '$19.40', '$3.90', '$4.59', '$15.46',
       '$3.04', '$4.29', '$2.60', '$3.28', '$4.60', '$28.99', '$2.95',
       '$2.90', '$1.97', '$200.00', '$89.99', '$2.56', '$30.99', '$3.61',
       '$394.99', '$1.26', '$1.20', '$1.04'], dtype=object)
```

- Los valores de la columna "Install" debe ser de tipo numérico entero (int).

- Se deben eliminar los caracteres ',' y '+'. / Ejemplo: '10 000+' a 10 000

- Los datos de la columna "Price" debe ser de tipo flotante.

- El sufijo de la moneda debe eliminarse si el precio es distinto de cero. Ejemplo: '$4,99' a 4,99

# 4. Columna “Last Updated” (Fecha de última actualización)

```
# Cambiamos la columna a formato fecha-hora
df['Last Updated'] = pd.to_datetime(df['Last Updated'])
df['Last Updated']
```

```
0       2018-01-07
1       2018-01-15
2       2018-08-01
3       2018-06-08
4       2018-06-20
           ...
10836   2017-07-25
10837   2018-07-06
10838   2017-01-20
10839   2015-01-19
10840   2018-07-25
Name: Last Updated, Length: 10840, dtype: datetime64[ns]
```

Es necesario hacer la actualización del tipo de dato de la columna de "Object" a fecha y hora de la librería de Pandas

- De igual manera se crearan dos nuevas columnas:

1) "Updated_month" para indicar el mes en el que fué actualizada la app

2) "Updated_Year" para indicar el año en el que fué actualizada la app

```
[27] df['Updated_Month']=df['Last Updated'].dt.month
     df['Updated_Year']=df['Last Updated'].dt.year
```

Eliminamos la columna “Last Updated”

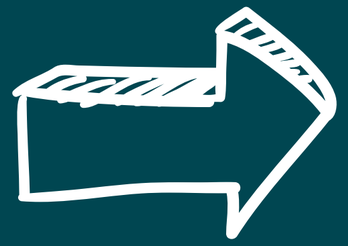
# Data set después del Preprocesado

| | App | Category | Rating | Reviews | Size | Installs | Type | Price | Content Rating | Genres | Current Ver | Android Ver | Updated_Month | Updated_Year |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Photo Editor & Candy Camera & Grid & ScrapBook | ART_AND_DESIGN | 4.1 | 159 | 19.0 | 10000 | Free | 0.0 | Everyone | Art & Design | 1.0.0 | 4.0.3 and up | 1 | 2018 |
| 1 | Coloring book moana | ART_AND_DESIGN | 3.9 | 967 | 14.0 | 500000 | Free | 0.0 | Everyone | Art & Design;Pretend Play | 2.0.0 | 4.0.3 and up | 1 | 2018 |
| 2 | U Launcher Lite – FREE Live Cool Themes, Hide ... | ART_AND_DESIGN | 4.7 | 87510 | 8.7 | 5000000 | Free | 0.0 | Everyone | Art & Design | 1.2.4 | 4.0.3 and up | 8 | 2018 |
| 3 | Sketch - Draw & Paint | ART_AND_DESIGN | 4.5 | 215644 | 25.0 | 50000000 | Free | 0.0 | Teen | Art & Design | Varies with device | 4.2 and up | 6 | 2018 |
| 4 | Pixel Draw - Number Art Coloring Book | ART_AND_DESIGN | 4.3 | 967 | 2.8 | 100000 | Free | 0.0 | Everyone | Art & Design;Creativity | 1.1 | 4.4 and up | 6 | 2018 |

Visual ldel data set con el procesado

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10840 entries, 0 to 10840
Data columns (total 14 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             10840 non-null  object 
 1   Category        10840 non-null  object 
 2   Rating          9366 non-null   float64
 3   Reviews         10840 non-null  int64  
 4   Size            9145 non-null   float64
 5   Installs        10840 non-null  int64  
 6   Type            10839 non-null  object 
 7   Price           10840 non-null  float64
 8   Content Rating  10840 non-null  object 
 9   Genres          10840 non-null  object 
 10  Current Ver     10832 non-null  object 
 11  Android Ver     10838 non-null  object 
 12  Updated_Month   10840 non-null  int64  
 13  Updated_Year    10840 non-null  int64  
dtypes: float64(3), int64(4), object(7)
memory usage: 1.5+ MB
```

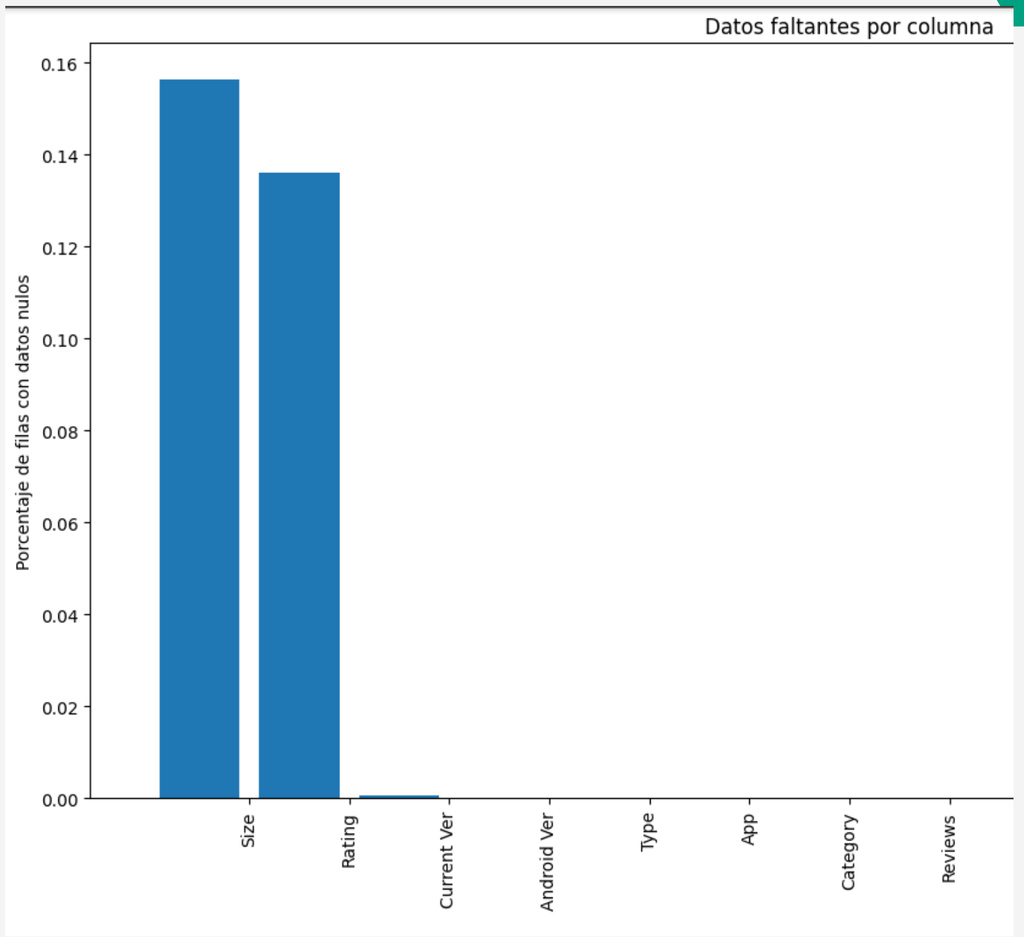


Tipo de Variables

# Limpieza de Datos

# Valores Nulos

| | Valores Nulos | Porcentaje de Valores Nulos |
|---|---|---|
| Size | 1695 | 15.636531 |
| Rating | 1474 | 13.597786 |
| Current Ver | 8 | 0.073801 |
| Android Ver | 2 | 0.018450 |
| Type | 1 | 0.009225 |
| App | 0 | 0.000000 |
| Category | 0 | 0.000000 |
| Reviews | 0 | 0.000000 |
| Installs | 0 | 0.000000 |
| Price | 0 | 0.000000 |
| Content Rating | 0 | 0.000000 |
| Genres | 0 | 0.000000 |
| Updated_Month | 0 | 0.000000 |
| Updated_Year | 0 | 0.000000 |



# Imputación de valores

En este caso haremos una imputación de los valores para las columnas "Size" y "Rating" usando la media, de esta manera se mantendrá una mejor distribución de los valores en la columna.

**RATING**

La media (Promedio) de los valores en la columna es de 4.2

**SIZE**

La media (Promedio) de los valores en la columna es de 13 Megas

De igual manera para la columna "Type" haremos la imputación bajo la moda.

**TYPE**

La moda de los valores en la columna es el ‘Free’ indicando que las apps no son pagas

# Imputación de valores adicionales

**Los valores de las versiones no se harán ninguna imputación por ahora, dado que más adelante se hará un estudio más profundo para determinar la relevancia de estas columnas en el estudio.**

---

## VALORES IMPUTADOS

```
df.isnull().sum()
```

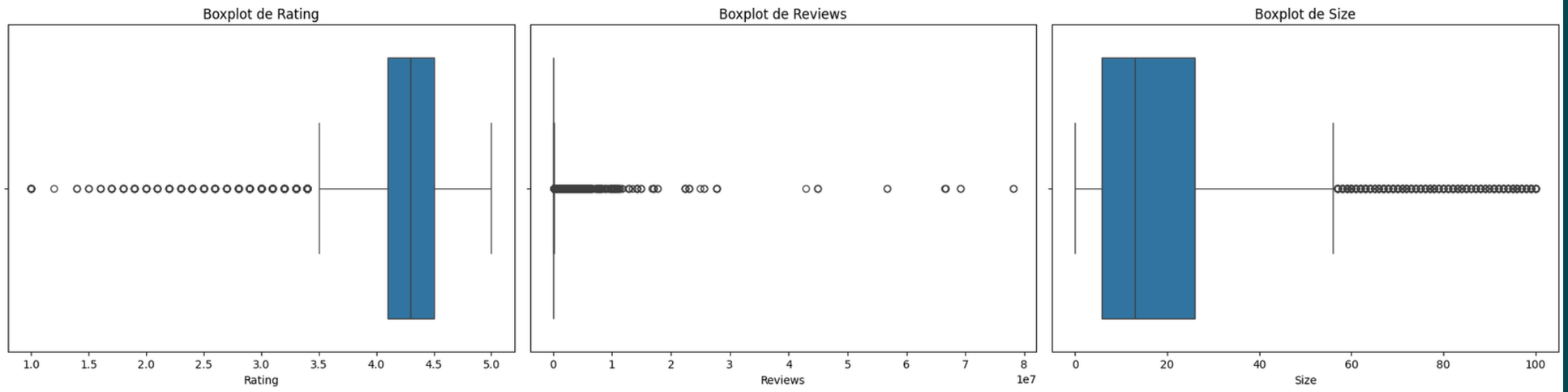
```
App               0
Category          0
Rating            0
Reviews           0
Size              0
Installs          0
Type              0
Price             0
Content Rating    0
Genres            0
Current Ver       8
Android Ver       2
Updated_Month     0
Updated_Year      0
dtype: int64
```

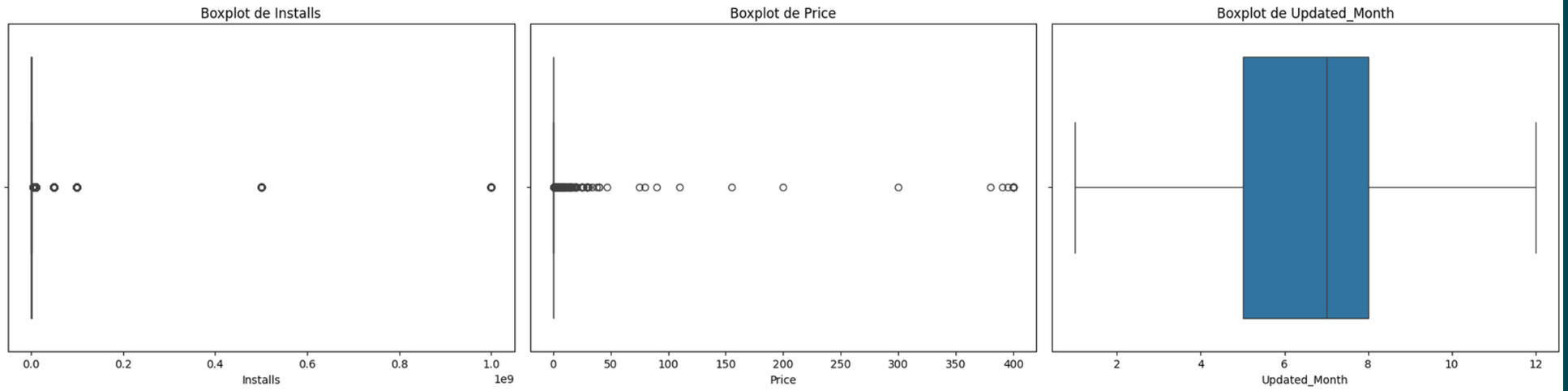
```
df.info()
```

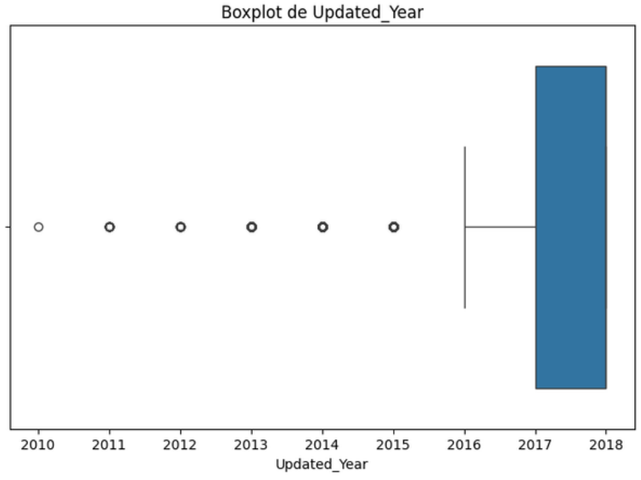
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10840 entries, 0 to 10840
Data columns (total 14 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   App             10840 non-null  object
 1   Category        10840 non-null  object
 2   Rating          10840 non-null  float64
 3   Reviews         10840 non-null  int64
 4   Size            10840 non-null  float64
 5   Installs        10840 non-null  int64
 6   Type            10839 non-null  object
 7   Price           10840 non-null  float64
 8   Content Rating  10840 non-null  object
 9   Genres          10840 non-null  object
 10  Current Ver     10832 non-null  object
 11  Android Ver     10838 non-null  object
 12  Updated_Month   10840 non-null  int64
 13  Updated_Year    10840 non-null  int64
dtypes: float64(3), int64(4), object(7)
memory usage: 1.5+ MB
```

# Análisis de Valores atípicos

# Columnas con valores cuantitativos




Dado que los datos atípicos son representativos de una parte legítima del conjunto de datos y no son el resultado de errores de medición o captura, eliminarlos puede afectar la representatividad de la muestra y sesgar los resultados del análisis, por lo tanto no se procederá con la eliminación.

# Análisis Exploratorio de Datos (EDA)

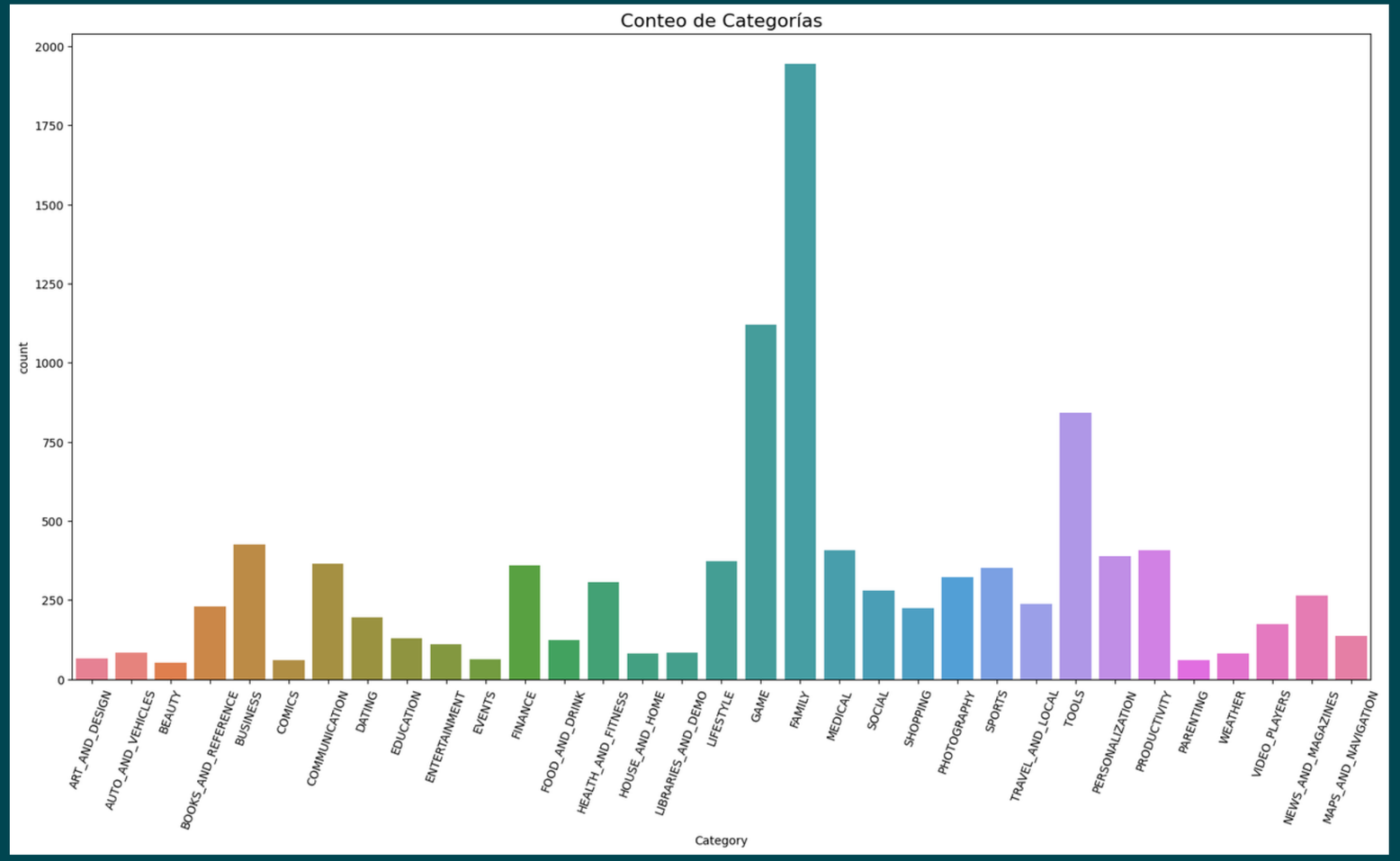
El objetivo principal del EDA es revelar patrones, tendencias, irregularidades, relaciones y características importantes en los datos, lo que ayuda a los científicos de datos a formular hipótesis, tomar decisiones informadas sobre el preprocesamiento de datos y seleccionar las técnicas de modelado adecuadas.

# Columna Categoría (Category)

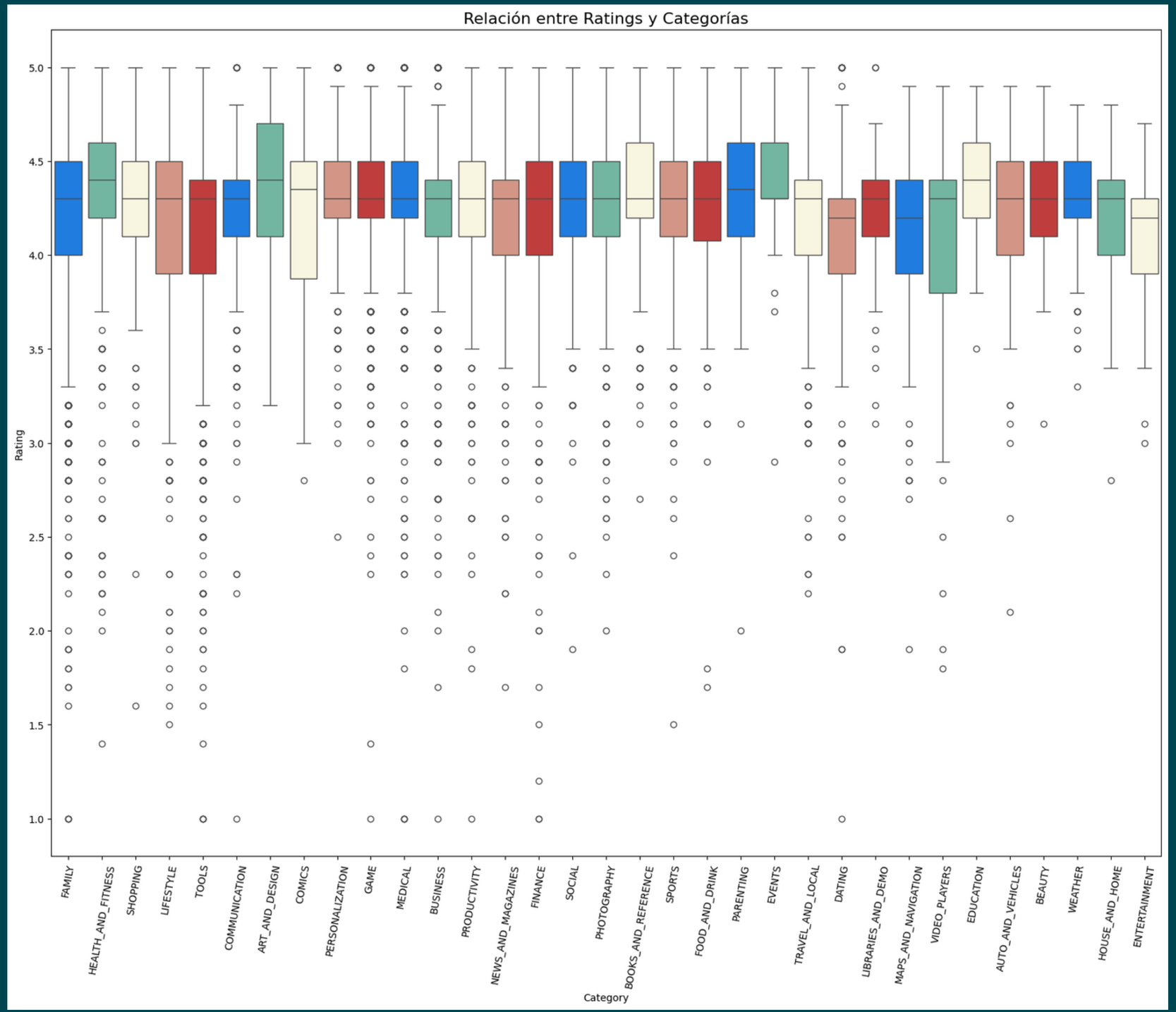
Hay un total de 33 categorías

```
#Verificamos la cantidad de variables en la columna "Category"
df['Category'].value_counts()
```

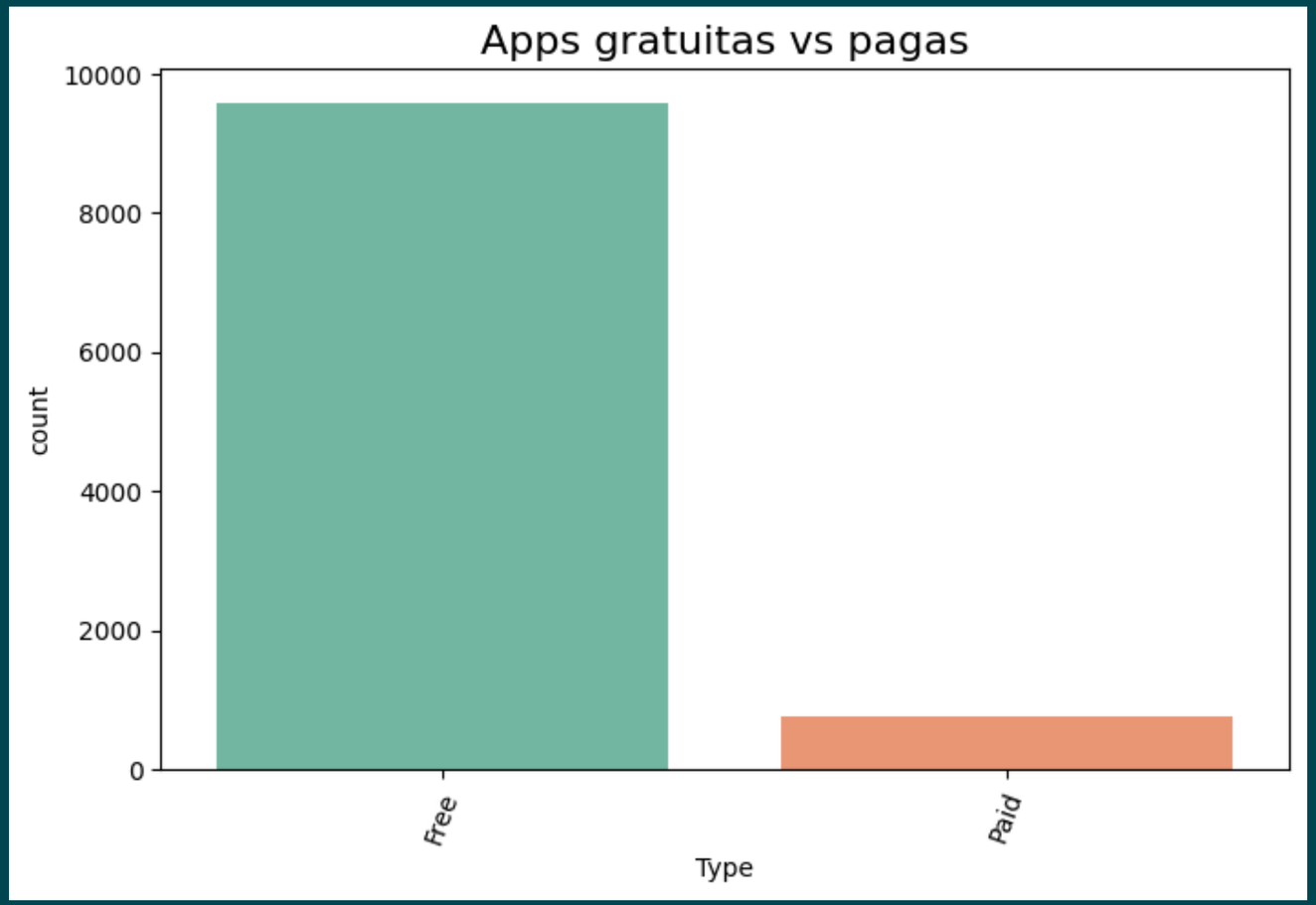
```
FAMILY                 1943
GAME                   1121
TOOLS                   842
BUSINESS                427
MEDICAL                 408
PRODUCTIVITY            407
PERSONALIZATION         388
LIFESTYLE               373
COMMUNICATION           366
FINANCE                 360
SPORTS                  351
PHOTOGRAPHY             322
HEALTH_AND_FITNESS      306
SOCIAL                  280
NEWS_AND_MAGAZINES      264
TRAVEL_AND_LOCAL        237
BOOKS_AND_REFERENCE     230
SHOPPING                224
DATING                  196
VIDEO_PLAYERS           175
MAPS_AND_NAVIGATION     137
EDUCATION               130
FOOD_AND_DRINK          124
ENTERTAINMENT           111
AUTO_AND_VEHICLES        85
LIBRARIES_AND_DEMO       85
WEATHER                  82
HOUSE_AND_HOME           80
ART_AND_DESIGN           65
EVENTS                   64
PARENTING                60
COMICS                   60
BEAUTY                   53
Name: Category, dtype: int64
```

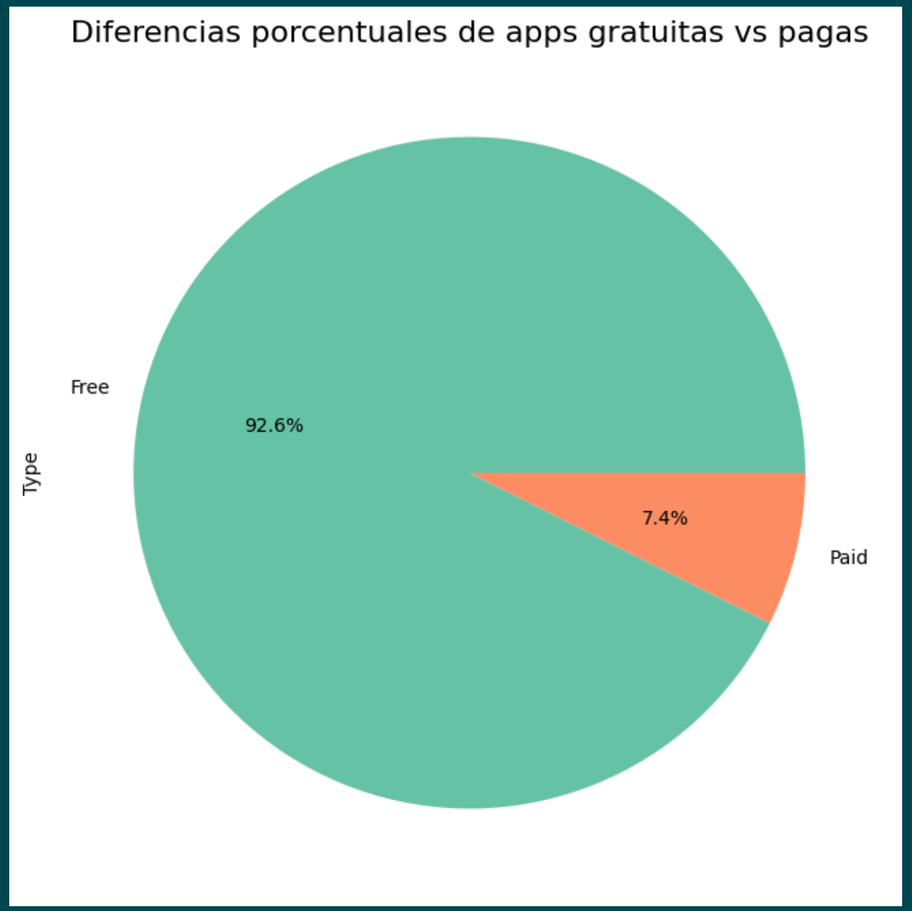


# Relación entre Categoría y Rating

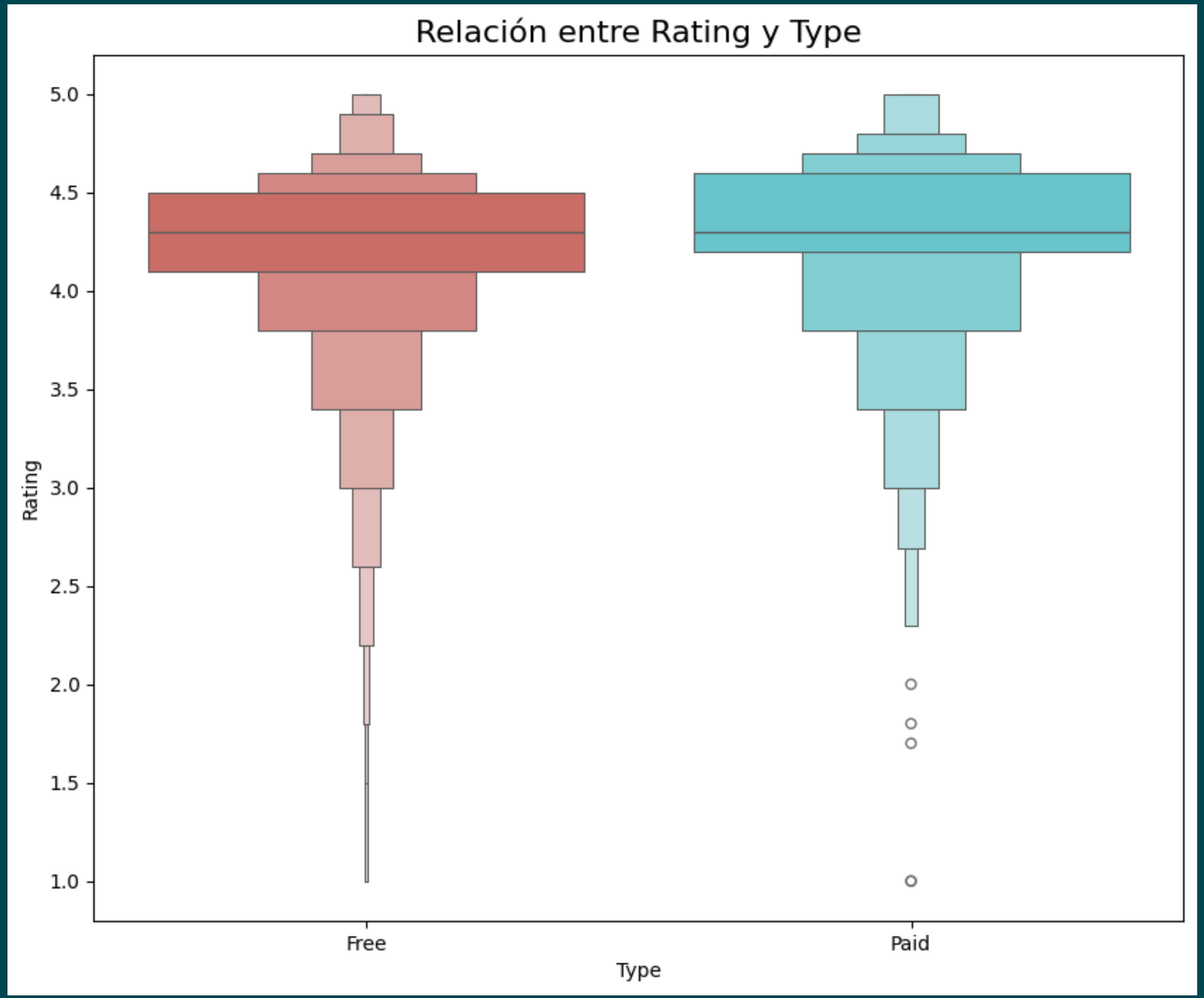


# Columna Tipo de App (Type)

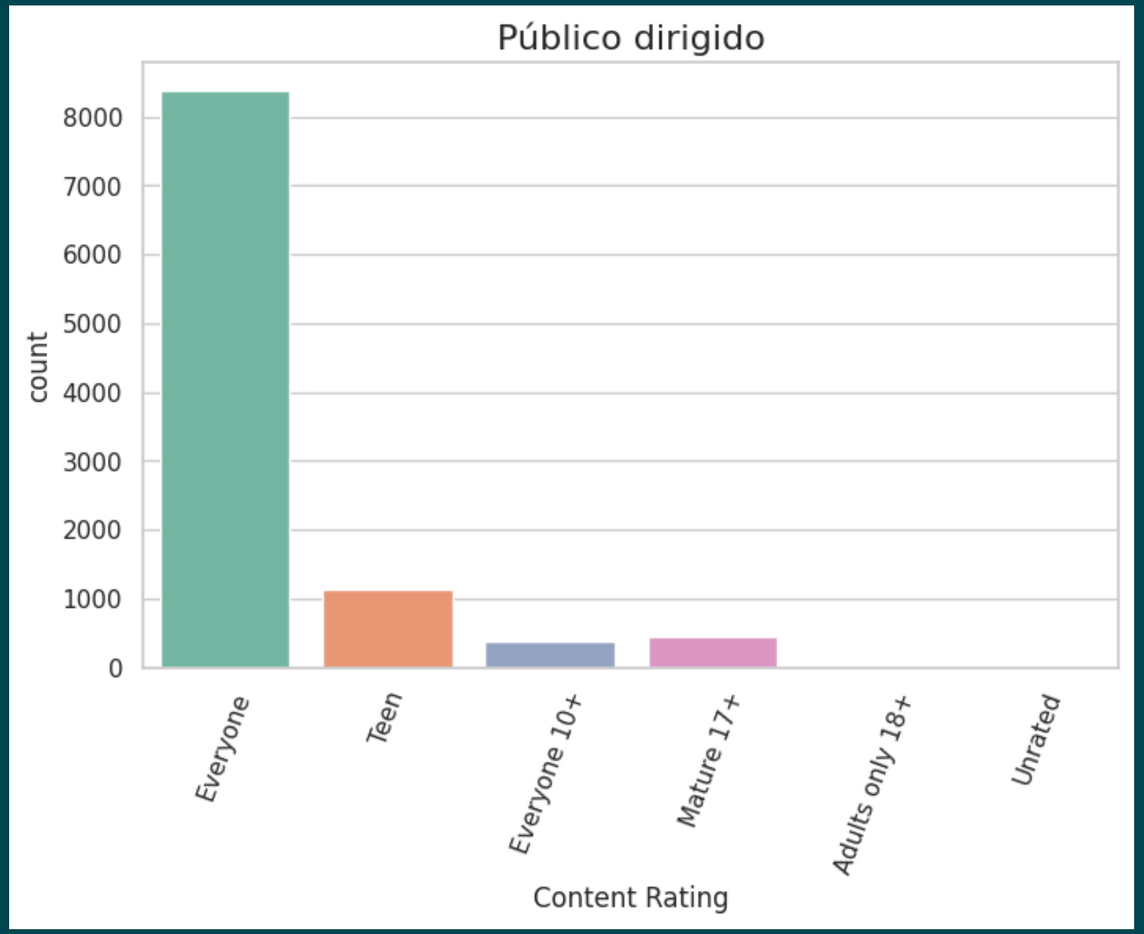


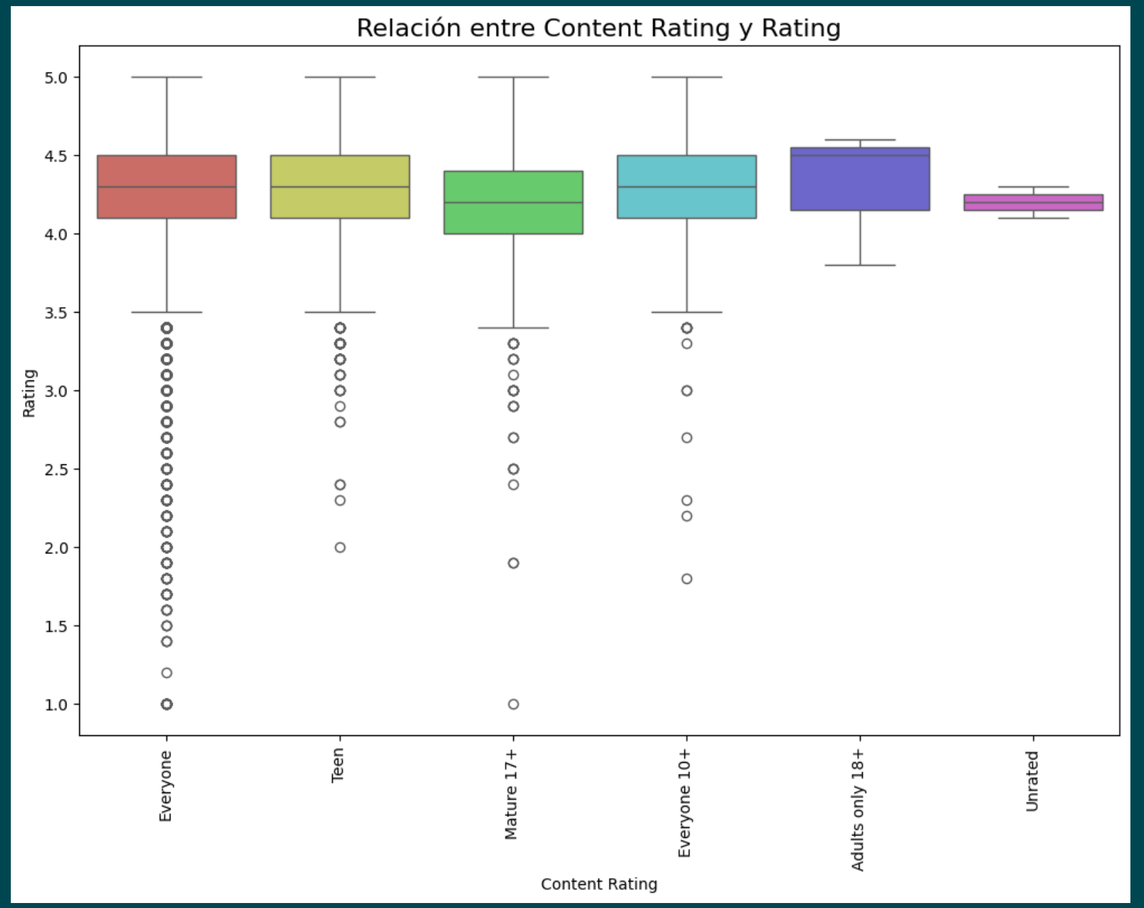
# Relación entre Type y Rating



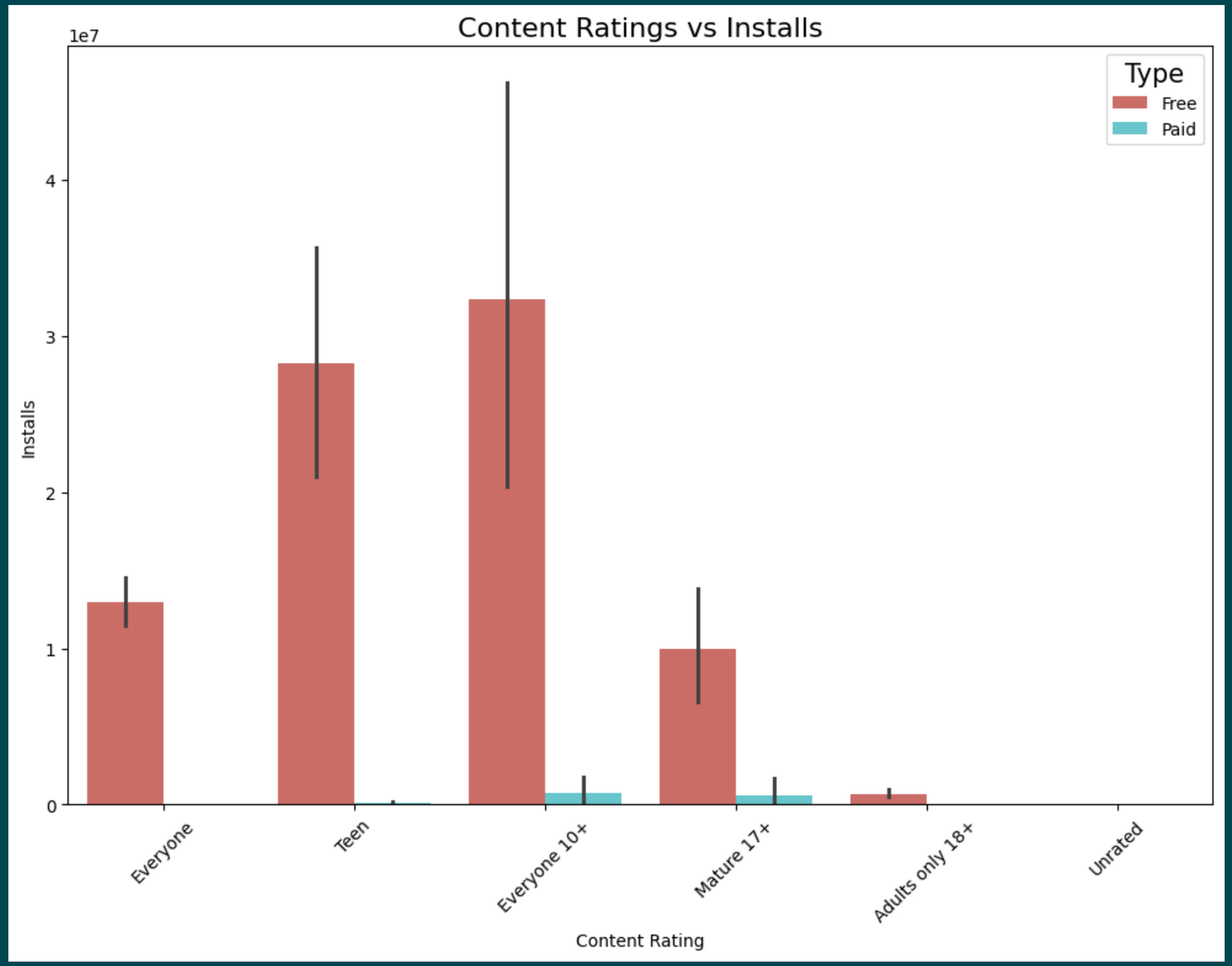
# Columna Público Dirigido (Content Rating)




## Relación entre Content Rating y Rating



# Público Dirigido vs Instalaciones



# Evaluación de otras columnas

## “Géneros”

```
df['Genres'].value_counts().head(30)
```

```
Tools                        841
Entertainment                588
Education                    527
Business                     427
Medical                      408
Productivity                 407
Personalization              388
Lifestyle                    372
Communication                366
Sports                       364
Finance                      360
Action                       356
Photography                  322
Health & Fitness             306
Social                       280
News & Magazines             264
Travel & Local               236
Books & Reference            230
Shopping                     224
Arcade                       218
Simulation                   199
Dating                       196
Casual                       191
Video Players & Editors      173
Maps & Navigation            137
Puzzle                       136
Food & Drink                 124
Role Playing                 109
Strategy                     105
Racing                        98
Name: Genres, dtype: int64
```

## “Versión de la App”

```
df['Current Ver'].value_counts()
```

```
Varies with device    1302
1.0                    802
1.1                    260
1.2                    177
2.0                    149
                      ...
3.18.5                   1
1.3.A.2.9                1
9.9.1.1910               1
7.1.34.28                1
2.0.148.0                1
Name: Current Ver, Length: 2831, dtype: int64
```

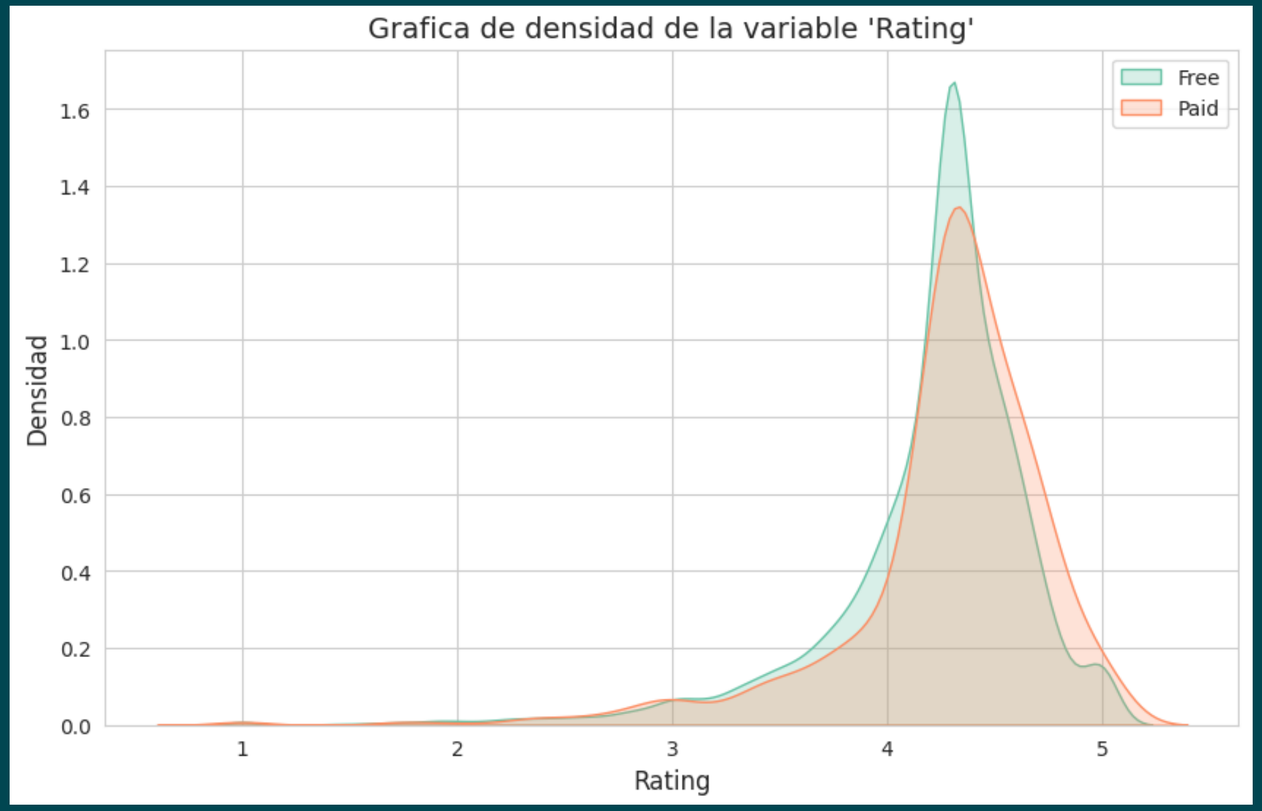
## “Versión de Anndroid”

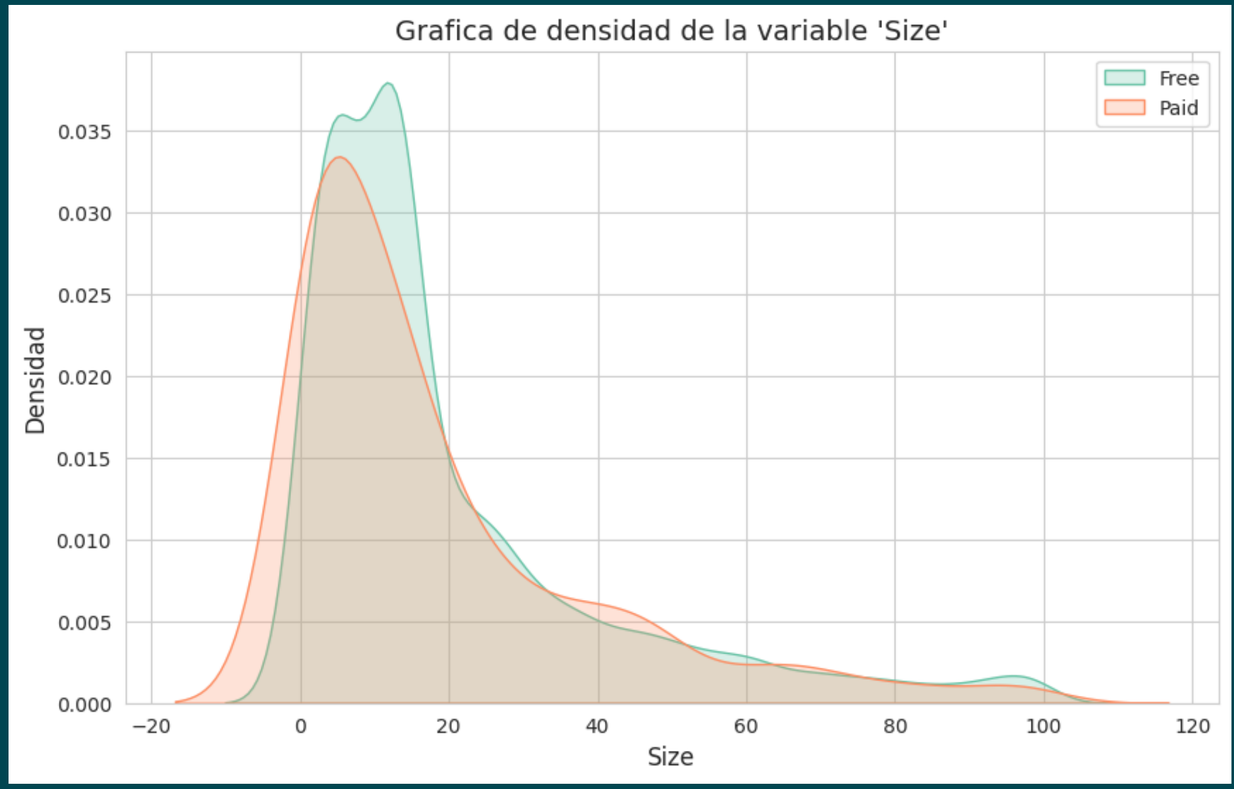
```
df['Android Ver'].value_counts()
```

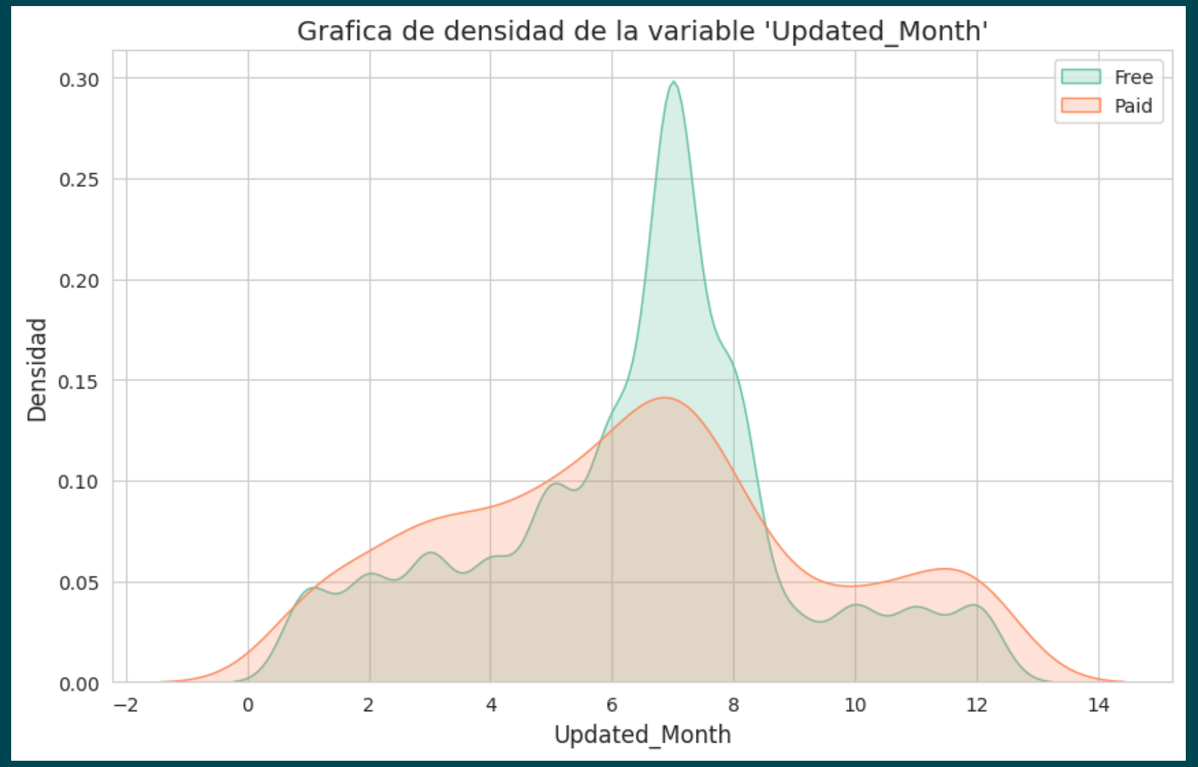
```
4.1 and up            2379
4.0.3 and up          1451
4.0 and up            1337
Varies with device    1221
4.4 and up             893
2.3 and up             643
5.0 and up             546
4.2 and up             387
2.3.3 and up           279
2.2 and up             239
3.0 and up             237
4.3 and up             235
2.1 and up             133
1.6 and up             116
6.0 and up              58
7.0 and up              42
3.2 and up              36
2.0 and up              32
5.1 and up              22
1.5 and up              20
4.4W and up             11
3.1 and up              10
2.0.1 and up             7
8.0 and up               6
7.1 and up               3
4.0.3 - 7.1.1            2
5.0 - 8.0                2
1.0 and up               2
7.0 - 7.1.1              1
4.1 - 7.1.1              1
5.0 - 6.0                1
2.2 - 7.1.1              1
5.0 - 7.1.1              1
Name: Android Ver, dtype: int64
```

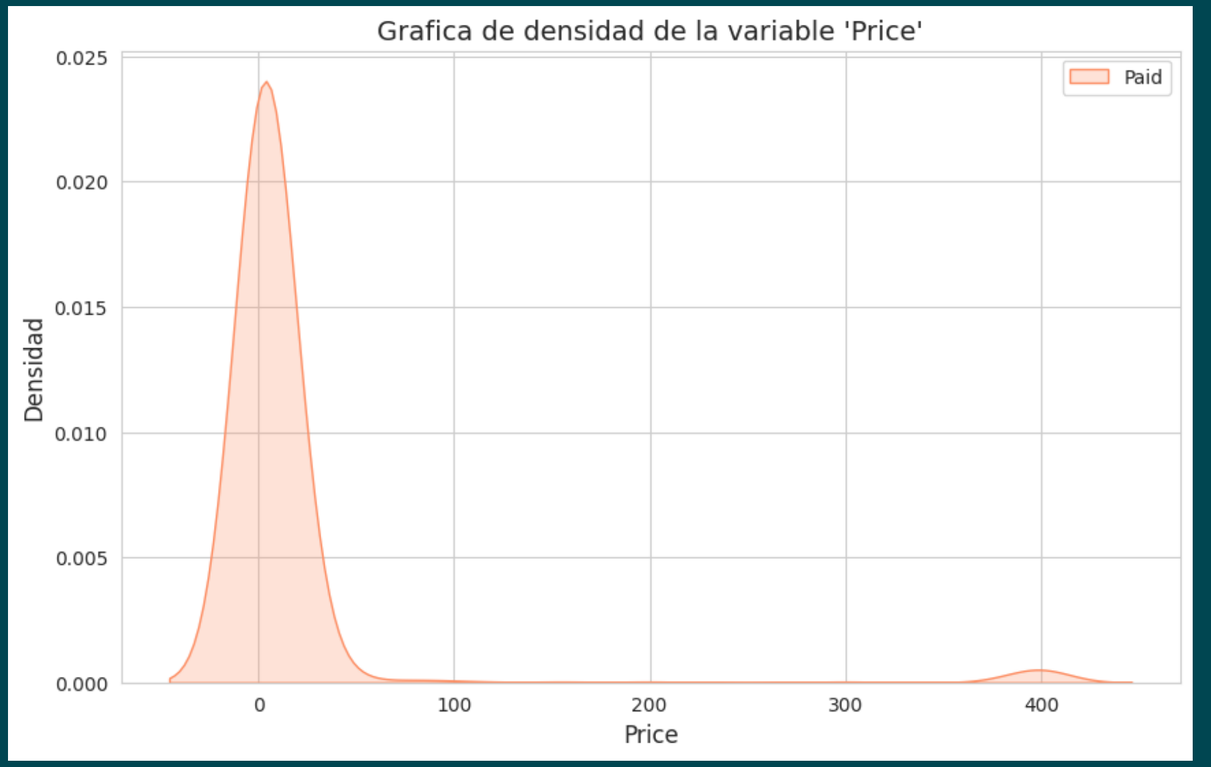
# Gráficas de Densidad

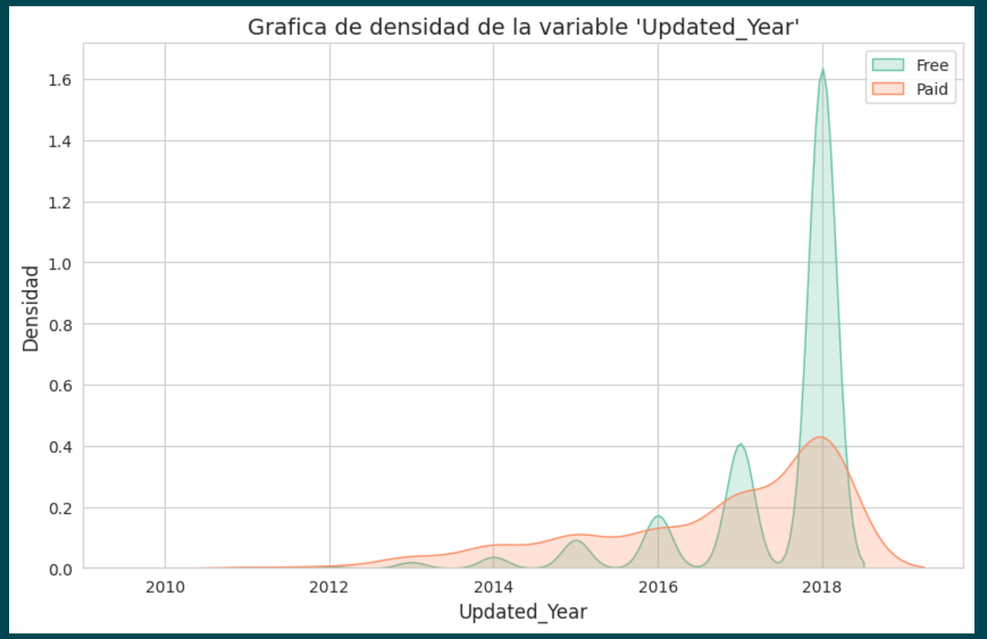
Recordemos que las gráficas de densidad permiten identificar la forma y la dispersión de los datos, así como la presencia de cualquier sesgo o agrupamiento. Esto ayuda a comprender mejor la naturaleza de los datos y a tomar decisiones informadas sobre el análisis estadístico y los modelos predictivos.



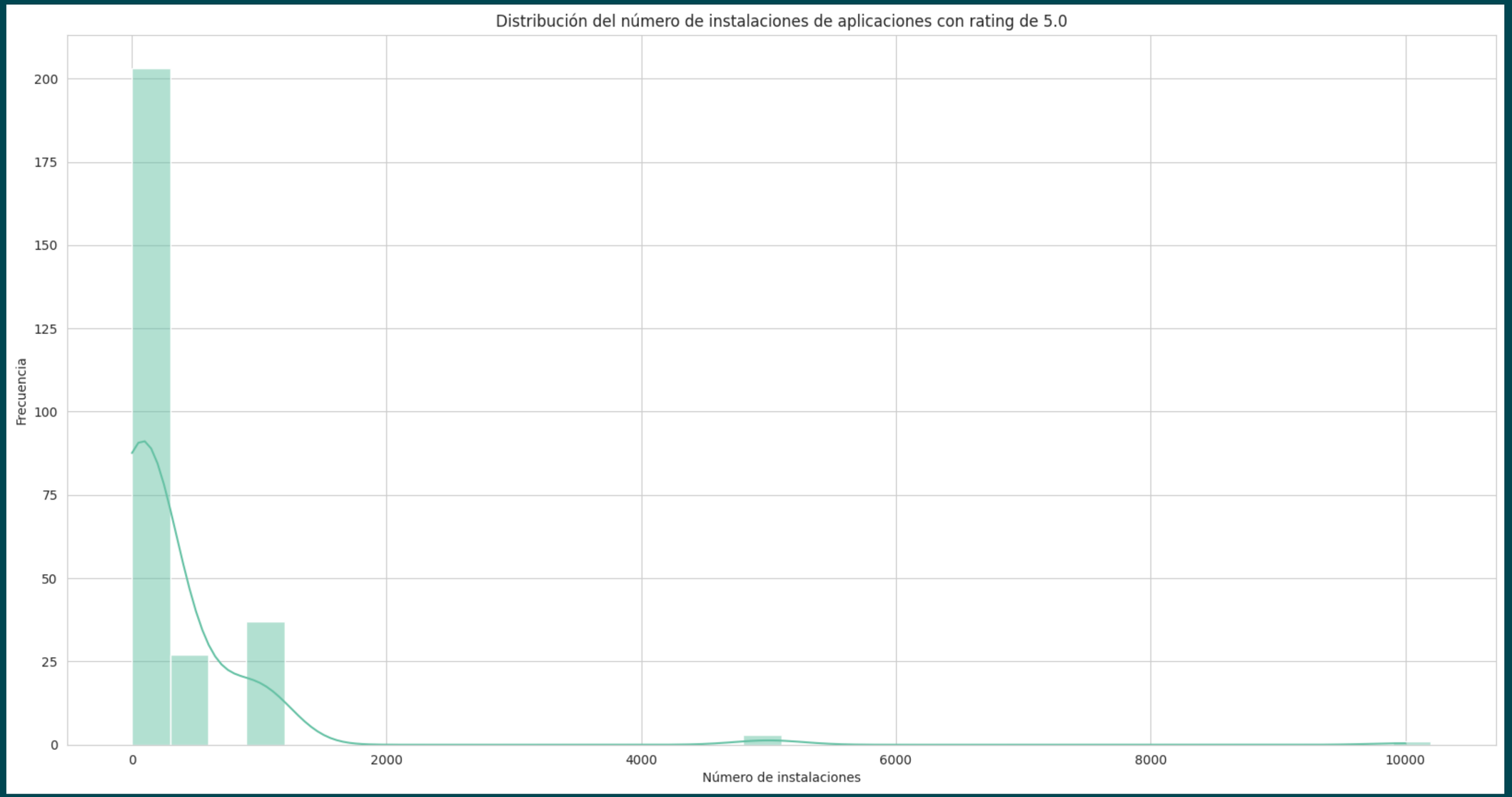




Grafica de densidad de la variable 'Updated_Month'
Free
Paid
Densidad
Updated_Month
0.00
0.05
0.10
0.15
0.20
0.25
0.30
−2
0
2
4
6
8
10
12
14

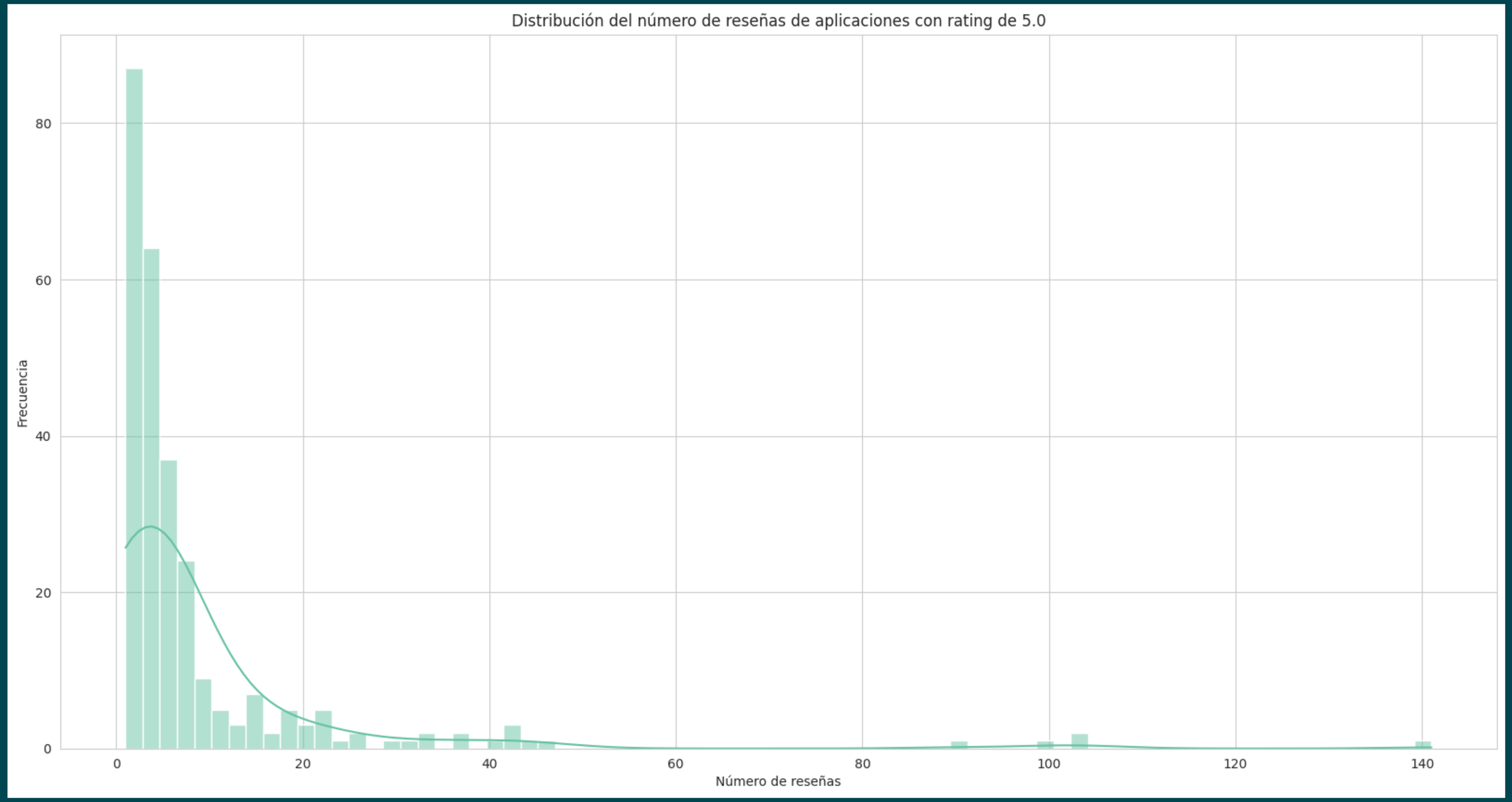

Grafica de densidad de la variable 'Price'
Paid
Densidad
Price
0.000
0.005
0.010
0.015
0.020
0.025
0
100
200
300
400


Grafica de densidad de la variable 'Updated_Year'
Free
Paid
Densidad
Updated_Year
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
2010
2012
2014
2016
2018

# Análisis de Apps con rating perfecto (5)




Si bien las aplicaciones han recibido calificaciones completas, la mayoría de ellas tienen un número de instalaciones relativamente bajo. Esto sugiere que, a pesar de sus buenas valoraciones, estas aplicaciones pueden no ser ampliamente adoptadas o consideradas como los mejores productos




La distribución está sesgada hacia la derecha, lo que indica que hay muchas aplicaciones con pocas reseñas que han recibido calificaciones de 5.0, lo cual puede resultar engañoso.

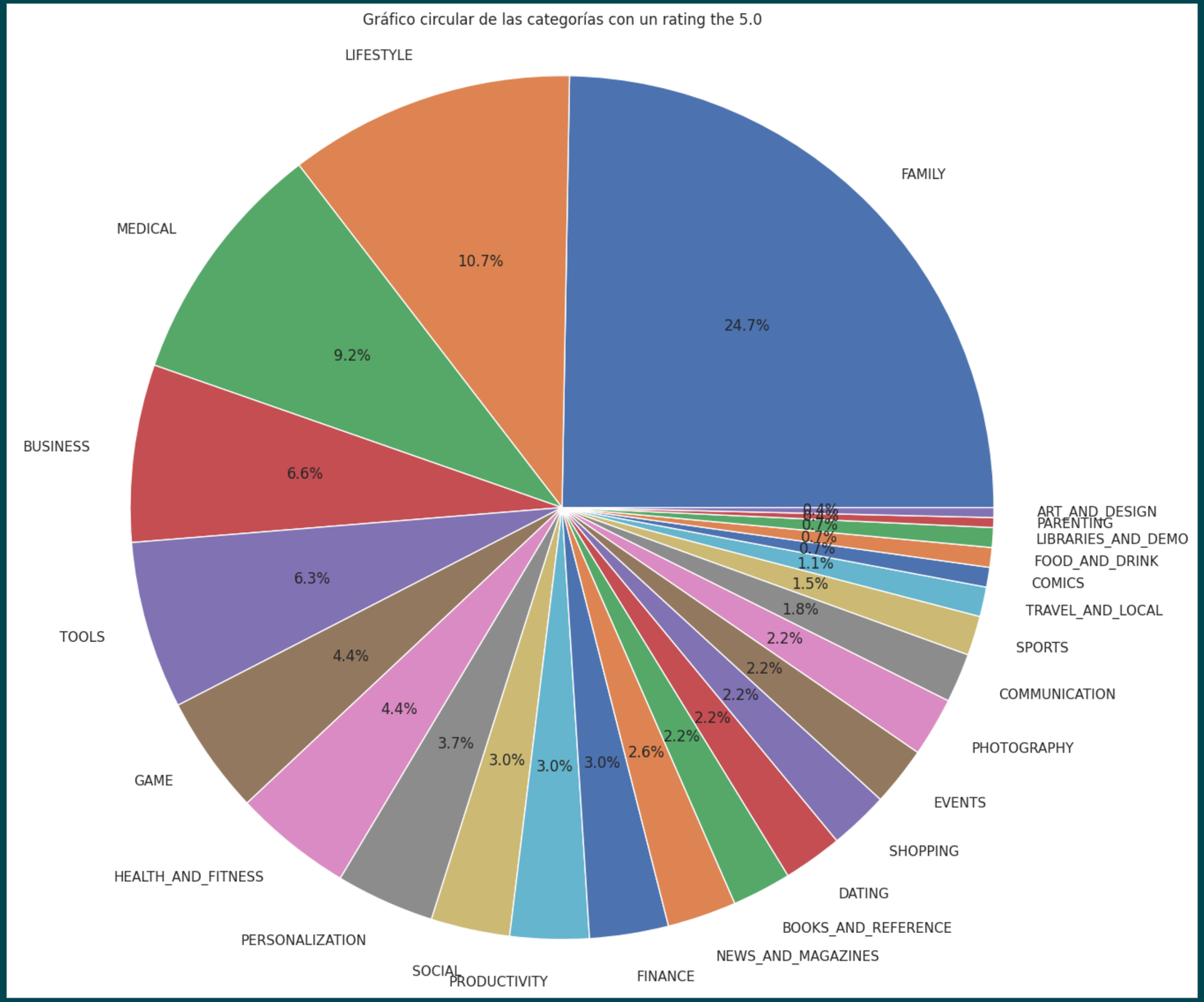


Gráfico circular de las categorías con un rating the 5.0
LIFESTYLE
FAMILY
MEDICAL
BUSINESS
TOOLS
GAME
HEALTH_AND_FITNESS
PERSONALIZATION
SOCIAL
PRODUCTIVITY
FINANCE
NEWS_AND_MAGAZINES
BOOKS_AND_REFERENCE
DATING
SHOPPING
EVENTS
PHOTOGRAPHY
COMMUNICATION
SPORTS
TRAVEL_AND_LOCAL
COMICS
FOOD_AND_DRINK
LIBRARIES_AND_DEMO
PARENTING
ART_AND_DESIGN
10.7%
24.7%
9.2%
6.6%
6.3%
4.4%
4.4%
3.7%
3.0%
3.0%
3.0%
2.6%
2.2%
2.2%
2.2%
2.2%
2.2%
1.8%
1.5%
1.1%
0.7%
0.7%
0.7%
0.4%
0.4%

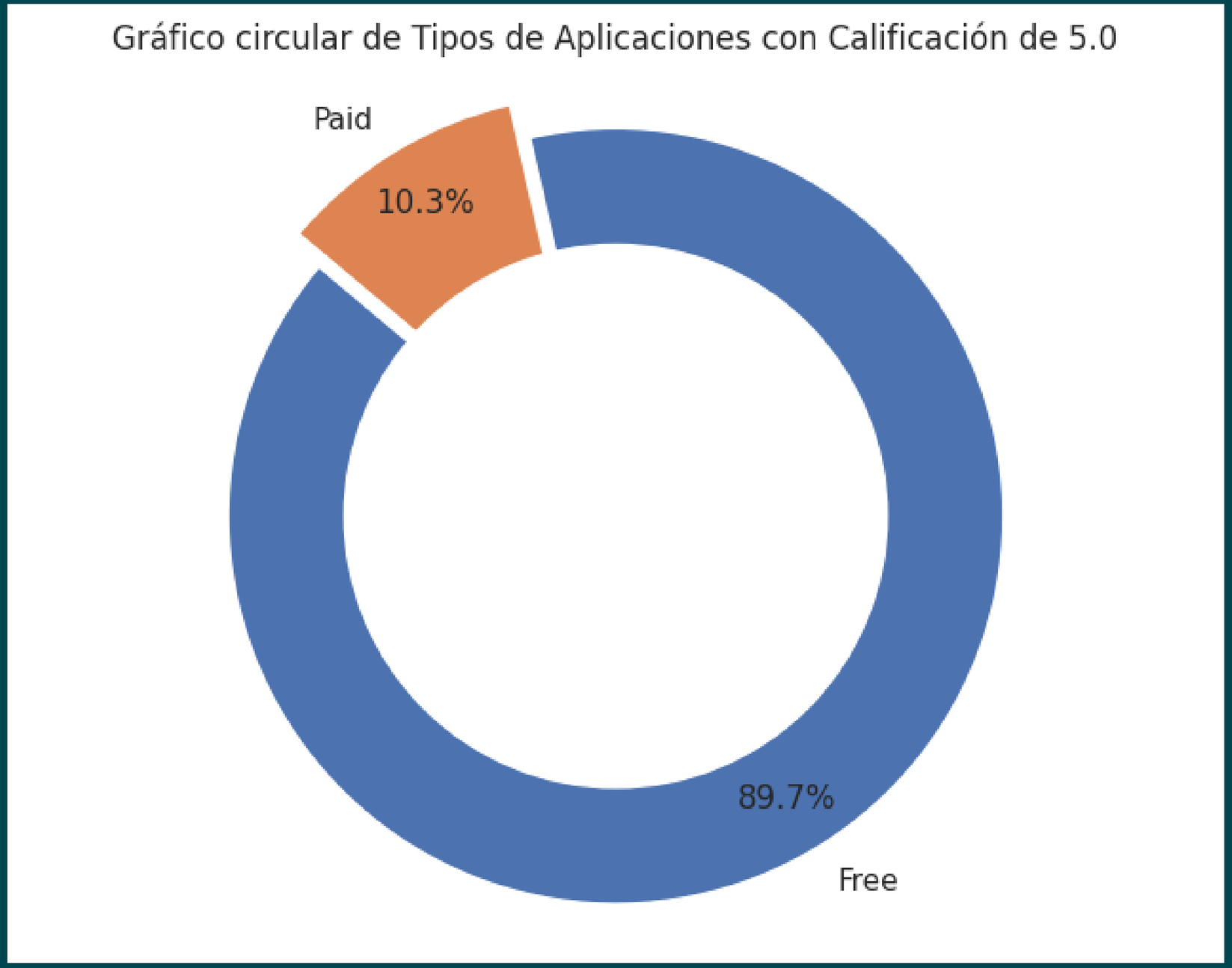


Gráfico circular de Tipos de Aplicaciones con Calificación de 5.0
Paid
10.3%
89.7%
Free

# Preparación para el Modelo

# Primer Paso: Eliminación de registros

**Optamos por eliminar las siguientes características:**

**- Nombre de la aplicación (App): los nombres de las aplicaciones no aportan valor al modelo.**

**- Géneros (Genres): la información contenida es redundante con la categoría de función.**

**- Versión actual: la versión actual de una aplicación carece de relevancia para el modelo.**

**- Versión de Android: la versión de Android de una aplicación no aporta valor significativo al análisis.**

# Segundo Paso: División de los datos

## ¿Por que se dividen los datos?

1. Entrenamiento del modelo: Estos datos de entrenamiento se utilizan para ajustar los parámetros del modelo de manera que pueda aprender patrones y relaciones en los datos.

2. Validación del modelo: Una vez que el modelo ha sido entrenado, es crucial evaluar su rendimiento en datos que no ha visto durante el entrenamiento. Esto ayuda a verificar si el modelo generaliza bien a nuevos datos y no está simplemente memorizando el conjunto de entrenamiento (fenómeno conocido como sobreajuste o sobreajuste).

3. Prueba del modelo: Después de validar el modelo, se utiliza otro conjunto de datos independiente, conocido como conjunto de prueba, para evaluar su rendimiento final de manera imparcial. Esto proporciona una estimación más precisa del rendimiento del modelo en datos del mundo real.

4. Control de sesgo y varianza: Dividir los datos en conjuntos de entrenamiento, validación y prueba ayuda a controlar el sesgo y la varianza del modelo. El conjunto de entrenamiento se utiliza para ajustar los parámetros del modelo, el conjunto de validación se utiliza para ajustar los hiperparámetros del modelo y el conjunto de prueba se utiliza para evaluar el rendimiento general del modelo.

## Variables Predictoras

| | Category | Reviews | Size | Installs | Type | Price | Content Rating | Updated_Month | Updated_Year |
|---|---|---|---|---|---|---|---|---|---|
| 0 | ART_AND_DESIGN | 159 | 19.0 | 10000 | Free | 0.0 | Everyone | 1 | 2018 |
| 1 | ART_AND_DESIGN | 967 | 14.0 | 500000 | Free | 0.0 | Everyone | 1 | 2018 |
| 2 | ART_AND_DESIGN | 87510 | 8.7 | 5000000 | Free | 0.0 | Everyone | 8 | 2018 |
| 3 | ART_AND_DESIGN | 215644 | 25.0 | 50000000 | Free | 0.0 | Teen | 6 | 2018 |
| 4 | ART_AND_DESIGN | 967 | 2.8 | 100000 | Free | 0.0 | Everyone | 6 | 2018 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 10836 | FAMILY | 38 | 53.0 | 5000 | Free | 0.0 | Everyone | 7 | 2017 |
| 10837 | FAMILY | 4 | 3.6 | 100 | Free | 0.0 | Everyone | 7 | 2018 |
| 10838 | MEDICAL | 3 | 9.5 | 1000 | Free | 0.0 | Everyone | 1 | 2017 |
| 10839 | BOOKS_AND_REFERENCE | 114 | 13.0 | 1000 | Free | 0.0 | Mature 17+ | 1 | 2015 |
| 10840 | LIFESTYLE | 398307 | 19.0 | 10000000 | Free | 0.0 | Everyone | 7 | 2018 |

10356 rows × 9 columns

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10356 entries, 0 to 10840
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Category        10356 non-null  object 
 1   Reviews         10356 non-null  int64  
 2   Size            10356 non-null  float64
 3   Installs        10356 non-null  int64  
 4   Type            10356 non-null  object 
 5   Price           10356 non-null  float64
 6   Content Rating  10356 non-null  object 
 7   Updated_Month   10356 non-null  int64  
 8   Updated_Year    10356 non-null  int64  
dtypes: float64(2), int64(4), object(3)
memory usage: 809.1+ KB
```

**Dividimos los datos  en 80% para el entrenamiento (X_Train) y el 20% para prueba (X_Test)**

# Tercer Paso: Codificar y Estandarizar

| | Category | Reviews | Size | Installs | Type | Price | Content Rating | Updated_Month | Updated_Year |
|---|---|---|---|---|---|---|---|---|---|
| 0 | ART_AND_DESIGN | 159 | 19.0 | 10000 | Free | 0.0 | Everyone | 1 | 2018 |
| 1 | ART_AND_DESIGN | 967 | 14.0 | 500000 | Free | 0.0 | Everyone | 1 | 2018 |
| 2 | ART_AND_DESIGN | 87510 | 8.7 | 5000000 | Free | 0.0 | Everyone | 8 | 2018 |
| 3 | ART_AND_DESIGN | 215644 | 25.0 | 50000000 | Free | 0.0 | Teen | 6 | 2018 |
| 4 | ART_AND_DESIGN | 967 | 2.8 | 100000 | Free | 0.0 | Everyone | 6 | 2018 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 10836 | FAMILY | 38 | 53.0 | 5000 | Free | 0.0 | Everyone | 7 | 2017 |
| 10837 | FAMILY | 4 | 3.6 | 100 | Free | 0.0 | Everyone | 7 | 2018 |
| 10838 | MEDICAL | 3 | 9.5 | 1000 | Free | 0.0 | Everyone | 1 | 2017 |
| 10839 | BOOKS_AND_REFERENCE | 114 | 13.0 | 1000 | Free | 0.0 | Mature 17+ | 1 | 2015 |
| 10840 | LIFESTYLE | 398307 | 19.0 | 10000000 | Free | 0.0 | Everyone | 7 | 2018 |

10356 rows × 9 columns

### Codificación de variables categóricas:

Las variables categóricas se representan como cadenas de texto o categorías. Los algoritmos de aprendizaje automático requieren que las variables sean numéricas, por lo que es necesario codificar estas variables categóricas en valores numéricos. Esto se hace para que el modelo pueda entender y trabajar con estas características.

### Estandarización de variables numéricas:

Las variables numéricas pueden tener diferentes escalas y rangos. Algunos algoritmos de aprendizaje automático, son sensibles a la escala de las características. La estandarización, que implica restar la media y dividir por la desviación estándar, ayuda a poner todas las variables en la misma escala, lo que puede mejorar el rendimiento del modelo y facilitar la interpretación de los coeficientes.

# Label Enconding para variables categóricas

El label encoding es una técnica de preprocesamiento de datos que se utiliza para convertir variables categóricas en valores numéricos. Consiste en asignar un valor numérico único a cada categoría de la variable categórica. Por ejemplo, si una variable categórica tiene tres categorías: "rojo", "verde" y "azul", el label encoding podría asignar los valores 0, 1 y 2 respectivamente a cada una de estas categorías.

1. Instanciamos
2. Entrenamos y tranformamos
3. Convertimos las variables en tipo “Catégory”

## Original

| | Category | Reviews | Size | Installs | Type | Price | Content Rating | Updated_Month | Updated_Year |
|---|---|---|---|---|---|---|---|---|---|
| 0 | ART_AND_DESIGN | 159 | 19.0 | 10000 | Free | 0.0 | Everyone | 1 | 2018 |
| 1 | ART_AND_DESIGN | 967 | 14.0 | 500000 | Free | 0.0 | Everyone | 1 | 2018 |
| 2 | ART_AND_DESIGN | 87510 | 8.7 | 5000000 | Free | 0.0 | Everyone | 8 | 2018 |
| 3 | ART_AND_DESIGN | 215644 | 25.0 | 50000000 | Free | 0.0 | Teen | 6 | 2018 |
| 4 | ART_AND_DESIGN | 967 | 2.8 | 100000 | Free | 0.0 | Everyone | 6 | 2018 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 10836 | FAMILY | 38 | 53.0 | 5000 | Free | 0.0 | Everyone | 7 | 2017 |
| 10837 | FAMILY | 4 | 3.6 | 100 | Free | 0.0 | Everyone | 7 | 2018 |
| 10838 | MEDICAL | 3 | 9.5 | 1000 | Free | 0.0 | Everyone | 1 | 2017 |
| 10839 | BOOKS_AND_REFERENCE | 114 | 13.0 | 1000 | Free | 0.0 | Mature 17+ | 1 | 2015 |
| 10840 | LIFESTYLE | 398307 | 19.0 | 10000000 | Free | 0.0 | Everyone | 7 | 2018 |

10356 rows × 9 columns

```
features_to_encode = X_train.select_dtypes(include=['category', 'object']).columns #Columnas Category, Type y content_rating

for col in features_to_encode:
    le = LabelEncoder()

    X_train[col] = le.fit_transform(X_train[col]) # Entrenamos y transformamos el data set de Entrenamiento
    X_train[col] = X_train[col].astype('category') # Convertimos la data transformada de variable numerica a "categoría"

    X_test[col] = le.transform(X_test[col]) # Solo transformamos los datos de testing
    X_test[col] = X_test[col].astype('category') #Convertimos la data transformada de variable numerica a "categoría"
```

Al convertir una variable categórica a tipo "Category", Python asigna un código entero a cada categoría única y guarda estos códigos en memoria en lugar de las cadenas originales. Esto puede reducir significativamente el uso de memoria, especialmente si la variable categórica tiene un número limitado de categorías pero aparece muchas veces en el conjunto de datos.

# Standard Scaler para variables continuas y discretas

El StandardScaler es útil cuando las características tienen escalas diferentes o varianzas muy diferentes entre sí. Al estandarizar las características, se centran alrededor de cero y tienen la misma escala, lo que puede mejorar el rendimiento de ciertos algoritmos de machine learning, como aquellos que son sensibles a la escala de las características

1. Instanciamos
2. Entrenamos y tranformamos

```
numeric_features
```

```
Index(['Reviews', 'Size', 'Installs', 'Price', 'Updated_Year'], dtype='object')
```

```
scaler = StandardScaler()

# Entrenamos y transformamos los datos de entrenamiento
X_train[numeric_features] = scaler.fit_transform(X_train[numeric_features])

# Solo transformamos los datos de prueba
X_test[numeric_features] = scaler.transform(X_test[numeric_features])
```

## Original

| | Category | Reviews | Size | Installs | Type | Price | Content Rating | Updated_Month | Updated_Year |
|---|---|---|---|---|---|---|---|---|---|
| 0 | ART_AND_DESIGN | 159 | 19.0 | 10000 | Free | 0.0 | Everyone | 1 | 2018 |
| 1 | ART_AND_DESIGN | 967 | 14.0 | 500000 | Free | 0.0 | Everyone | 1 | 2018 |
| 2 | ART_AND_DESIGN | 87510 | 8.7 | 5000000 | Free | 0.0 | Everyone | 8 | 2018 |
| 3 | ART_AND_DESIGN | 215644 | 25.0 | 50000000 | Free | 0.0 | Teen | 6 | 2018 |
| 4 | ART_AND_DESIGN | 967 | 2.8 | 100000 | Free | 0.0 | Everyone | 6 | 2018 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 10836 | FAMILY | 38 | 53.0 | 5000 | Free | 0.0 | Everyone | 7 | 2017 |
| 10837 | FAMILY | 4 | 3.6 | 100 | Free | 0.0 | Everyone | 7 | 2018 |
| 10838 | MEDICAL | 3 | 9.5 | 1000 | Free | 0.0 | Everyone | 1 | 2017 |
| 10839 | BOOKS_AND_REFERENCE | 114 | 13.0 | 1000 | Free | 0.0 | Mature 17+ | 1 | 2015 |
| 10840 | LIFESTYLE | 398307 | 19.0 | 10000000 | Free | 0.0 | Everyone | 7 | 2018 |

10356 rows × 9 columns

# Data Set Final para proceso de Modelado

| | Category | Reviews | Size | Installs | Type | Price | Content Rating | Updated_Month | Updated_Year |
|---|---|---|---|---|---|---|---|---|---|
| **1948** | 14 | -0.143720 | 1.428843 | -0.160552 | 0 | -0.061293 | 3 | 8 | 0.559437 |
| **1201** | 13 | -0.090481 | -0.334315 | -0.111286 | 0 | -0.061293 | 1 | 7 | 0.559437 |
| **4700** | 28 | -0.146763 | -0.334315 | -0.172253 | 1 | 0.226698 | 1 | 1 | -0.338524 |
| **9156** | 11 | -0.071331 | 3.620878 | -0.111286 | 0 | -0.061293 | 1 | 6 | 0.559437 |
| **3632** | 32 | -0.079929 | -0.334315 | -0.049704 | 0 | -0.061293 | 1 | 8 | 0.559437 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **6193** | 24 | -0.145016 | -0.515397 | -0.160552 | 0 | -0.061293 | 1 | 6 | 0.559437 |
| **5648** | 14 | -0.137160 | 1.428843 | -0.171637 | 1 | 0.154520 | 4 | 4 | -2.134446 |
| **5847** | 18 | -0.147529 | -0.334315 | -0.172856 | 1 | 0.298876 | 3 | 7 | 0.559437 |
| **1008** | 10 | -0.137829 | -0.334315 | -0.111286 | 0 | -0.061293 | 1 | 8 | 0.559437 |
| **7740** | 14 | -0.146482 | 0.285173 | -0.166710 | 0 | -0.061293 | 4 | 7 | -0.338524 |

8284 rows × 9 columns

# Aplicación de Modelos

# Identificación del modelo “adecuado”

**Se desarrollaron ocho modelos supervisados, divididos en cuatro para regresión y cuatro para clasificación.**

**Entre los modelos de regresión utilizados se incluyen:**

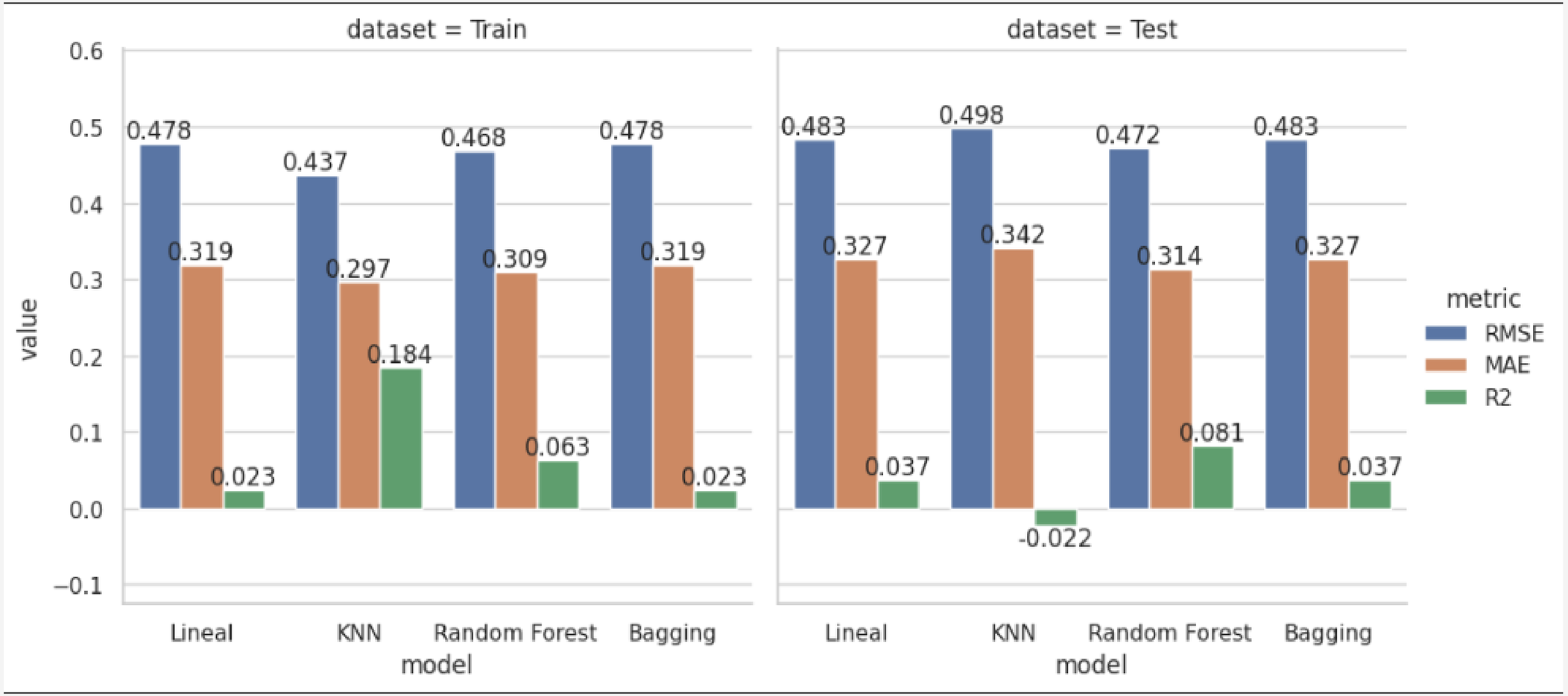
1. **Regresión lineal**
2. **Vecinos más cercanos para regresión (KNN)**
3. **Bosques aleatorios para regresión y**
4. **Bagging con regresión lineal.**

**Asimismo, entre los modelos de clasificación empleados se encuentran:**

1. **Regresión logística**
2. **Vecinos más cercanos para clasificación (KNN)**
3. **Bosques aleatorios para clasificación**
4. **Bagging con KNN para clasificación.**

**A continuación, se presentan los resultados obtenidos de los modelos aplicados.**

# Resultados Modelos de Regresión

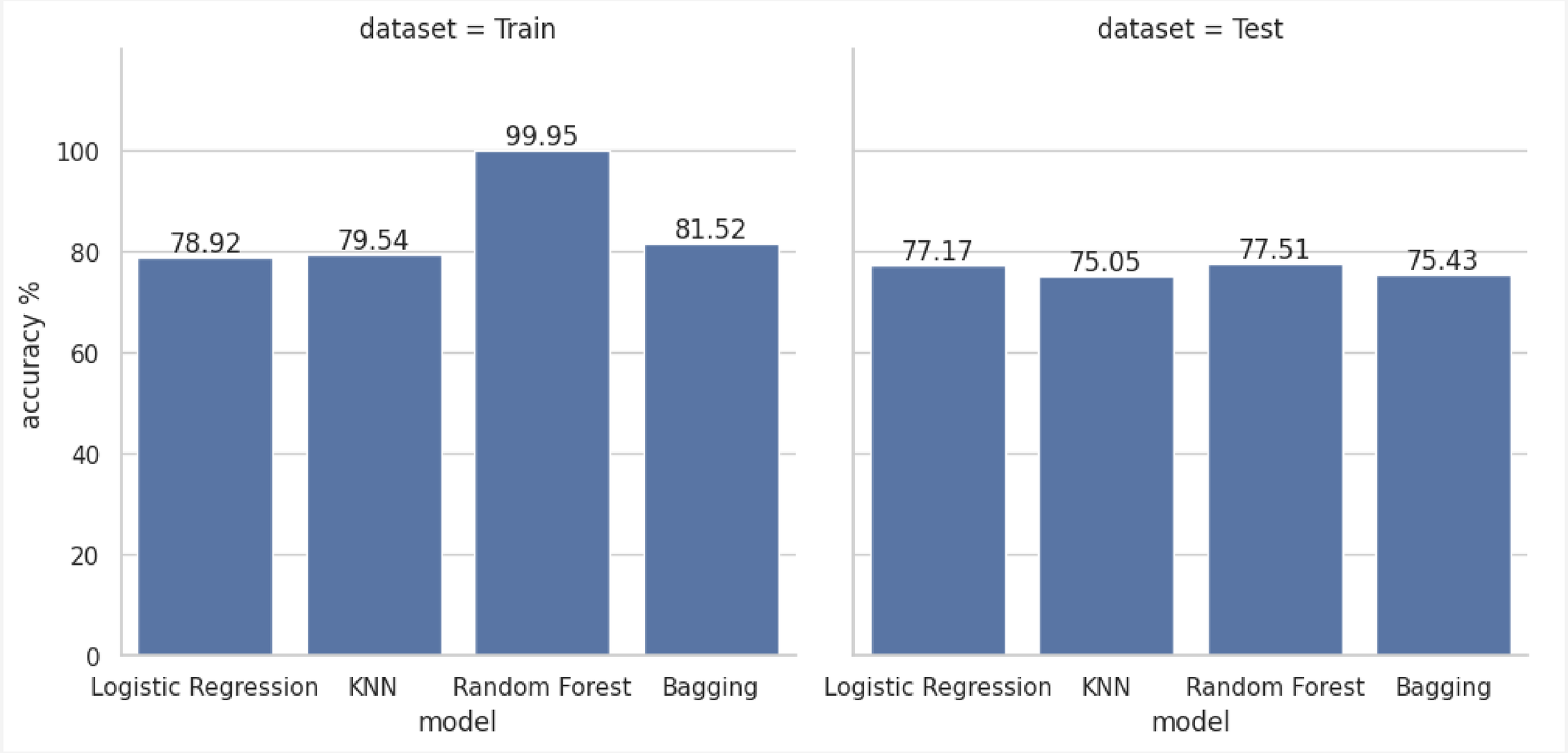



**R2 (Coeficiente de determinación)** : indica qué tan bien las variaciones en las características explicativas del modelo se corresponden con las variaciones en la variable objetivo. Un valor de R2 cercano a 1 significa que el modelo se ajusta muy bien a los datos, mientras que un valor cercano a 0 significa que el modelo no explica la variabilidad de la variable objetivo.

**MAE (Error Absoluto)** : se calcula como la media de las diferencias absolutas entre las predicciones del modelo y los valores reales. En otras palabras, el MAE nos dice cuánto, en promedio, las predicciones del modelo se desvían de los valores reales. Un MAE más bajo indica que el modelo tiene un mejor rendimiento en términos de precisión de predicción.

**RMSE (Error Cuadrático Medio)** : El RMSE es similar al MAE, pero da más peso a los errores grandes porque los errores se elevan al cuadrado antes de tomar la media y luego se calcula la raíz cuadrada del resultado. Esto significa que el RMSE penaliza más los errores grandes que el MAE. El RMSE también se interpreta en las mismas unidades que la variable objetivo, lo que lo hace más interpretable. Un RMSE más bajo indica un mejor ajuste del modelo a los datos.

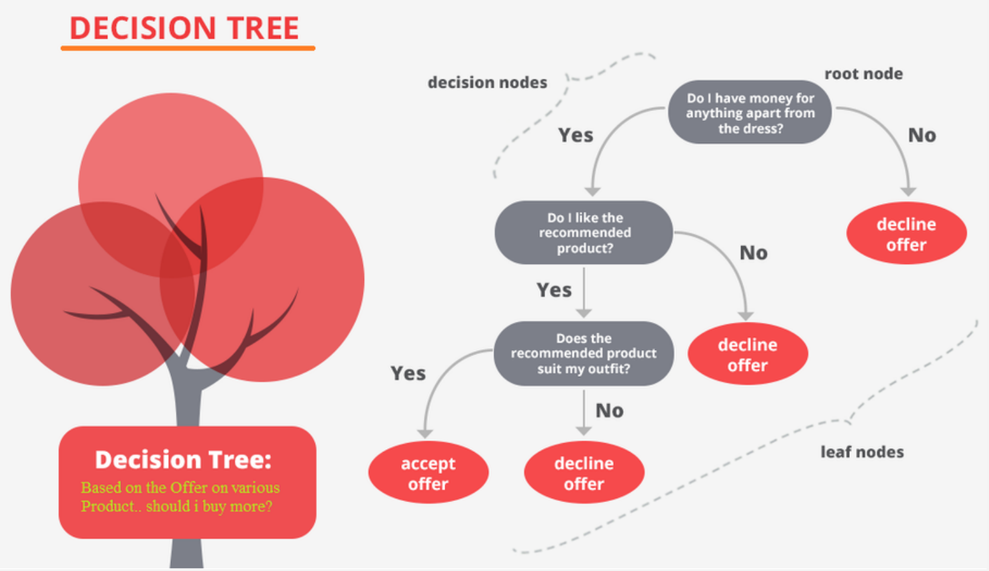
# Resultados Modelos de Clasificación




**Se consideró exclusivamente la precisión del modelo, la cual representa una comparación entre los datos reales y las predicciones generadas por el modelo.**

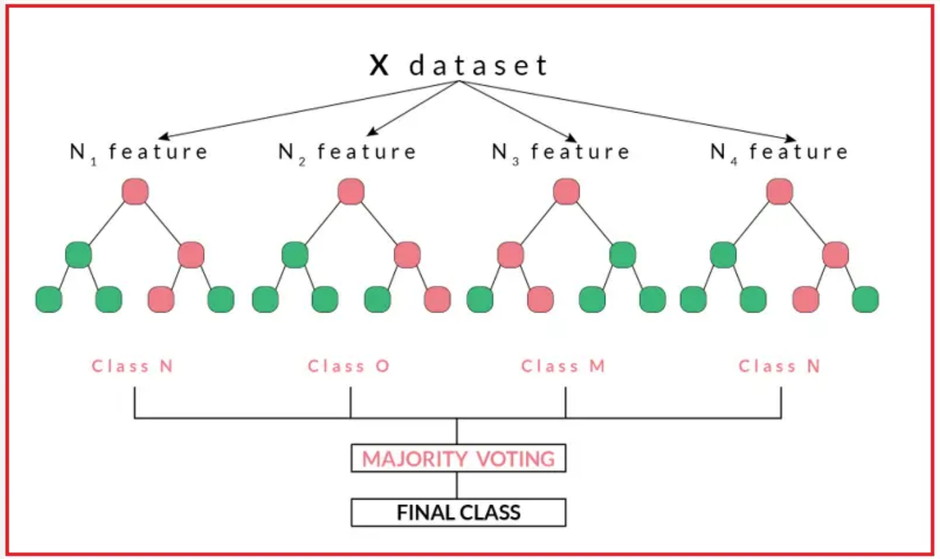
# Demostración del Modelo con Bosques Aleatorios para Clasificación

# ¿Que son los Bosques Aleatorios?




**Imagina que estás tratando de tomar una decisión importante, pero en lugar de confiar en la opinión de una sola persona, decides consultar a un grupo de personas expertas en diferentes áreas. Cada persona te da su opinión, y luego tú tomas una decisión basada en lo que la mayoría de ellos piensa.**

**Los bosques aleatorios funcionan de manera similar. En lugar de depender de un solo "experto" (es decir, un solo árbol de decisión), se crean muchos árboles de decisión diferentes utilizando diferentes subconjuntos de datos y características aleatorias. Cada árbol de decisión en el "bosque" tiene su propia opinión sobre cómo clasificar o predecir los datos.**



# Pipelines e Hiperparámetros

```
# Pipeline Random Forest
pipeline_rf_clf = Pipeline([
    ('rf', RandomForestClassifier())
])

params_rf_clf = [{'rf__n_estimators': [50, 70, 100],
                  'rf__criterion': ['gini', 'entropy']}]
```

**- n_estimators: Este parámetro indica el número de árboles de decisión que se utilizarán en el bosque.**

**- criterion: Este parámetro especifica la función para medir la calidad de una división en un nodo del árbol. Las dos opciones principales son:**

**'gini': Utiliza el índice Gini para medir la pureza de los nodos. El índice Gini es una medida de impureza que se utiliza para clasificación. Nos mide la probabilidad de no sacar dos registros de la misma clase del nodo.**

**'entropy': Utiliza la ganancia de información (entropía) para medir la calidad de la división. La entropía es otra medida de impureza que también se utiliza para clasificación.**

# Mejores Parámetros para el modelo

| | mean fit time | std fit time | mean score time | std score time | param rf__criterion | param rf__n_estimators | params | split0 test score | split1 test score | split2 test score | split3 test score | split4 test score | mean test score | std test score | rank test score |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.162313 | 0.004391 | 0.009407 | 0.000343 | gini | 50 | {'rf__criterion': 'gini', 'rf__n_estimators': 50} | 0.742169 | 0.727711 | 0.741546 | 0.758454 | 0.768116 | 0.747599 | 0.014145 | 1 |
| 5 | 0.366385 | 0.008192 | 0.016822 | 0.002824 | entropy | 100 | {'rf__criterion': 'entropy', 'rf__n_estimators... | 0.751807 | 0.725301 | 0.748792 | 0.753623 | 0.751208 | 0.746146 | 0.010536 | 2 |
| 4 | 0.262981 | 0.009707 | 0.011782 | 0.000821 | entropy | 70 | {'rf__criterion': 'entropy', 'rf__n_estimators... | 0.739759 | 0.727711 | 0.743961 | 0.753623 | 0.760870 | 0.745185 | 0.011432 | 3 |
| 1 | 0.229087 | 0.004683 | 0.012133 | 0.000808 | gini | 70 | {'rf__criterion': 'gini', 'rf__n_estimators': 70} | 0.739759 | 0.720482 | 0.743961 | 0.746377 | 0.768116 | 0.743739 | 0.015218 | 4 |
| 2 | 0.319291 | 0.008052 | 0.014877 | 0.000182 | gini | 100 | {'rf__criterion': 'gini', 'rf__n_estimators': ... | 0.759036 | 0.715663 | 0.743961 | 0.746377 | 0.753623 | 0.743732 | 0.015013 | 5 |
| 3 | 0.187319 | 0.008399 | 0.008845 | 0.000236 | entropy | 50 | {'rf__criterion': 'entropy', 'rf__n_estimators... | 0.759036 | 0.703614 | 0.746377 | 0.743961 | 0.748792 | 0.740356 | 0.019075 | 6 |

**En este caso, n_estimators=50 significa que se construirán 50 árboles en el bosque.**

**En este caso, criterion='gini' indica que se utilizará el índice Gini para medir la calidad de las divisiones en los árboles de decisión del bosque.**

# Ajuste del Modelo y presición

```
[150] rf_clf = RandomForestClassifier(n_estimators=50, criterion='gini')
      rf_clf.fit(X_train, y_train_int)
```

```
RandomForestClassifier(n_estimators=50)
```

```
[151] df_metrics_clf.loc['Random Forest', 'Train'] = rf_clf.score(X_train, y_train_int)
      df_metrics_clf.loc['Random Forest', 'Test'] = rf_clf.score(X_test, y_test_int)
```

## Nota Adicional

**En situaciones donde el número de clases es demasiado grande, el uso directo de la matriz de confusión y el classification report puede volverse problemático debido a la dificultad para visualizar y comprender la información, así como a la posibilidad de una sobrecarga de datos.**

# Conclusión