Práctica 2 - Tipología y Ciclo de Vida de los Datos

Esteban Braganza Cajas y Ana Álvarez Sánchez

2024-05-22

```
# Importamos librerias de analisis de datos
library(tidyverse) # Conjunto de paquetes para manejo de datos
library(magrittr) # Pipe
library(dplyr)
library(recipes)
library(tidymodels) # Machine Learning en R
library(skimr) # Descriptivas univariadas masivas
library(ggplot2)
library(rpart) # Arboles
library(rpart.plot) # Graficar arboles
```

1. Descripción del dataset

¿Por qué es importante y qué pregunta/problema pretende responder? Resume brevemente las variables que lo forman y su tamaño.

Factores que influyen en el número de favoritos de las imágenes en DeviantArt

Nuestro objetivo es identificar y predecir los factores que determinan el número de favoritos que recibe cada imagen en DeviantArt. Buscamos entender las variables que explican esta interacción y cómo se relacionan con la popularidad de las imágenes.

Específicamente, queremos responder las siguientes preguntas:

1. ¿Qué características de las imágenes (como tamaño, resolución y número de comentarios) están más correlacionadas con el número de favoritos?
2. ¿Cómo influyen las variables temporales (como el día de la semana y el momento del día en que se publica una imagen) en la cantidad de favoritos?
3. ¿Qué temas de búsqueda son más propensos a recibir un alto número de favoritos?
4. ¿Existen patrones específicos en la comunidad de DeviantArt que puedan predecir la popularidad de una imagen?

Al entender estos factores, no solo podremos predecir mejor el número de favoritos que puede recibir una imagen, sino también ofrecer recomendaciones a los artistas sobre cómo optimizar sus publicaciones para aumentar su visibilidad y popularidad en la plataforma.

```
# Carga del conjunto de datos
data <- read.csv("../data/images_db.csv")
```

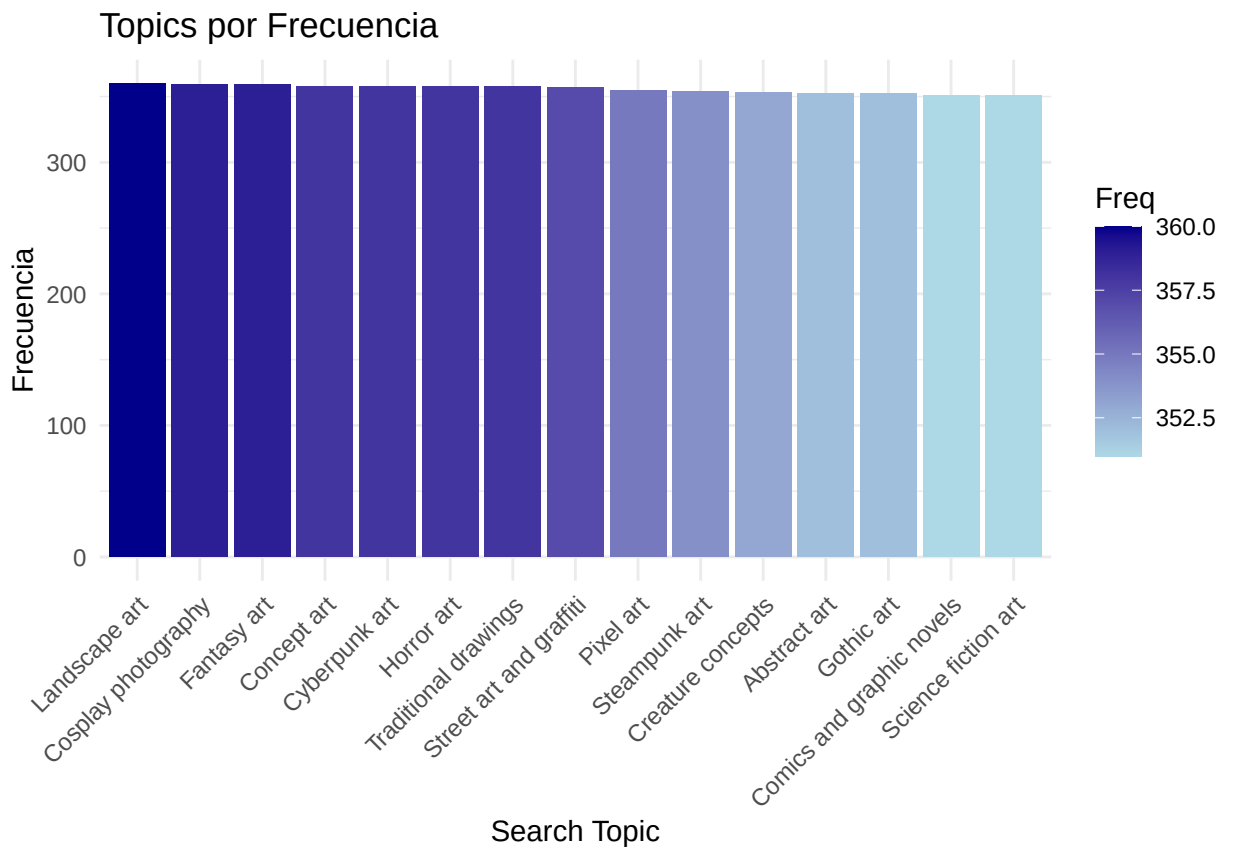
El fichero de datos contiene 7067 registros y 18 variables:

- `search_topic`: la imagen es resultado de la búsqueda por este tema
- `page_num`: la imagen aparece en este número de página de la búsqueda
- `image_page`: enlace a la página con la información de la imagen
- `image_url`: enlace a la imagen
- `image_title`: título de la imagen
- `image_author`: autor/a de la imagen
- `image_favs`: número de veces que le han dado a "me gusta" en la imagen
- `image_com`: número de comentarios que tiene la imagen
- `image_views`: número de vistas a la imagen
- `private_collections`: número de veces que ha sido incluida en una colección privada
- `tags`: etiquetas que se le han asignado a la imagen para facilitar su descubrimiento
- `location`: país o localización geográfica, si el autor la quiere identificar description: campo de texto abierto creado por el autor, que acompaña a la imagen. Puede incluir detalles técnicos o enlaces a las redes sociales del autor/a.
- `description`: descripción de la imagen que hace el autor.
- `image_px`: dimensiones de la imagen, en pixeles
- `image_size`: peso de la imagen en MB.
- `published_date`: fecha de publicación de la imagen.
- `last_comment`: último comentario añadido a la imagen.
- `license`: licencia de la imagen

En cuanto a la descripción de cada una de estas variables:

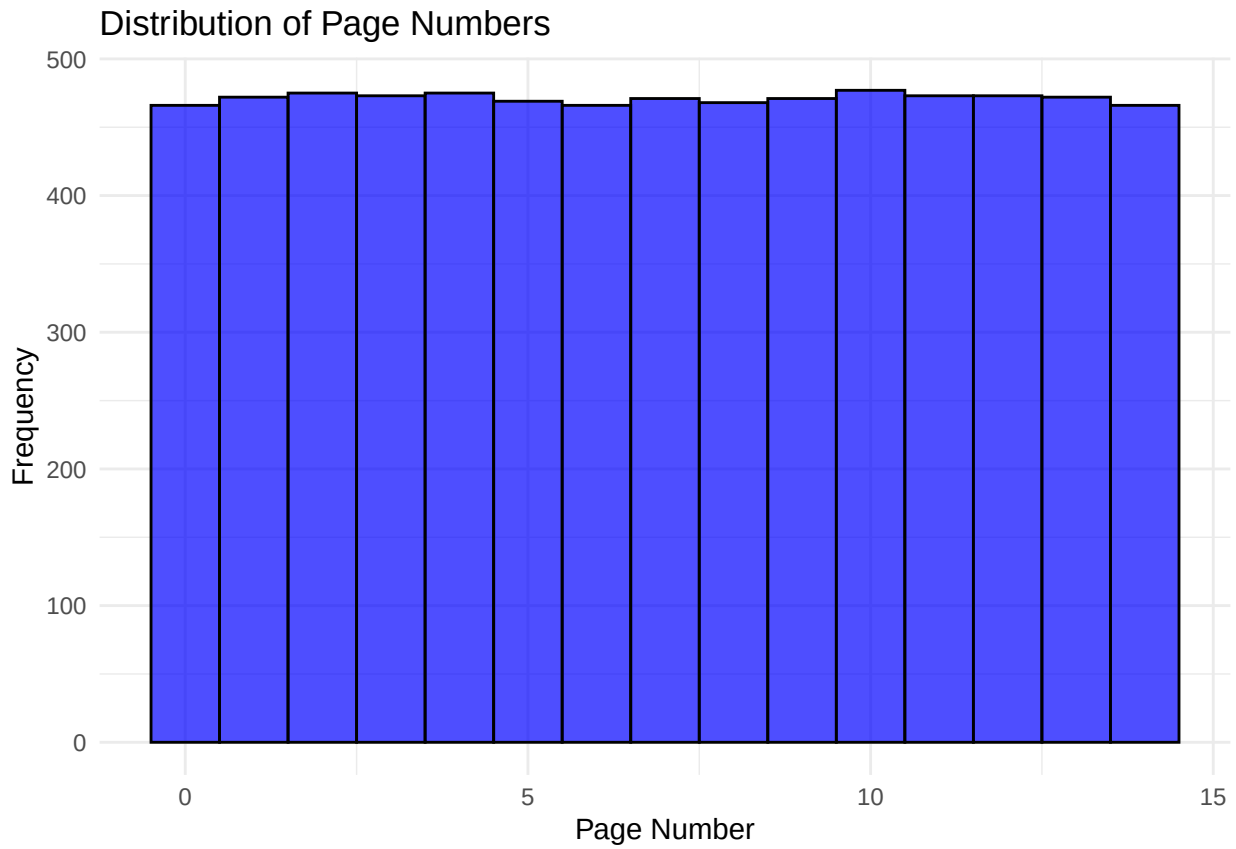
- Las variables `description`, `image_title` y `last_comment` son campos de texto abiertos.
- Las variables `image_url`, `image_page` incluyen los enlaces a la página de la imagen y a la imagen en sí.
- La variable `search_topic` incluye 20 posibilidades, que son: Fantasy art, Science fiction art, Anime and manga art, Fan art (for specific fandoms), Digital paintings, Traditional drawings, Character designs, Creature concepts, Landscape art, Abstract art, Surrealism, Steampunk art, Cyberpunk art, Gothic art, Horror art, Cosplay photography, Pixel art, Concept art, Comics and graphic novels, Street art and graffiti. Las frecuencias de cada una de estas categorías en el conjunto de datos es esta:

```
# Frecuencias de la variable search_topic
frecuencias <- table(data$search_topic)
df_frecuencias <- as.data.frame(frecuencias)
df_frecuencias_ordenado <- df_frecuencias[order(-df_frecuencias$Freq), ]

ggplot(head(df_frecuencias_ordenado, 15), aes(x = reorder(Var1, -Freq), y = Freq, fill = Freq)) +
  geom_bar(stat = "identity") +
  scale_fill_gradient(low = "lightblue", high = "darkblue") +
  labs(title = "Topics por Frecuencia",
       x = "Search Topic",
       y = "Frecuencia") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



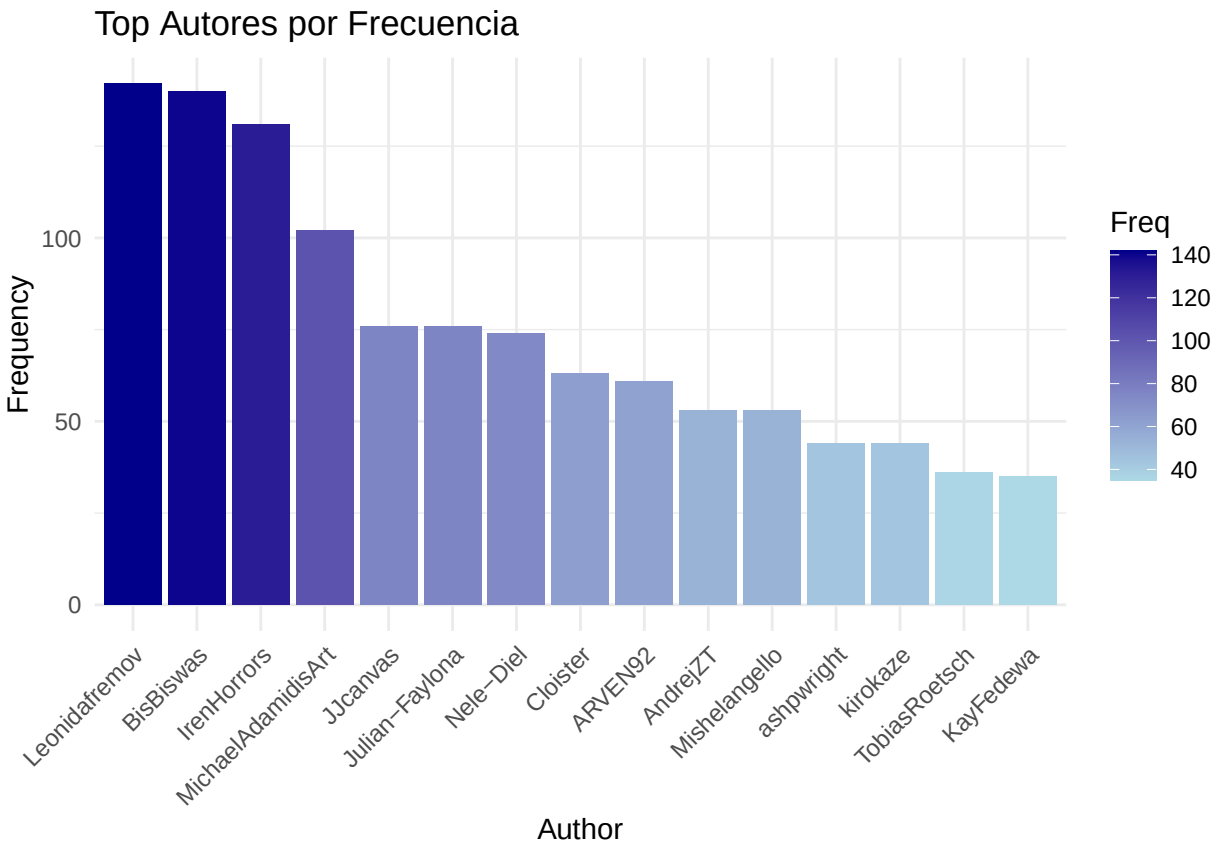
- La variable `page_num` incluye 15 posibilidades, que están en el rango 0, 14, porque en la descarga se han elegido las primeras 15 páginas de cada topic buscado. Podemos ver que tenemos un número similar de imágenes en cada número de página.

```
# Histograma de la variable page_num
ggplot(data, aes(x = page_num)) +
  geom_histogram(binwidth = 1, fill = "blue", color = "black", alpha = 0.7) +
  labs(title = "Distribution of Page Numbers",
       x = "Page Number",
       y = "Frequency") +
  theme_minimal()
```



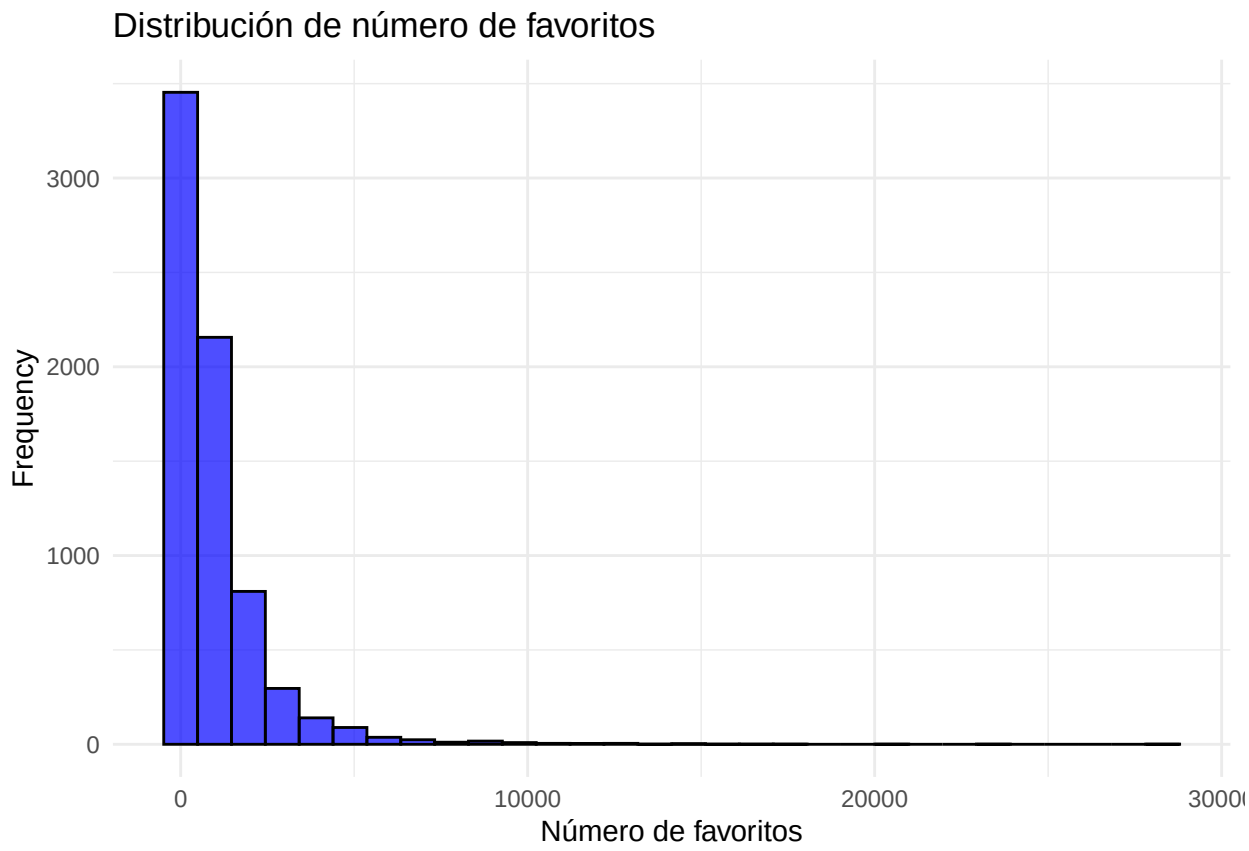
- La variable `image_author` incluye 2903 posibilidades. El autor/a con más registros tiene 142 obras. 2047 tienen una sola obra en el dataset. La media de imágenes por autor es de 2.43.

```
# Autores con mayor número de imágenes en el dataset
frecuencias <- table(data$image_author)
df_frecuencias <- as.data.frame(frecuencias)
df_frecuencias_ordenado <- df_frecuencias[order(-df_frecuencias$Freq), ]

ggplot(head(df_frecuencias_ordenado, 15), aes(x = reorder(Var1, -Freq), y = Freq, fill = Freq)) +
  geom_bar(stat = "identity") +
  scale_fill_gradient(low = "lightblue", high = "darkblue") +
  labs(title = "Top Autores por Frecuencia",
       x = "Author",
       y = "Frequency") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



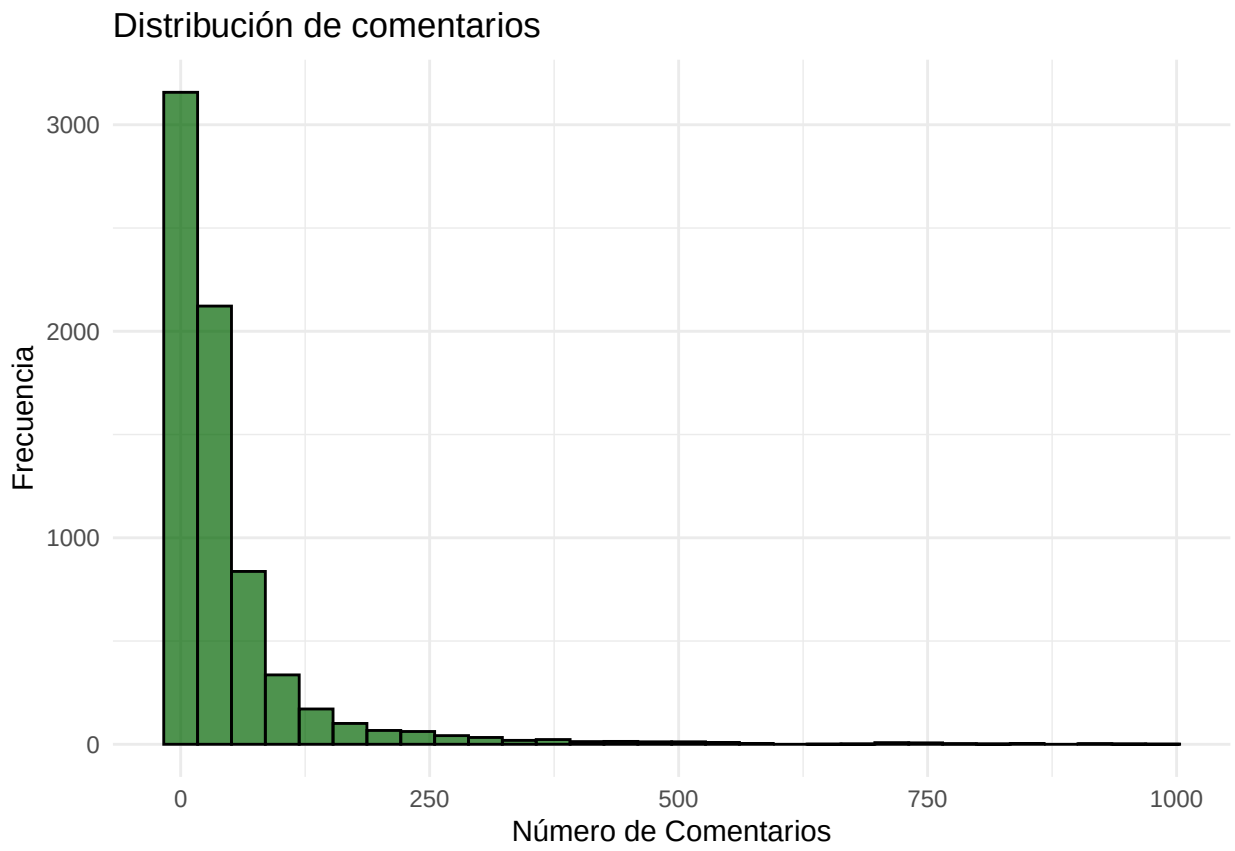
- La variable `image_favs` es numérica, con un valor mínimo de 0, un valor máximo de 28300, una media de 967.7, con esta distribución: Se puede observar que está sesgada a la izquierda con la mayoría de imágenes con un número de favoritos bajo aunque se mira la presencia de outliers.

```
ggplot(data, aes(x = image_favs)) +
  geom_histogram(fill = "blue", color = "black", alpha = 0.7) +
  labs(title = "Distribución de número de favoritos",
       x = "Número de favoritos",
       y = "Frequency") +
  theme_minimal()
```

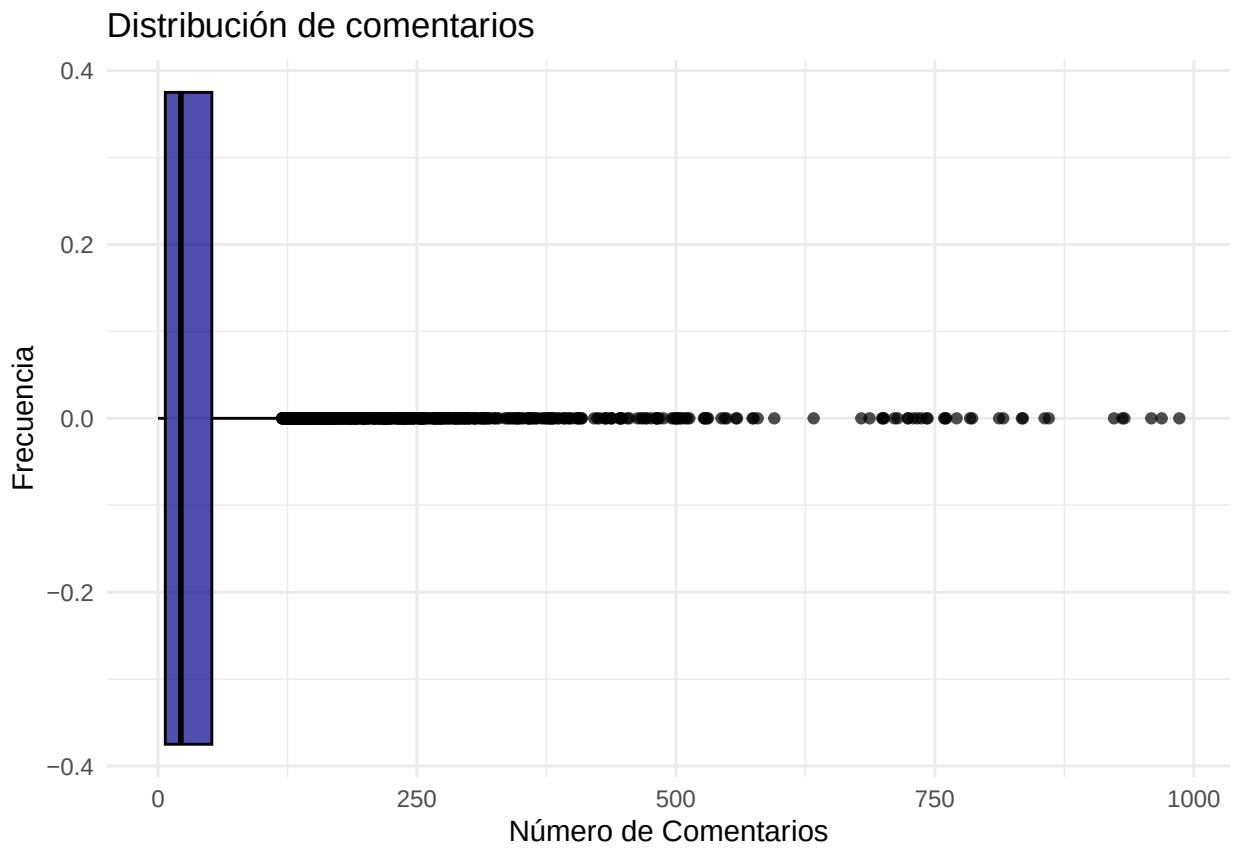


- La variable `image_com` es numérica, con un valor mínimo de 0, un valor máximo de 986, una media de 47.93, con esta distribución que es fuertemente sesgada a la izquierda. Este suele ser un comportamiento común en las interacciones de usuarios online.

```
# Histograma de la variable image_com (número de comentarios)
ggplot(data, aes(x = image_com)) +
  geom_histogram(fill = "darkgreen", color = "black", alpha = 0.7) +
  labs(title = "Distribución de comentarios",
       x = "Número de Comentarios",
       y = "Frecuencia") +
  theme_minimal()
```

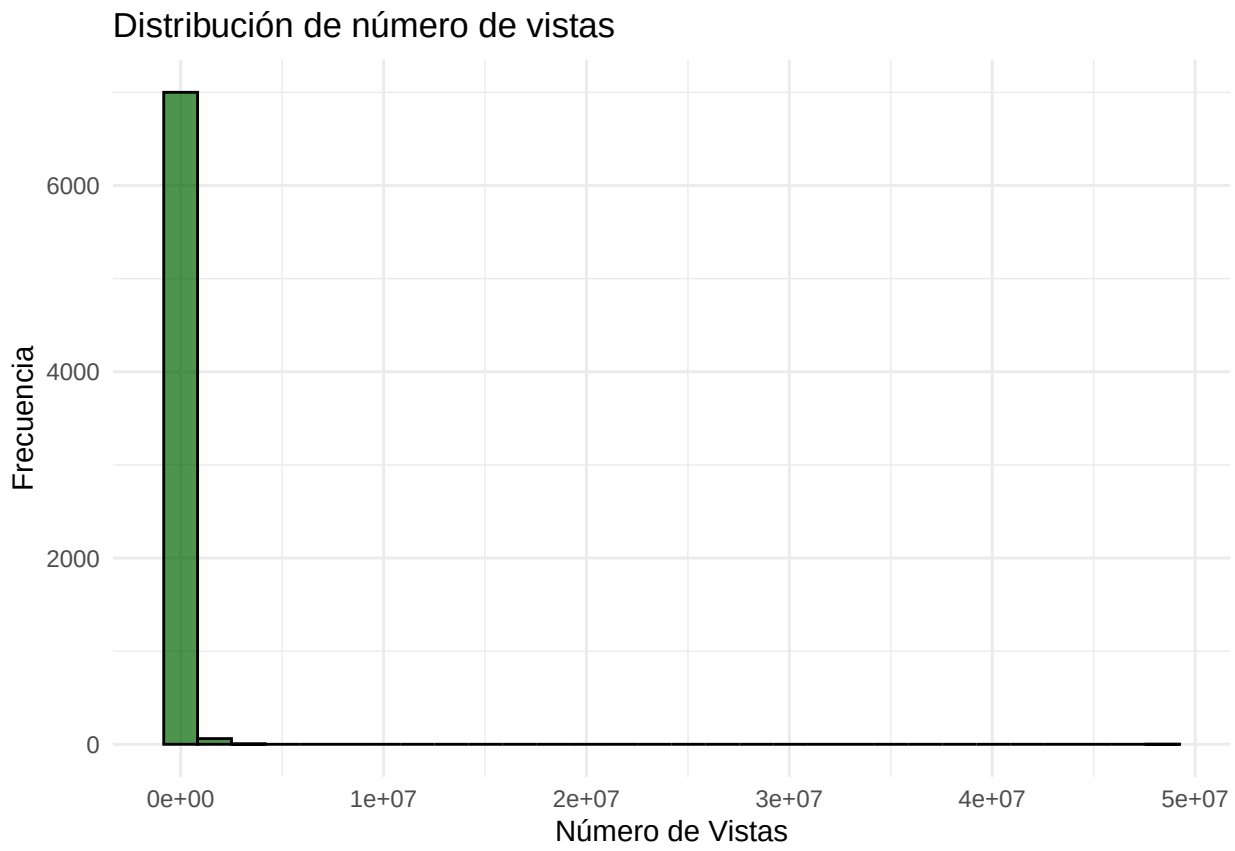



```r
# Boxplot de la variable image_com (número de comentarios)
ggplot(data, aes(x = image_com)) +
  geom_boxplot(fill = "darkblue", color = "black", alpha = 0.7) +
  labs(title = "Distribución de comentarios",
       x = "Número de Comentarios",
       y = "Frecuencia") +
  theme_minimal()
```

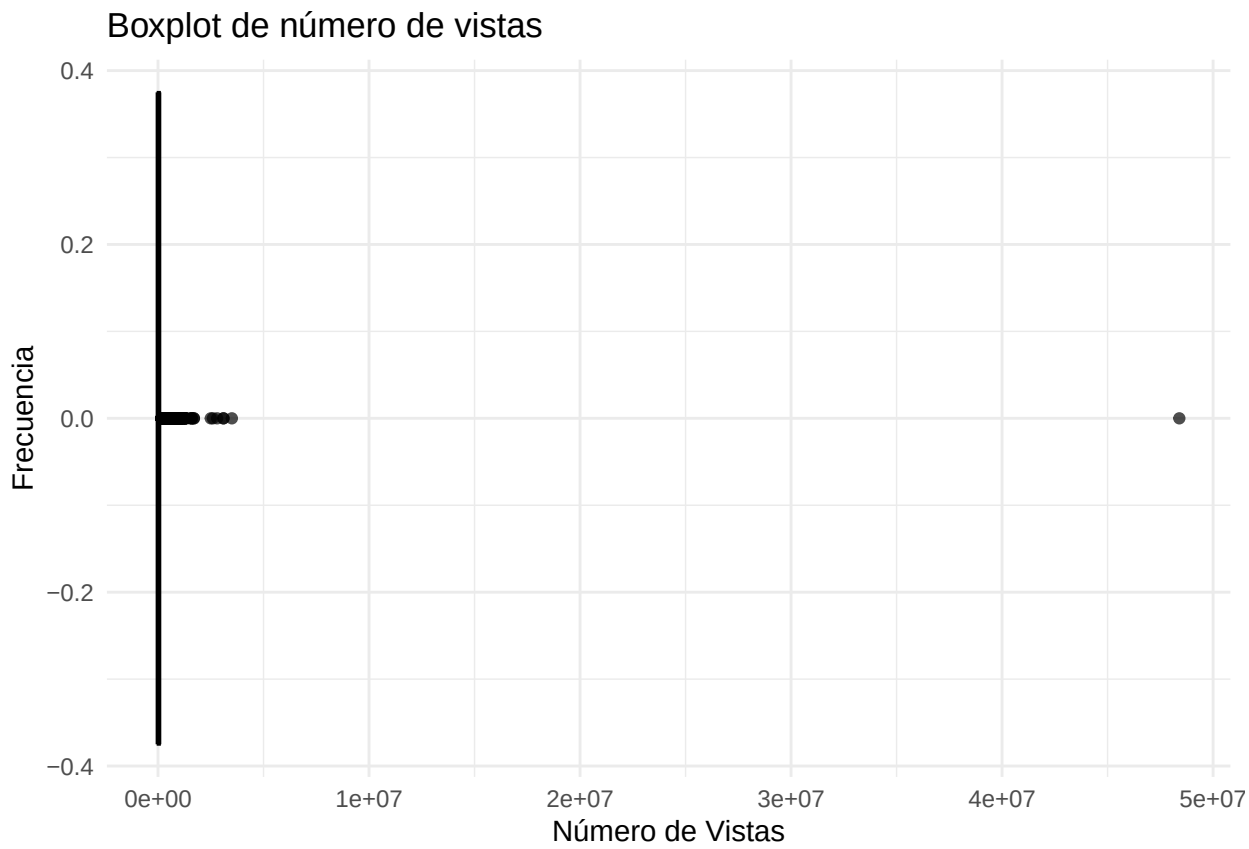


- La variable `image_views` es numérica, con un valor mínimo de 1, un valor máximo de 48400000, una media de 9.926775×10^4, una mediana de 20900, con esta distribución:

```r
# Exploración de la variable image_views
ggplot(data, aes(x = image_views)) +
  geom_histogram(fill = "darkgreen", color = "black", alpha = 0.7) +
  labs(title = "Distribución de número de vistas",
       x = "Número de Vistas",
       y = "Frecuencia") +
  theme_minimal()
```



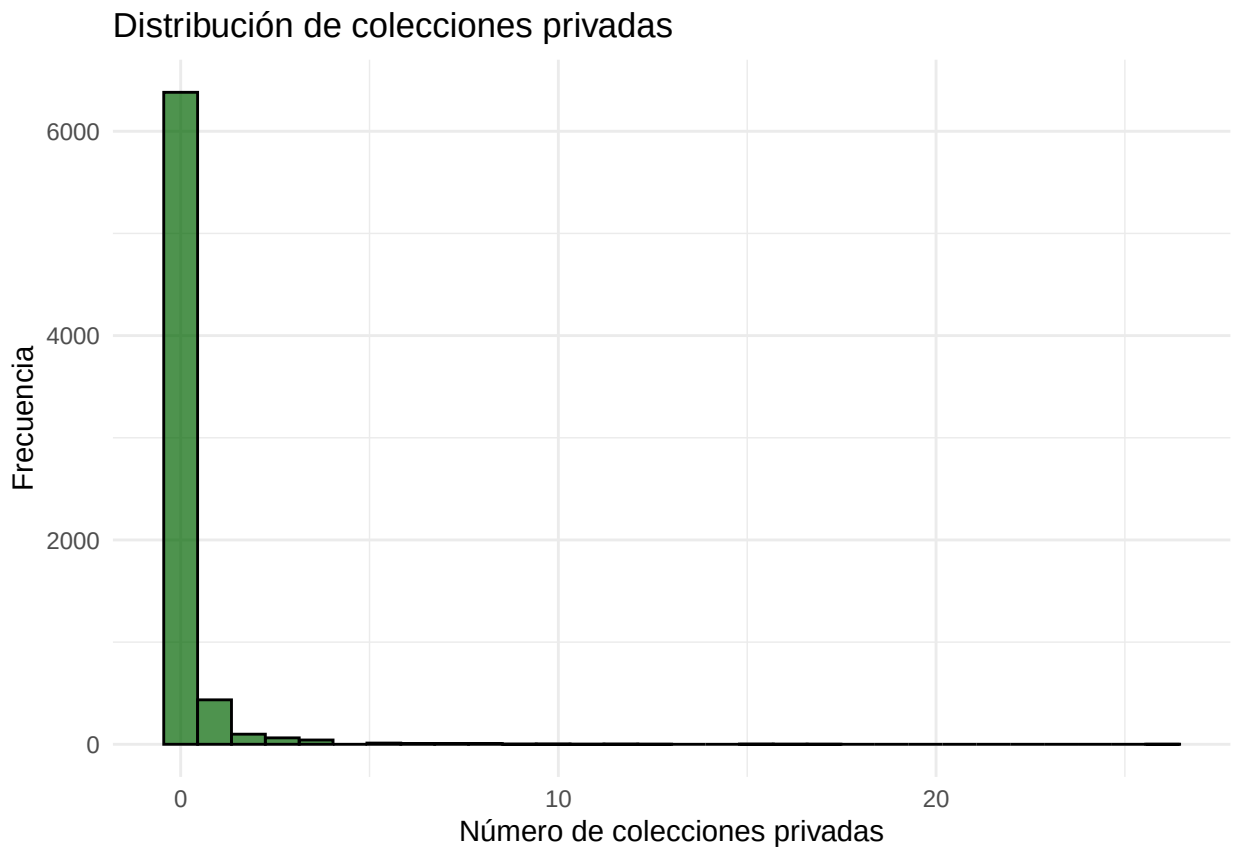

```r
# Boxplot de la variable image_views
ggplot(data, aes(x = image_views)) +
  geom_boxplot(fill = "darkblue", color = "black", alpha = 0.7) +
  labs(title = "Boxplot de número de vistas",
       x = "Número de Vistas",
       y = "Frecuencia") +
  theme_minimal()
```



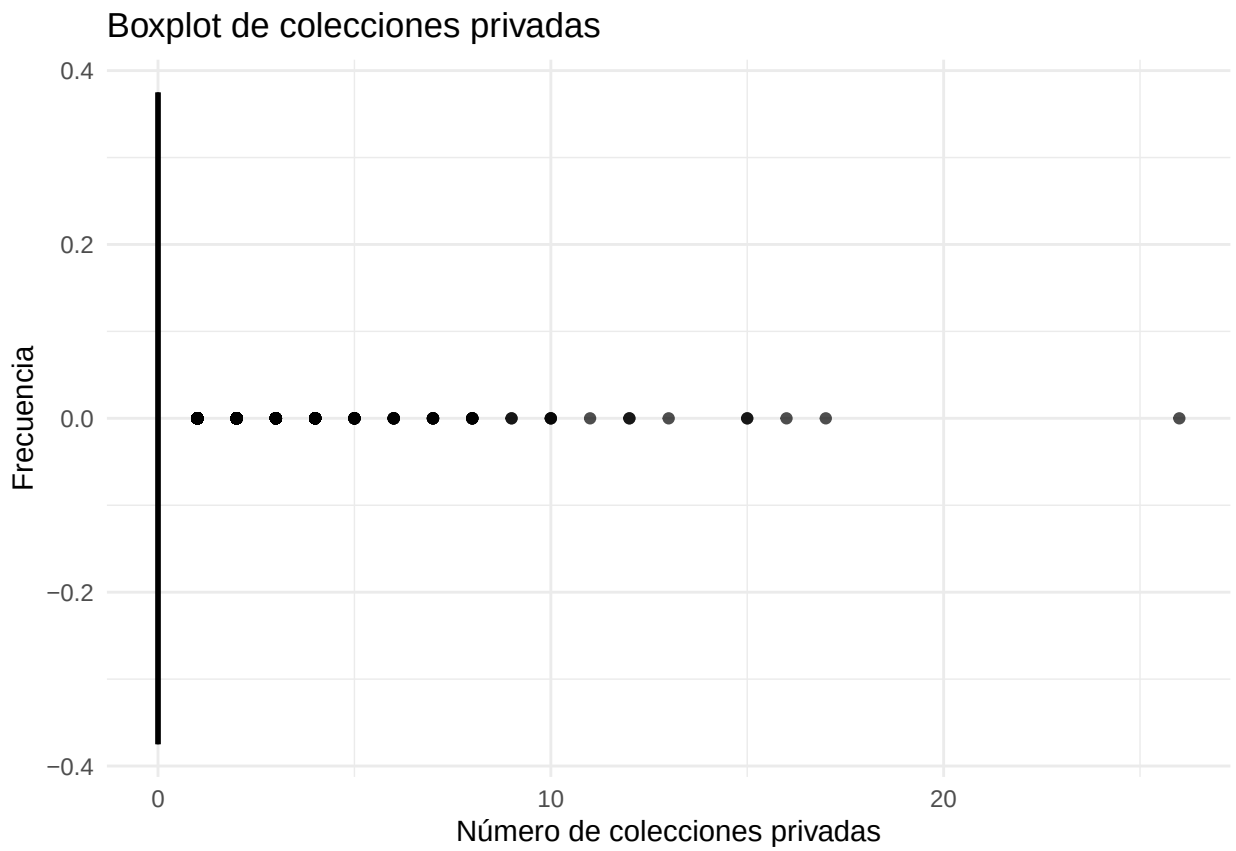
Esto indica la presencia de outliers en esta variable. Además que es mucho más sesgada a la iziquierda que las variables de comentarios y favoritos en donde podemos ver que la mayoría de imágenes tienen una exposición muy baja por lo que seguramente se podrá relacionar a su ubicación en la búsqueda o en el número de página en la que se encuentran.

- La variable `private_collections` es numérica, con un valor mínimo de 0, un valor máximo de 26, una media de 0.2, con esta distribución:

```
# Histograma de la variable private_collections
ggplot(data, aes(x = private_collections)) +
  geom_histogram(fill = "darkgreen", color = "black", alpha = 0.7) +
  labs(title = "Distribución de colecciones privadas",
       x = "Número de colecciones privadas",
       y = "Frecuencia") +
  theme_minimal()
```



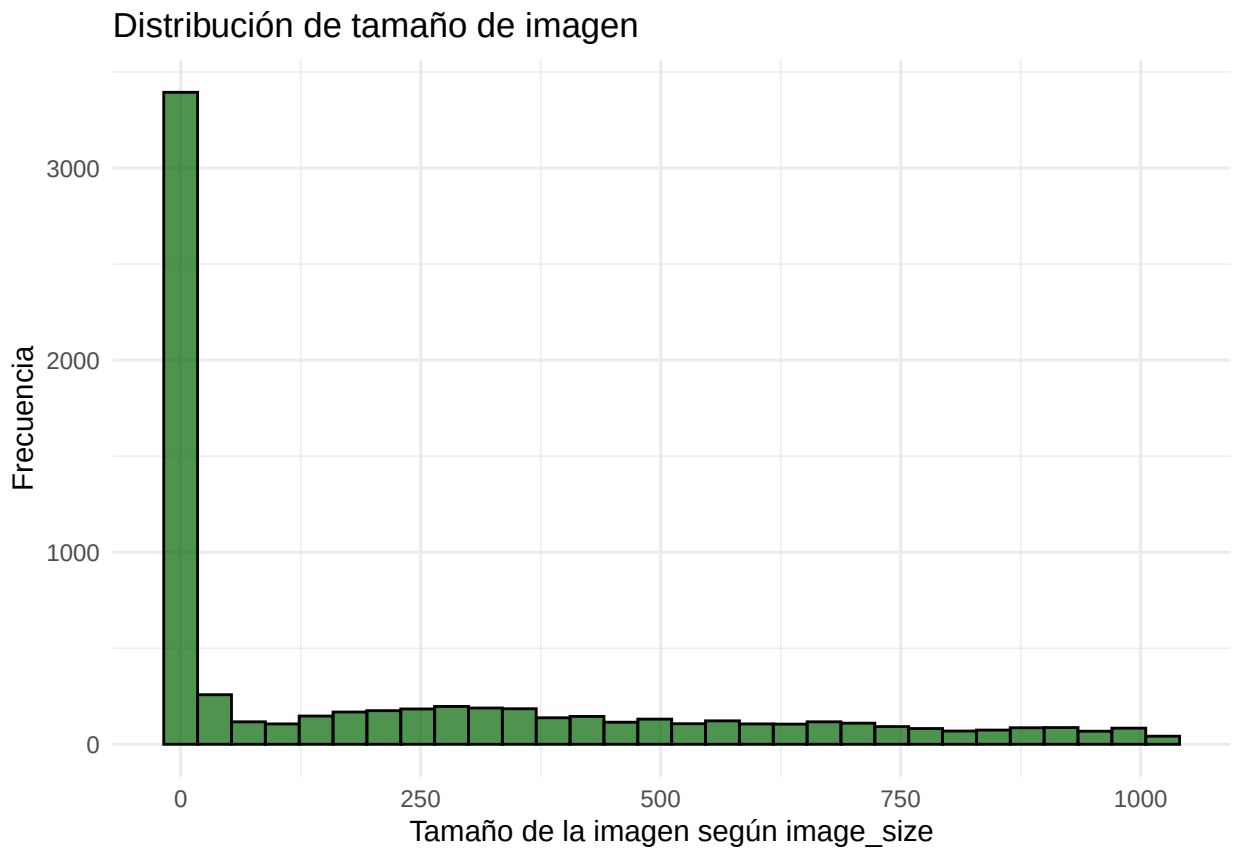

```r
# Boxplot de la variable private_collections
ggplot(data, aes(x = private_collections)) +
  geom_boxplot(fill = "darkblue", color = "black", alpha = 0.7) +
  labs(title = "Boxplot de colecciones privadas",
       x = "Número de colecciones privadas",
       y = "Frecuencia") +
  theme_minimal()
```



El 90.29% de las imágenes del dataset no está en ninguna colección privada. Puede ser una variable que no aporte demasiada información. Sin embargo la analizaremos más adelante en detalle.

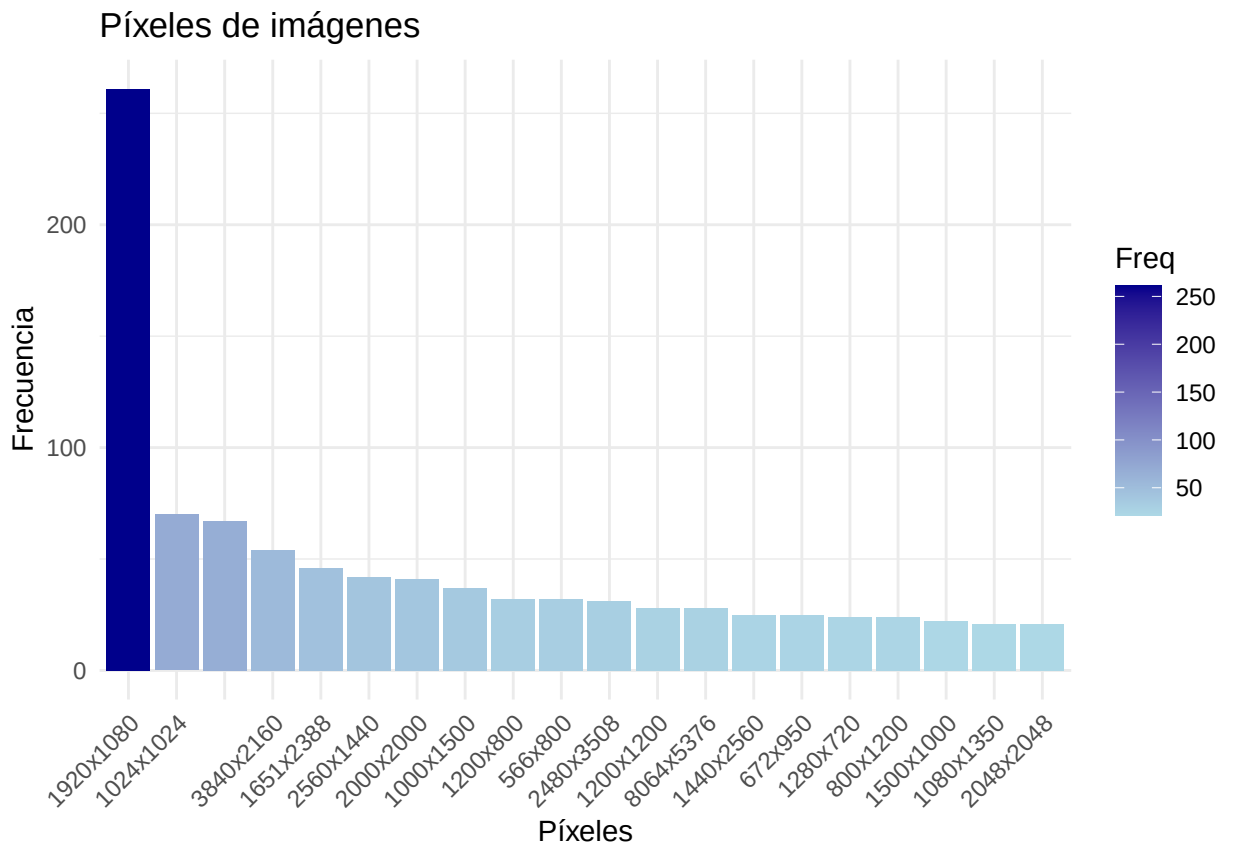
- La variable `image_size` es numérica, con un valor mínimo de 1, un valor máximo de 1023.92, una media de 227.92. Esta variable necesitará ser transformada, puesto que la distribución nos indica que aquellos valores pequeños corresponden a MB mientras que aquellos valores grandes son KB. Será necesaria una transformación a una sola escala para poder analizar mejor esta variable:

```
# Exploración de la variable image_size
ggplot(data, aes(x = image_size)) +
  geom_histogram(fill = "darkgreen", color = "black", alpha = 0.7) +
  labs(title = "Distribución de tamaño de imagen",
       x = "Tamaño de la imagen según image_size",
       y = "Frecuencia") +
  theme_minimal()
```



- Por otro lado la variable `image_px` es una variable categórica que contiene el número de pixeles que contiene la imagen en alto y ancho. Podemos ver que tenemos un número extenso de resoluciones de cada imagen sin embargo el grupo más importante es el de aquellas imágenes que tienen una resolución de 1920X1080. Esta variable guarda además relación con la distribución de `image_size`. Ambas variables se concentran a la izquierda.

```
frecuencias <- table(data$image_px)
df_frecuencias <- as.data.frame(frecuencias)
df_frecuencias_ordenado <- df_frecuencias[order(-df_frecuencias$Freq), ]

ggplot(head(df_frecuencias_ordenado, 20), aes(x = reorder(Var1, -Freq), y = Freq, fill = Freq)) +
  geom_bar(stat = "identity") +
  scale_fill_gradient(low = "lightblue", high = "darkblue") +
  labs(title = "Píxeles de imágenes",
       x = "Píxeles",
       y = "Frecuencia") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



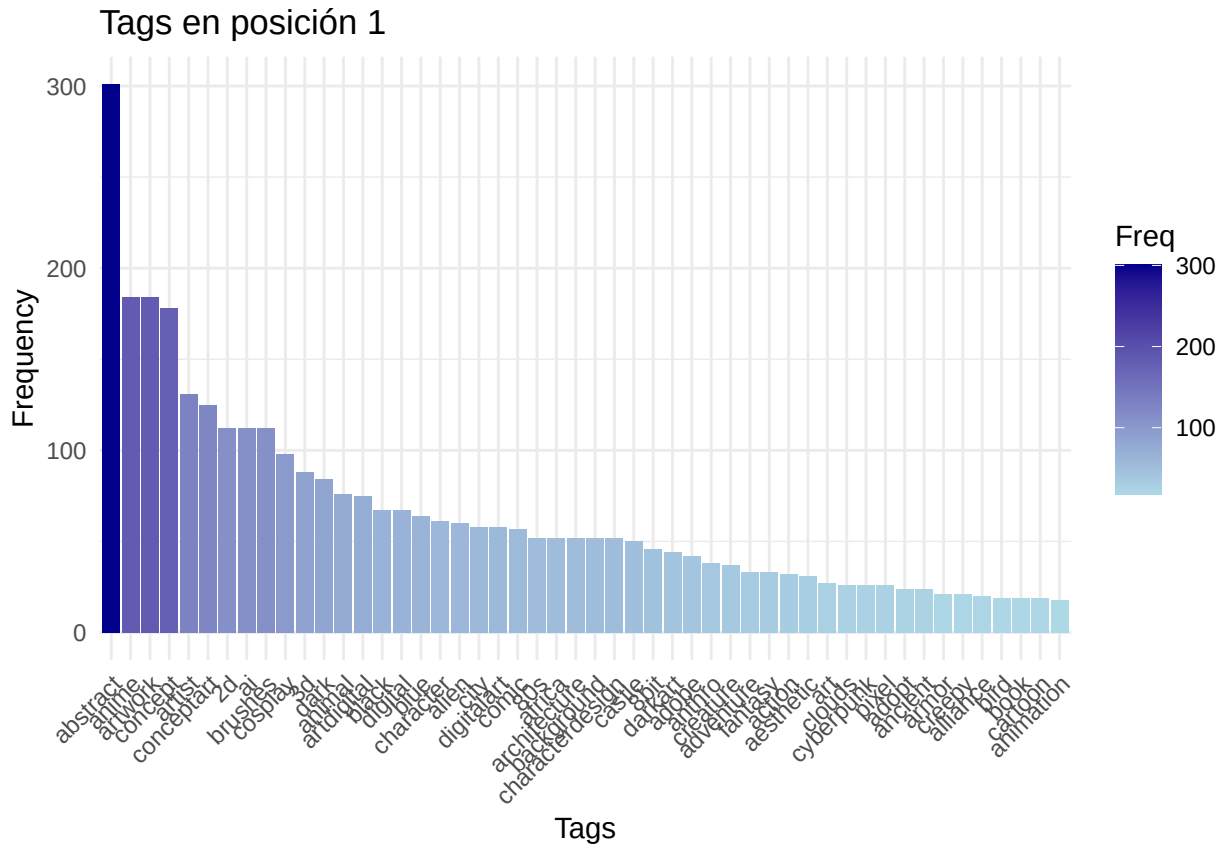
- La variable `tags` es de tipo character e incluye una lista de tags con los que el autor/a de la imagen la ha etiquetado.

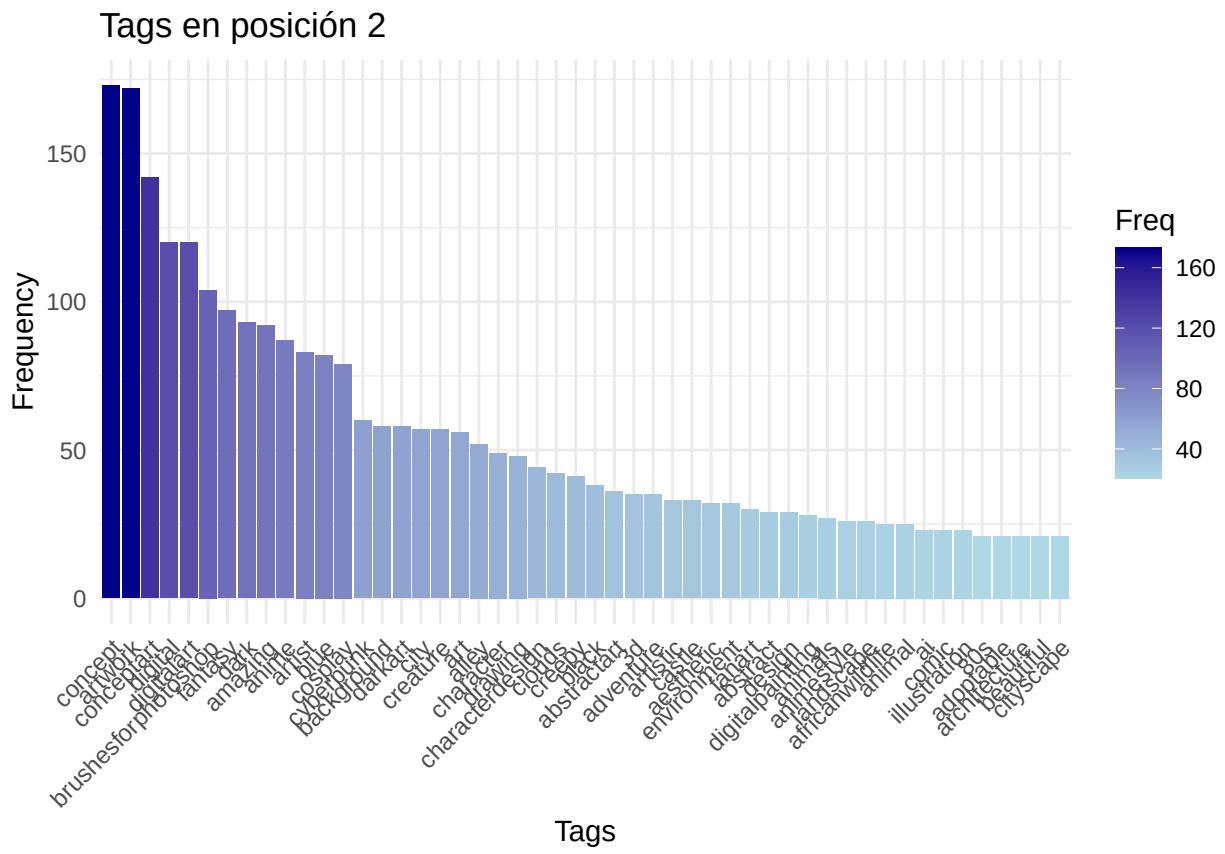
Para esta variable analizaremos los tags mas utilizados. Para ello elegimos la posicion que tiene el tag dentro de la lista de tags que se estan en cada imagen. Por ejemplo si elegimos la posición 1 obtendremos solamente el primer tag de cada imagen el cual podríamos decir que es el más importante por imagen. A continuación veremos los tags más comunes en las primeras 5 posiciones.

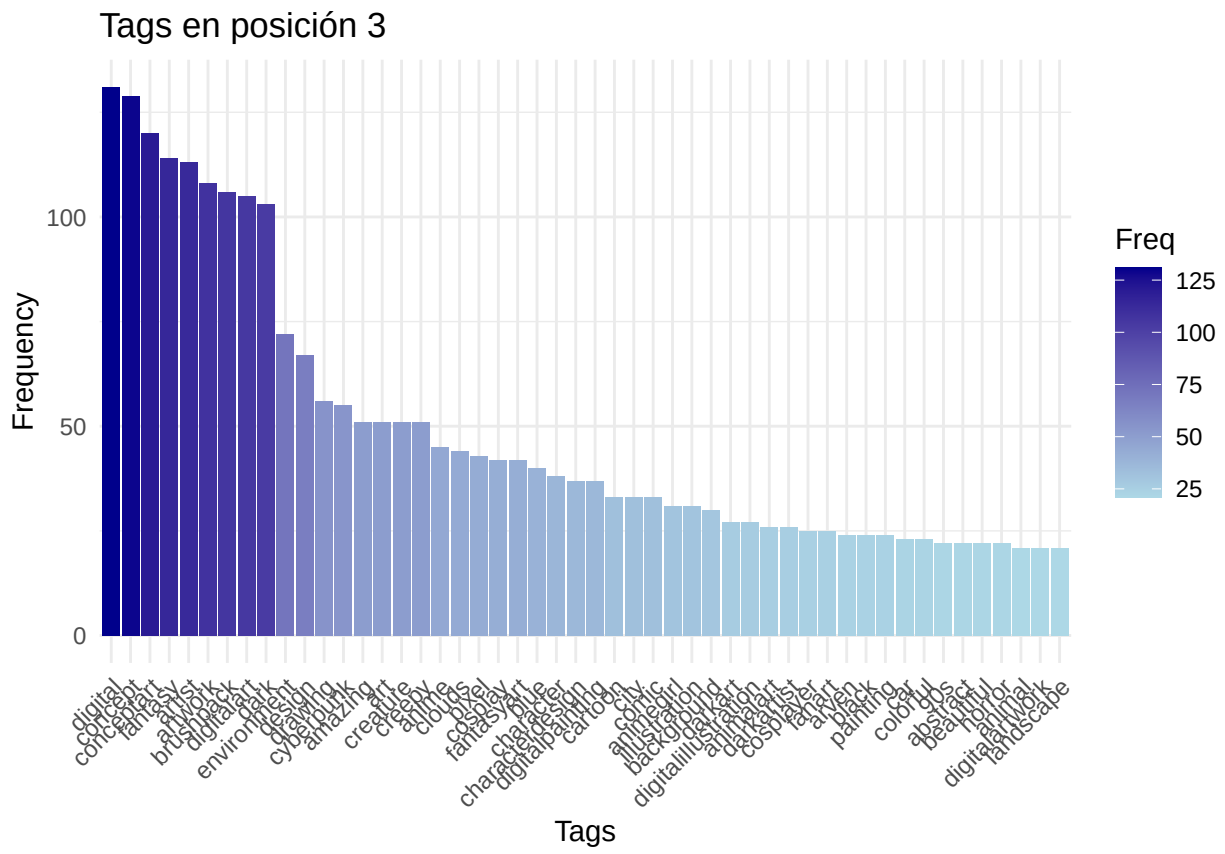
Podemos ver que entre los tags más comunes en la posición 1 están “concept”, “anime”, “artwork”, “concept”, “conceptart” y “digitalart”.

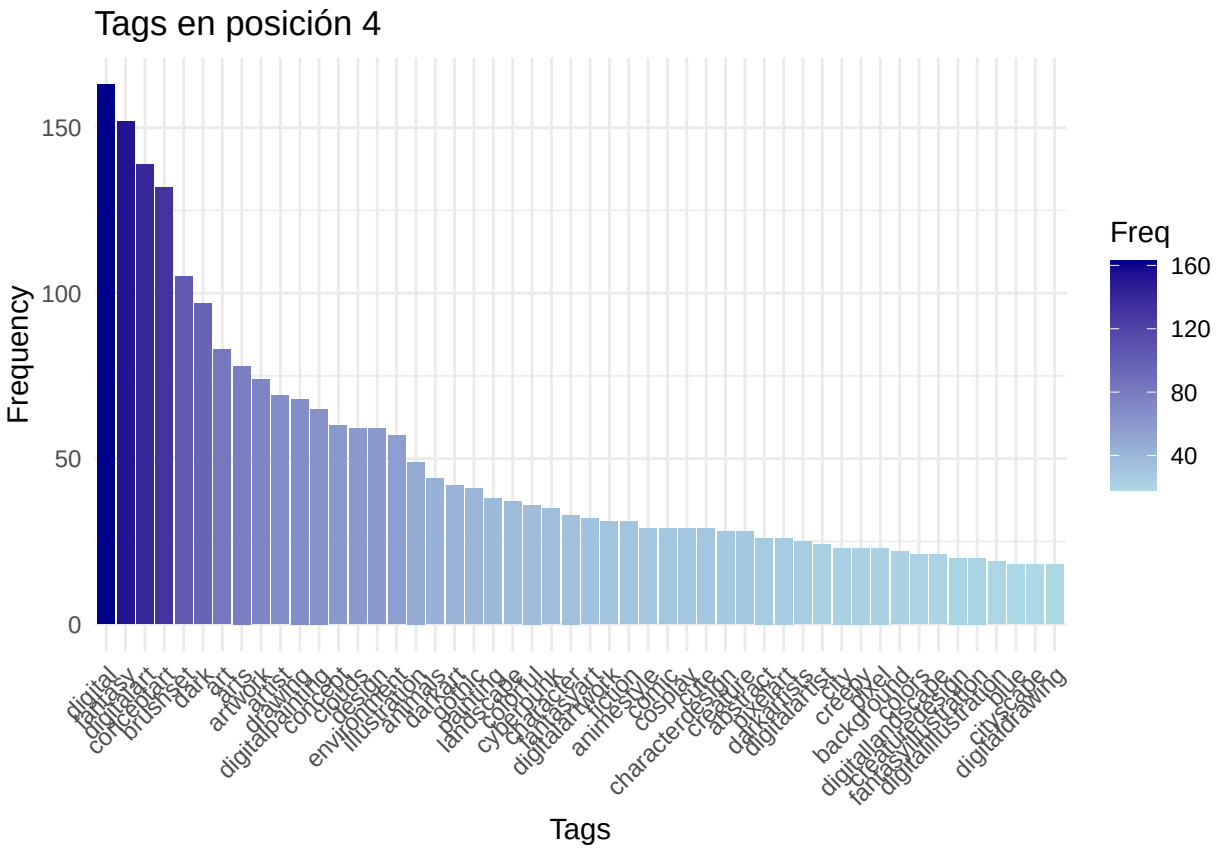
```
# Exploración de la frecuencia de los tags más comunes

# Función para limpiar y convertir las cadenas de tags en listas de tags
clean_and_split_tags <- function(tags_string, position) {
  # Elimina corchetes y comillas
  tags_string <- gsub("\\[|\\]|'", "", tags_string)
  if (tags_string == "") {
    # Devuelve un vector vacío si la cadena está vacía
    return(NA)
  }
  strsplit(tags_string, ", ")[[1]][position]  # Dividir la cadena en una lista de tags
}

# Aplicar la función a la columna de tags
#data$tags_list <- lapply(data$tags, clean_and_split_tags, 1)
```

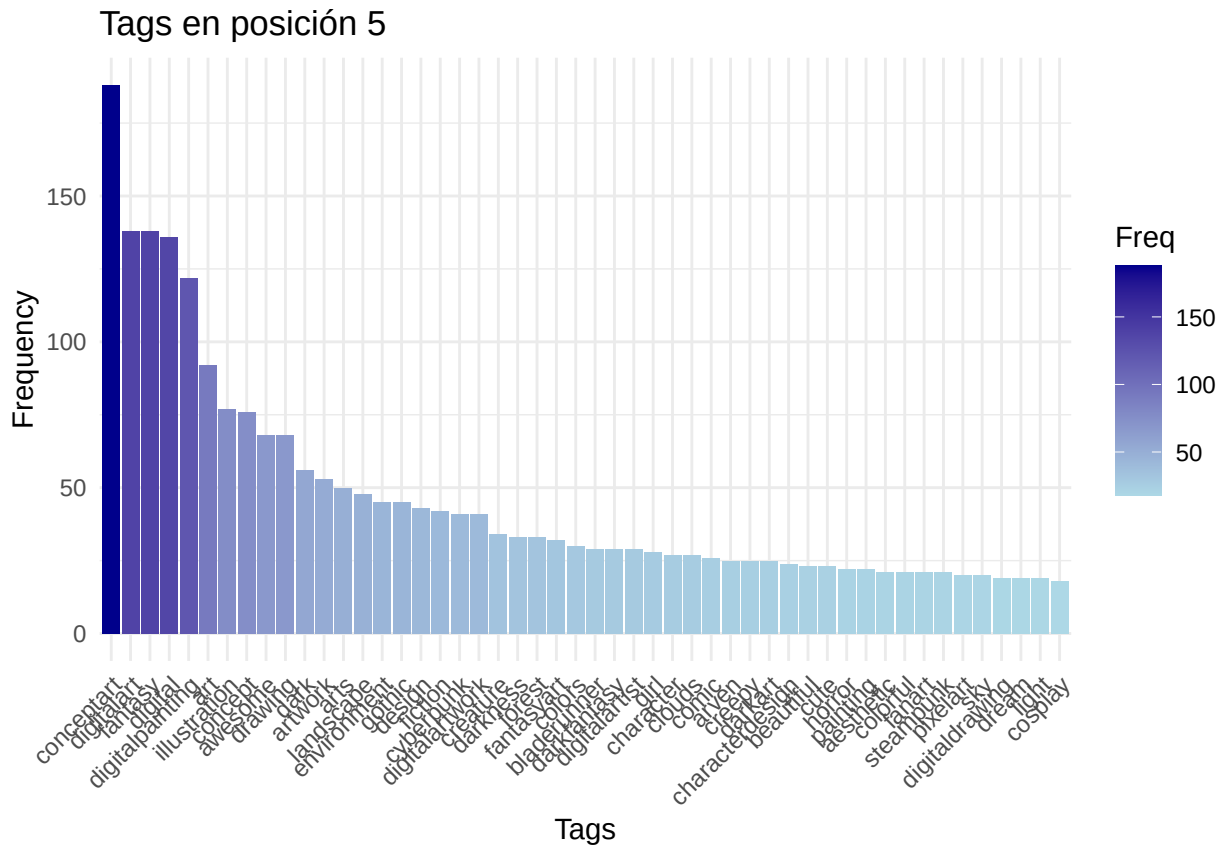
```r
for (i in 1:5) {
# Crear lista total de tags
all_tags <- unlist(lapply(data$tags, clean_and_split_tags, i))

# Crear una tabla de frecuencias
tag_frequencies <- table(all_tags)
df_frecuencias <- as.data.frame(tag_frequencies)
df_frecuencias_ordenado <- df_frecuencias[order(-df_frecuencias$Freq), ]

p<-ggplot(head(df_frecuencias_ordenado, 50), aes(x = reorder(all_tags, -Freq), y = Freq, fill = Freq)) +
  geom_bar(stat = "identity") +
  scale_fill_gradient(low = "lightblue", high = "darkblue") +
  labs(title = paste("Tags en posición", i),
       x = "Tags",
       y = "Frequency") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

print(p)

}
```




Tags en posición 2
Frequency
150
100
50
0
Freq
160
120
80
40
concept
artwork
conceptart
digitalart
digitalpainting
brushesforphotoshop
fantasy
dark
amazing
anime
artist
blue
cosplay
cyberpunk
background
darkart
creature
art
alien
character
drawing
characterdesign
cgsociety
creepy
black
abstractart
adventure
artistic
cgi
aesthetic
environment
fanart
abstract
design
digitalpainting
animation
animestyle
landscape
africanwildlife
animal
ai
comic
illustration
sci
adoptable
architecture
beautiful
cityscape
Tags


Tags en posición 3
Frequency
100
50
0
Tags
Freq
125
100
75
50
25


Tags en posición 4
Frequency
150
100
50
0
Tags
Freq
160
120
80
40



En el apartado de limpieza de los datos, se limpiarán las variables `location` (candidata a ser eliminada por número de NA), `published_date` (conversión a tipo Date), `image_license`, unificando las licencias copyright y convirtiéndola a categórica, `image_px` (unificamos formato en ancho x largo y extraemos superficie de la imagen, el resto de valores los convertimos a NA). Se desarrollarán en el apartado de limpieza de los datos, entre otras transformaciones que se realizarán.

Por otro lado se transformará en una misma unidad a la variable `image_size` se la dejará a todo en Kilobytes multiplicando por 1024 los valores que se encuentran en MB.

2. Integración y selección de los datos de interés a analizar

Puede ser el resultado de adicionar diferentes datasets o una subselección útil de los datos originales, en base al objetivo que se quiera conseguir. Si se decide trabajar con una selección de los datos, es muy importante que esta esté debidamente justificada. Además, se recomienda mostrar un resumen de los datos que permita ver a simple vista las diferentes variables y sus rangos de valores.

Elegimos variables que nos parecen relevantes para un análisis numérico mientras que dejamos fuera otras variables de texto libre, como `description`, `image_url`, `image_page`, `last_comment`.

Eliminanos las variables con gran número de NA (`location`). Mantener una de las dos variables de dimensiones: `size` o `superficie`, eliminar la otra si es que encontramos una correlación grande entre las dos.

- La variable `location` tiene un alto porcentaje de valores vacíos (“ “), que constituyen el 96.79% de los registros. Decidimos no incluir esta variable en el análisis posterior.
- Esta selección de variables será parte de la limpieza de datos. que viene en el siguiente apartado.

3. Limpieza de los datos

Convertimos todos los valores en el dataset que son texto vacío (“ “) en NA, puesto que se trata de valores ausentes.

```
# Convertir todos los valores vacíos ("") en el dataset a NA
data <- data.frame(lapply(data, function(x) {
  x[x == ""] <- NA
  return(x)
}), stringsAsFactors = FALSE)
```

Valores ausentes de cada variable: Tenemos presencia de NAs en variables como `image_license`, `image_px` e `image_size` aunque no es un problema demasiado grande. Se puede optar por eliminar estas observaciones pues son un número reducido de casos.

```
colSums(is.na(data))
```

```
##        search_topic            page_num          image_page           image_url
##                   0                   0                   0                   0
##         image_title        image_author          image_favs           image_com
##                   0                   0                   0                   0
##         image_views private_collections                tags            location
##                   0                   0                   0                6840
##         description            image_px          image_size      published_date
##                 453                  67                  67                   0
##        last_comment       image_license
##                2734                 135
```

Transformaciones de limpieza de datos:

1. Se convierte la variable `search_topic` a factor.
2. Se convierte la variable `published_date` a un objeto de fecha y hora (`datetime`).
3. Se extrae el año de la variable `published_date` y se almacena en una nueva columna `year`.
4. Se extrae el mes de la variable `published_date` y se almacena en una nueva columna `month`.
5. Se convierte el día de la semana de la variable `published_date` a factor y se almacena en una nueva columna `day_of_week`.
6. Se extrae la hora de la variable `published_date` y se almacena en una nueva columna `hour`.
7. Se crea una nueva variable `moment_of_day` que categoriza las horas en “Morning”, “Afternoon”, “Evening” y “Night”.
8. Se convierte la variable `image_license` a factor, identificando si contiene un símbolo de copyright y etiquetándola como “Copyright”.
9. Se crea una nueva variable `superficie_px2` que calcula el área en píxeles multiplicando las dimensiones de la imagen.
10. Se convierte la variable `image_size` a kilobytes si el tamaño está en bytes.
11. Se limpia y divide la variable `tags` en una lista de etiquetas.
12. Se crea una nueva variable `image_favs_cat` que categoriza el número de favoritos en cuartiles: “Very Low”, “Low”, “Medium” y “High”.
13. Se seleccionan las columnas relevantes para el análisis.
14. Se eliminan las filas con valores NA.

Definición del preprocesamiento:

1. Se elimina cualquier fila que contenga NA en las variables predictoras.

2. Se aplica una transformación logarítmica a las variables `superficie_px2`, `image_size_kb`, `image_com`, `image_views` y `image_favs` para estabilizar la varianza y manejar la escala.
3. Se normalizan las variables `superficie_px2`, `image_size_kb`, `image_com`, `image_views` y `image_favs` para que tengan una media de 0 y una desviación estándar de 1.

Receta de limpieza de datos

```
# Preprocesado de datos para dejarlos en un formato más útil que al inicio.
data_parsed <- data %>%
  mutate(search_topic = as.factor(search_topic),
         published_date = as_datetime(published_date),
         year = year(published_date),
         month= month(published_date),
         day_of_week =as.factor(weekdays(published_date)),
         hour = hour(published_date),
         moment_of_day = as.factor(case_when(
                         hour >= 5 & hour < 12 ~ "Morning",
                         hour >= 12 & hour < 17 ~ "Afternoon",
                         hour >= 17 & hour < 21 ~ "Evening",
                         hour >= 21 | hour < 5 ~ "Night"
                         )),
         image_license = as.factor(ifelse(grepl("©", data$image_license),
                                 "Copyright", image_license)),
         image_px = ifelse(grepl("x", image_px), image_px, NA),
         superficie_px2 =sapply(strsplit(as.character(image_px), "x"),
                                function(dim) as.numeric(dim[1]) * as.numeric(dim[2])),
         image_size_kb = ifelse(
         nchar(gsub("\\..*", "", image_size)) < 3,
           image_size * 1024,
           image_size ),
         tags_list_1 = lapply(tags, clean_and_split_tags, 1),
         image_favs_cat = cut(image_favs,
                                breaks = quantile(image_favs,
                                                  probs = seq(0, 1, by = 0.25)),
                                labels = c("Very Low", "Low", "Medium", "High"),
                                include.lowest = TRUE)
          ) %>% select(
            all_of(c("search_topic",
                     "year",
                     "month",
                     "hour",
                     "day_of_week",
                     "moment_of_day",
                     "image_license",
                     "superficie_px2",
                     "image_size_kb",
                     "image_com",
                     "image_views",
                     "page_num",
                     "private_collections",
                     "image_favs_cat",
                     "image_favs")
```

```
                    )
            ) %>% na.omit()

# Receta del preprocesamiento de las variables resultantes del data frame.
recipe <- data_parsed %>%
              recipe(image_favs_cat ~ . ) %>% ## Crea la receta
              step_naomit(all_predictors(), -all_outcomes()) %>%  # Elimina filas con NA
              step_log(c("superficie_px2", "image_size_kb",
                         "image_com", "image_views",
                         "image_favs"), offset = 1) %>%
              step_normalize(c("superficie_px2", "image_size_kb",
                           "image_com", "image_views", "image_favs"))

train <-prep(recipe, data_parsed)
data_preprocessed <- bake(train, data_parsed)

# Save the dataframe to a CSV file
write.csv(data_preprocessed, file = "../data/images_db_preprocessed.csv", row.names = FALSE)
```

Preparación de datos

Este fragmento de código realiza las siguientes operaciones:

1. **Fijar la semilla para la reproducibilidad**:
   - Se establece una semilla para asegurar que los resultados sean reproducibles cada vez que se ejecuta el código.
2. **Crear conjuntos de entrenamiento y prueba**:
   - Se divide el conjunto de datos `data_parsed` en conjuntos de entrenamiento (80%) y prueba (20%) de manera estratificada, asegurando que la distribución de la variable `image_favs_cat` se mantenga en ambos conjuntos.
3. **Preprocesar los conjuntos de datos**:
   - Se aplican las transformaciones definidas en la receta de preprocesamiento (`train`) tanto al conjunto de entrenamiento como al conjunto de prueba.
4. **Seleccionar variables relevantes**:
   - Se seleccionan únicamente las columnas relevantes para el análisis y se eliminan las demás.
5. **Eliminar filas con valores faltantes**:
   - Se eliminan las filas que contienen valores faltantes en ambos conjuntos de datos, `data_train` y `data_test`, para asegurar que solo se utilicen datos completos en el análisis posterior.

```
#  Partición de los datos

library(caret)
library(randomForest)

set.seed(123)

trainIndex <- createDataPartition(data_parsed$image_favs_cat, p = .8, list = FALSE, times = 1)
```

```
data_train <- bake(train, new_data = data_parsed[trainIndex,])
data_test <- bake(train, new_data = data_parsed[-trainIndex,])

# Eliminar las variables de texto y la numérica de image_favs
selected_vars = c('image_favs_cat', 'search_topic', 'page_num', 'image_com', 'image_views', 'image_size

data_train <- data_train[ , (names(data_train) %in% selected_vars)]
data_test <- data_test[ , (names(data_test) %in% selected_vars)]

# Eliminar filas con valores perdidos en data_train y data_test
data_train <- data_train[complete.cases(data_train), ]
data_test <- data_test[complete.cases(data_test), ]
```

EDA

A continuacion realizamos una analisis exploratorio de las variables y su relación con la variable objetivo.

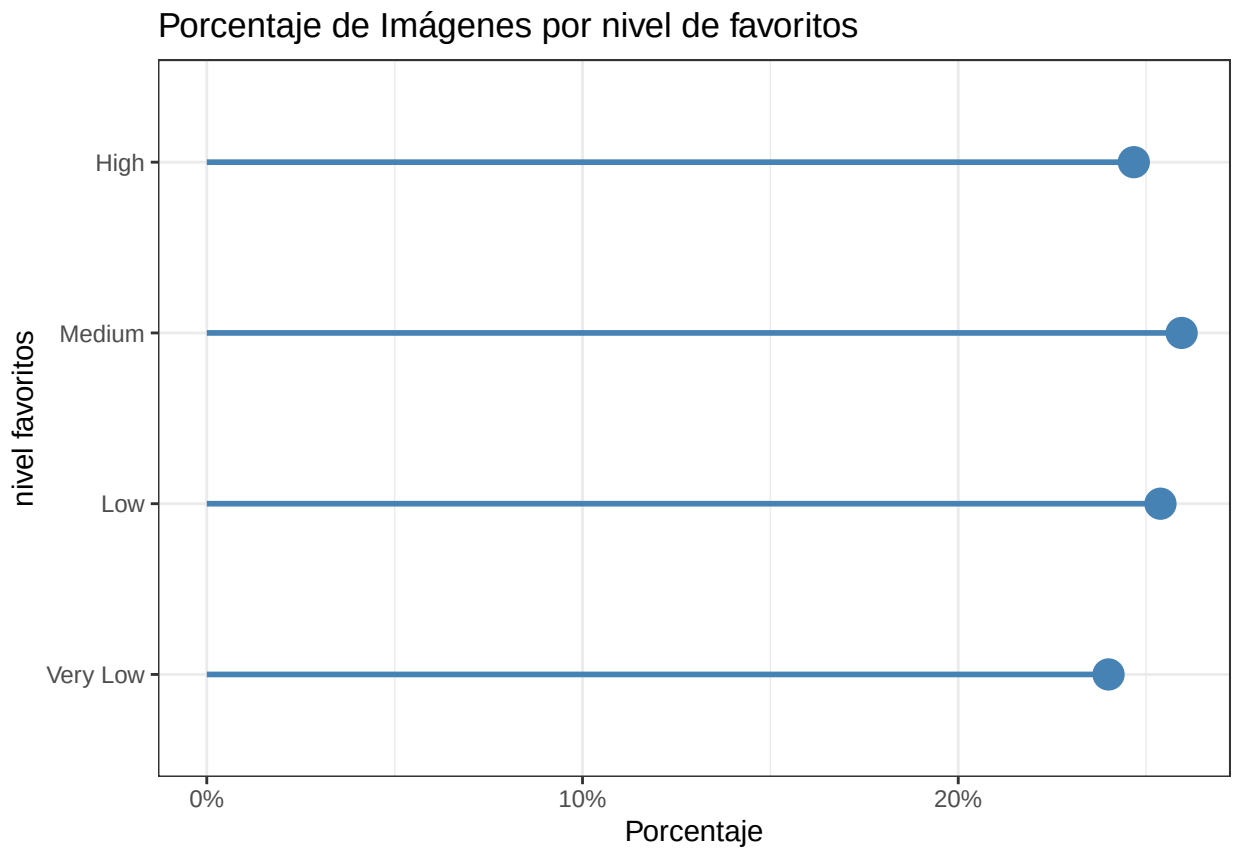
La variable objetivo del modelo supervisado es `image_favs_cat`, que contiene 4 categorías que agrupan al número de favs que tiene una imagen (very low, low, medium, high). En este caso, el problema es de clasificación y no de regresión, ya que `image_favs_cat` contiene categorías en lugar de valores numéricos continuos. Vamos a usar un modelo supervisado de clasificación, como Random Forest, para predecir estas categorías.

Antes de realizar esta modelización vamos a analizar los datos disponibles.

Análisis de la Variable Objetivo

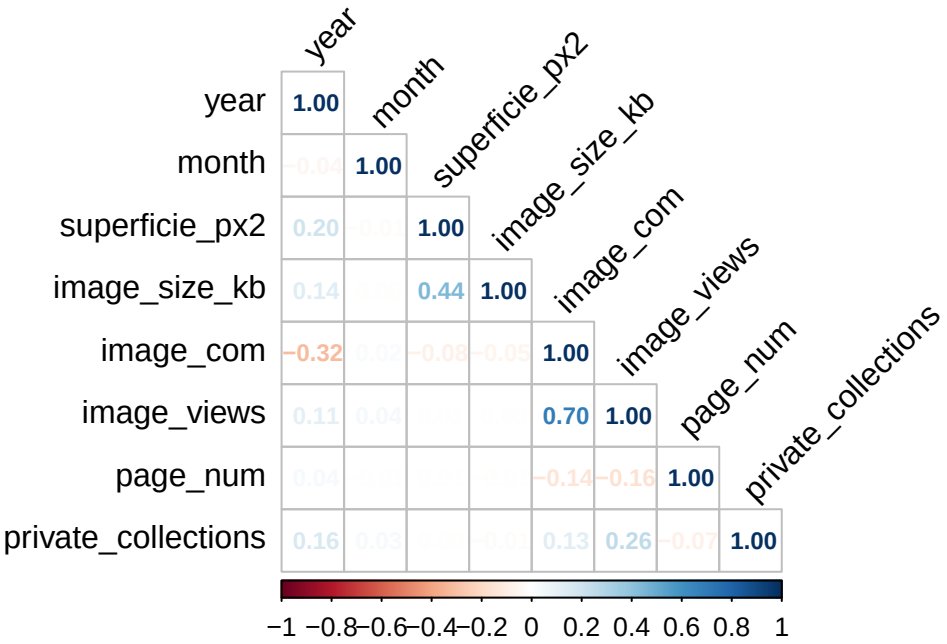
Al analizar la variable objetivo `image_favs_cat`, observamos que tenemos un número balanceado de casos en cada una de las categorías. Esto significa que las clases de la variable `image_favs_cat` están bien representadas y distribuidas de manera equitativa en el conjunto de datos, lo que es beneficioso para el rendimiento de los modelos de predicción. Un conjunto de datos balanceado ayuda a evitar sesgos hacia una clase específica y permite que los algoritmos de aprendizaje automático aprendan de manera más efectiva las características de cada categoría.

```
# Ver los rangos de cada categoría

data_train %>%
    group_by( image_favs_cat) %>%
    count( name = 'frec') %>%
    ungroup() %>%
    mutate( Porc= frec/sum(frec)) %>%
    ggplot( aes(x= image_favs_cat, y= Porc)) +
    geom_segment( aes(xend= image_favs_cat, y=0, yend=Porc),
    color= "steelblue", linewidth= 1) +
    geom_point( size=5, color= "steelblue") +
    coord_flip() +
    scale_y_continuous( labels = percent_format()) +
    labs(title= 'Porcentaje de Imágenes por nivel de favoritos',
    y= "Porcentaje", x= "nivel favoritos") +
    theme_bw()
```



Análisis de Correlación entre Variables

```
data_train %>%
  select_if(is.numeric) %>%
  cor() %>%
  corrplot::corrplot(
    method = "number",
    type = "lower",
    tl.col = "black",
    tl.srt = 45,
    number.cex = 0.75,
    cl.pos = "b",
    addCoef.col = "black",
    title = "Correlation Matrix",
    mar = c(0,0,2,0)
  )
```



Al analizar la correlación entre las variables, observamos que solo existe una alta correlación entre las variables `image_com` e `image_views`, con un coeficiente de correlación de 0.7. Esta alta correlación podría no ser tan problemática para incluirlas en el análisis, ya que no alcanza el umbral de 0.8 que a menudo se considera crítico para la multicolinealidad. La multicolinealidad puede afectar la interpretación de los coeficientes en los modelos de regresión, pero con un valor de correlación de 0.7, es probable que las variables aún aporten información única y valiosa al modelo.

Por otro lado, las demás variables presentan valores de correlación más bajos, lo que indica que no están fuertemente correlacionadas entre sí. Esto es deseable ya que permite que cada variable proporcione información única y diversa al modelo de predicción, evitando redundancias y mejorando la capacidad del modelo para capturar la variabilidad en los datos.

Análisis de Scatter Plots

```
# Crear los scatter plots
plot1 <- ggplot(data_train, aes(x = image_size_kb, y = superficie_px2, color = image_favs_cat)) +
  geom_point() +
  labs(title = "Scatter Plot 1: image_size_kb vs superficie_px2",
       x = "image_size_kb",
       y = "superficie_px2",
       color = "image_favs_cat")

plot2 <-
  ggplot( data_train %>% filter(image_views < max(image_views)), aes(x = image_com, y = image_views, co
  geom_point() +
```

```
  labs(title = "Scatter Plot 2: image_com vs image_views",
       x = "image_com",
       y = "image_views",
       color = "image_favs_cat")

# Crear un grid de los scatter plots
library(gridExtra)
grid.arrange(plot1, plot2, ncol = 1, nrow=2)
```

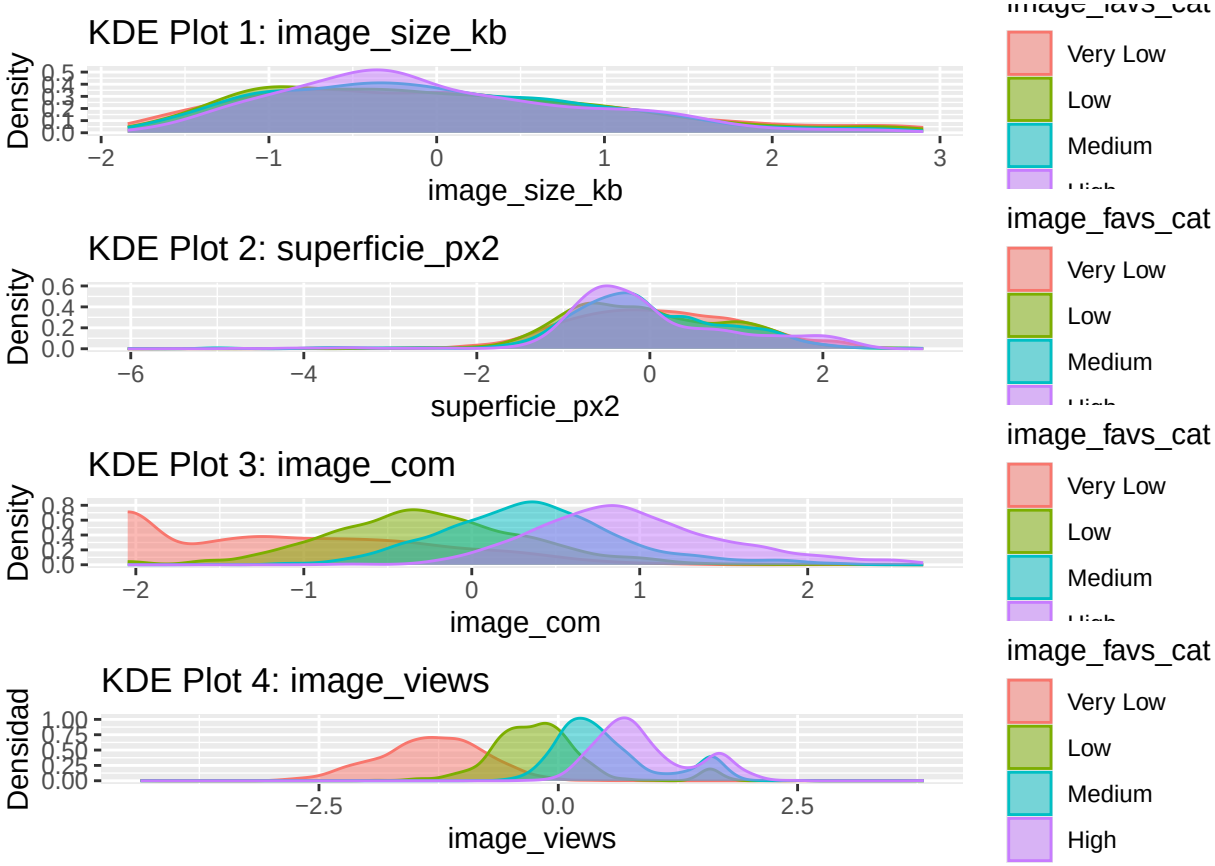
En estos scatter plots entre las variables predictoras y la variable objetivo, observamos lo siguiente:

1. **Scatter Plot 1: image_size_kb vs superficie_px2**:
   - En este gráfico, no se observa una clara separación de los grupos de la variable objetivo (`image_favs_cat`). Los puntos se distribuyen de manera dispersa y no muestran una tendencia definida que permita distinguir claramente entre las diferentes categorías de la variable objetivo.
2. **Scatter Plot 2: image_com vs image_views**:
   - En este gráfico, se observa una mejor separación de los grupos de la variable objetivo (`image_favs_cat`). Se puede notar claramente cómo los valores más altos en las variables `image_com` y `image_views` tienden a estar asociados con los grupos "High" y "Medium", mientras que los valores más bajos se relacionan con los grupos "Low" y "Very Low". Esta separación sugiere que estas variables pueden ser buenas predictoras de la variable objetivo.

En resumen, mientras que las variables `image_views` y `image_com` muestran una clara separación de los grupos de la variable objetivo, las variables `image_size_kb` y `superficie_px2` no parecen ser tan efectivas para distinguir entre las diferentes categorías de la variable objetivo.

Análisis de Gráficos de Densidad (KDE Plots)

```r
# KDE plot 1: image_size_kb vs superficie_px2
kde_plot1 <- ggplot(data_train, aes(x = image_size_kb, fill = image_favs_cat, color = image_favs_cat)) +
  geom_density(alpha = 0.5) +
  labs(title = "KDE Plot 1: image_size_kb",
       x = "image_size_kb",
       y = "Density",
       fill = "image_favs_cat",
       color = "image_favs_cat")

kde_plot2 <- ggplot(data_train, aes(x = superficie_px2, fill = image_favs_cat, color = image_favs_cat)) +
  geom_density(alpha = 0.5) +
  labs(title = "KDE Plot 2: superficie_px2",
       x = "superficie_px2",
       y = "Density",
       fill = "image_favs_cat",
       color = "image_favs_cat")

# KDE plot 3: image_com vs image_views
kde_plot3 <- ggplot(data_train, aes(x = image_com, fill = image_favs_cat, color = image_favs_cat)) +
  geom_density(alpha = 0.5) +
  labs(title = "KDE Plot 3: image_com",
       x = "image_com",
       y = "Density",
       fill = "image_favs_cat",
       color = "image_favs_cat")

kde_plot4 <- ggplot(data_train, aes(x = image_views, fill = image_favs_cat, color = image_favs_cat)) +
  geom_density(alpha = 0.5) +
  labs(title = "KDE Plot 4: image_views",
       x = "image_views",
       y = "Densidad",
       fill = "image_favs_cat",
       color = "image_favs_cat")

# Crear un grid de los KDE plots
library(gridExtra)
grid.arrange(kde_plot1, kde_plot2, kde_plot3, kde_plot4, nrow = 4)
```



Al analizar estos gráficos de densidad de las variables predictoras en relación con la variable objetivo (`image_favs_cat`), observamos lo siguiente:

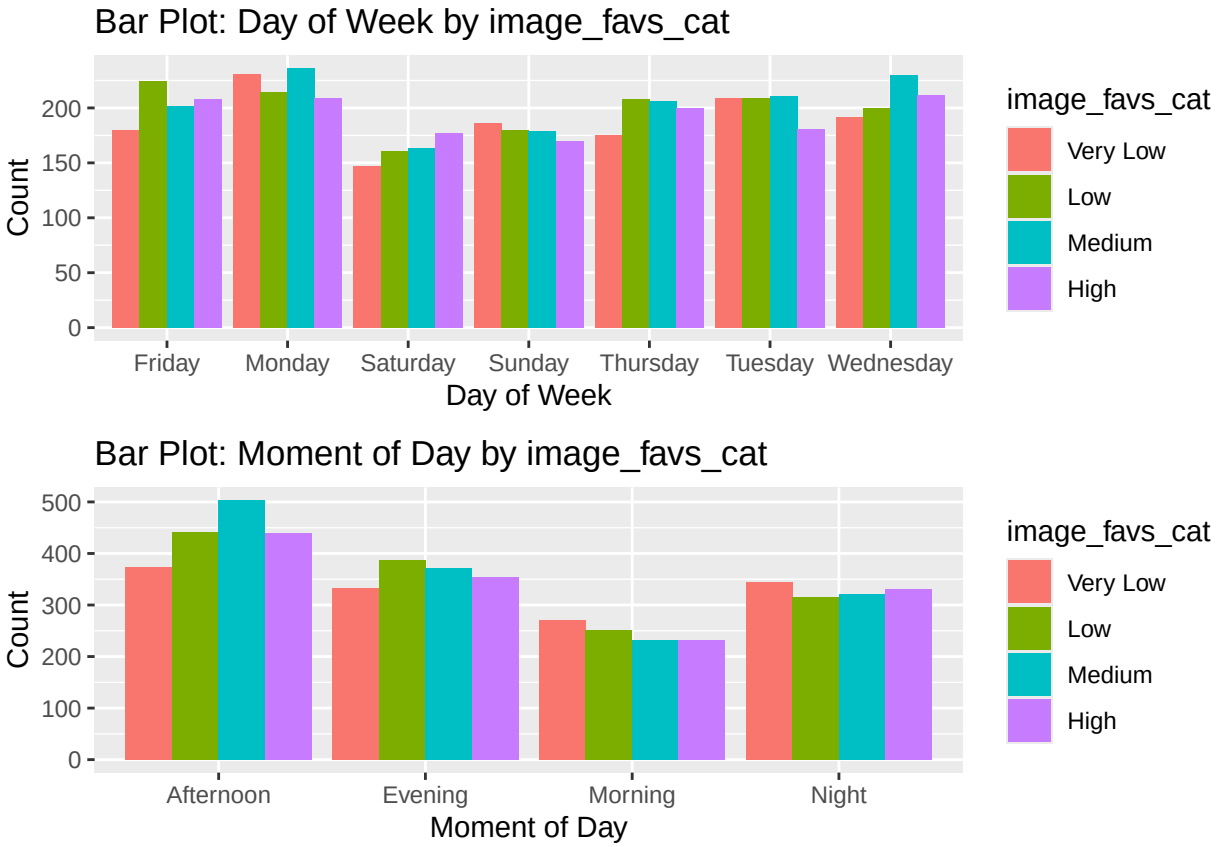
1. **KDE Plot 1: image_size_kb**:
   - En este gráfico de densidad, no se observa una clara diferencia en las distribuciones de las diferentes categorías de la variable objetivo. Las curvas de densidad se superponen en gran medida, lo que sugiere que la variable `image_size_kb` puede no ser muy discriminativa para predecir la variable objetivo.
2. **KDE Plot 2: superficie_px2**:
   - Al igual que en el gráfico anterior, las distribuciones de las diferentes categorías de la variable objetivo se superponen considerablemente. Esto indica que la variable `superficie_px2` puede no ser muy útil para distinguir entre las diferentes categorías de la variable objetivo.
3. **KDE Plot 3: image_com**:
   - En este gráfico de densidad, se observa una clara diferencia en las distribuciones de las diferentes categorías de la variable objetivo. Cada grupo de la variable objetivo (`image_favs_cat`) se distribuye en niveles más altos de `image_com`, confirmando lo observado en los scatterplots anteriores.
4. **KDE Plot 4: image_views**:
   - De manera similar al gráfico anterior, se observa una clara diferencia en las distribuciones de las diferentes categorías de la variable objetivo. Cada grupo de la variable objetivo se distribuye en niveles más altos de `image_views`, confirmando lo observado en los scatterplots anteriores.

En resumen, los gráficos de densidad confirman que las variables `image_views` e `image_com` muestran distribuciones especialmente diferentes para cada grupo de la variable objetivo, lo que sugiere que estas variables pueden ser predictores importantes de la variable objetivo.

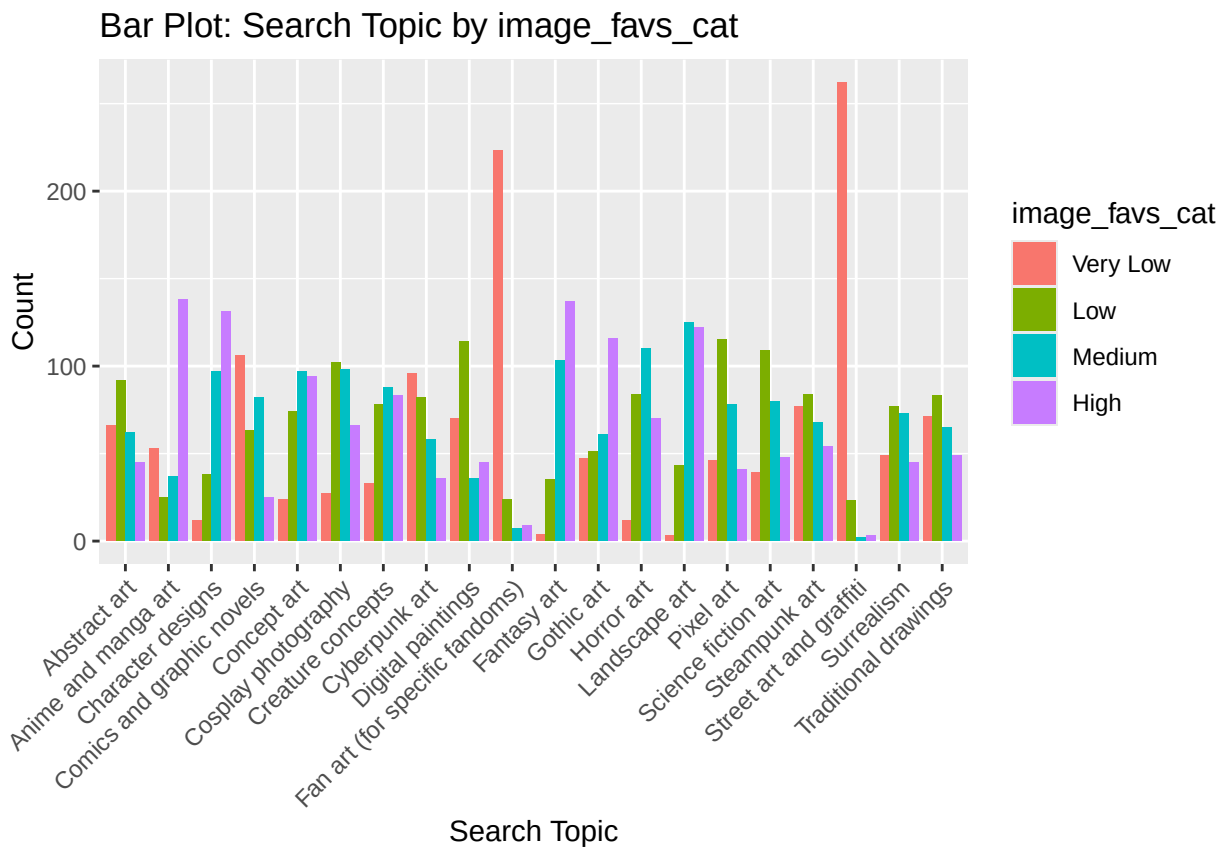
Conclusiones de los Gráficos de Barras

```
# Bar plot for day_of_week
bar_plot1 <- ggplot(data_train, aes(x = day_of_week, fill = image_favs_cat)) +
  geom_bar(position = "dodge") +
  labs(title = "Bar Plot: Day of Week by image_favs_cat",
       x = "Day of Week",
       y = "Count",
       fill = "image_favs_cat")

# Bar plot for moment_of_day
bar_plot2 <- ggplot(data_train, aes(x = moment_of_day, fill = image_favs_cat)) +
  geom_bar(position = "dodge") +
  labs(title = "Bar Plot: Moment of Day by image_favs_cat",
       x = "Moment of Day",
       y = "Count",
       fill = "image_favs_cat")

# Bar plot for search_topic
bar_plot3 <- ggplot(data_train, aes(x = search_topic, fill = image_favs_cat)) +
  geom_bar(position = "dodge") +
  labs(title = "Bar Plot: Search Topic by image_favs_cat",
       x = "Search Topic",
       y = "Count",
       fill = "image_favs_cat") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Bar plot for search_topic
bar_plot4 <- ggplot(data_train, aes(x = page_num, fill = image_favs_cat)) +
  geom_bar(position = "dodge") +
  labs(title = "Bar Plot: Page num por image_favs_cat",
       x = "Page num",
       y = "Count",
       fill = "image_favs_cat") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Bar plot for search_topic
bar_plot5 <- ggplot(data_train, aes(x = month, fill = image_favs_cat)) +
  geom_bar(position = "dodge") +
  labs(title = "Bar Plot: Mes por image_favs_cat",
       x = "mes",
       y = "Count",
       fill = "image_favs_cat") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Bar plot for search_topic
bar_plot6 <- ggplot(data_train, aes(x = private_collections, fill = image_favs_cat)) +
  geom_bar(position = "dodge") +
```

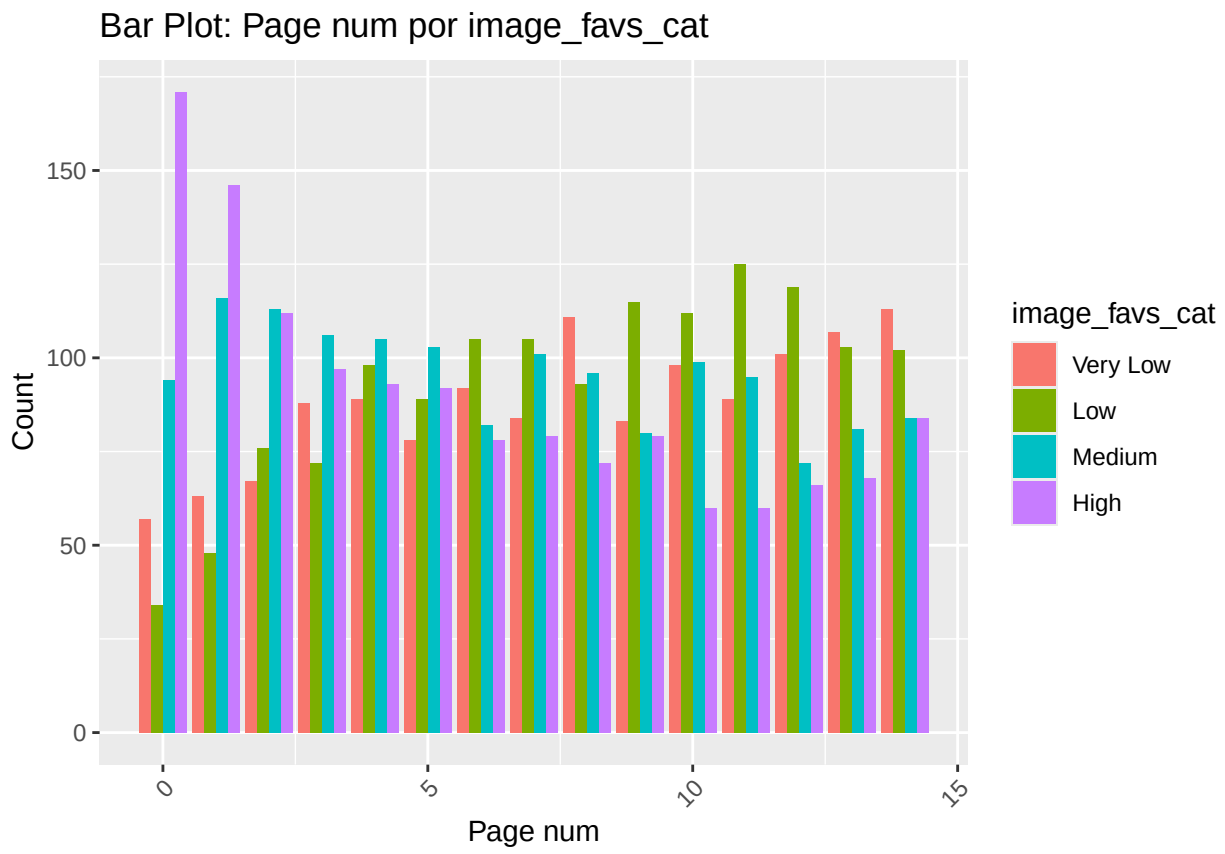
```r
  labs(title = "Bar Plot: Private collections por image_favs_cat",
       x = "private_collections",
       y = "Count",
       fill = "image_favs_cat") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))


# Crear un grid de los bar plots
library(gridExtra)
grid.arrange(bar_plot1, bar_plot2, nrow = 2)
```



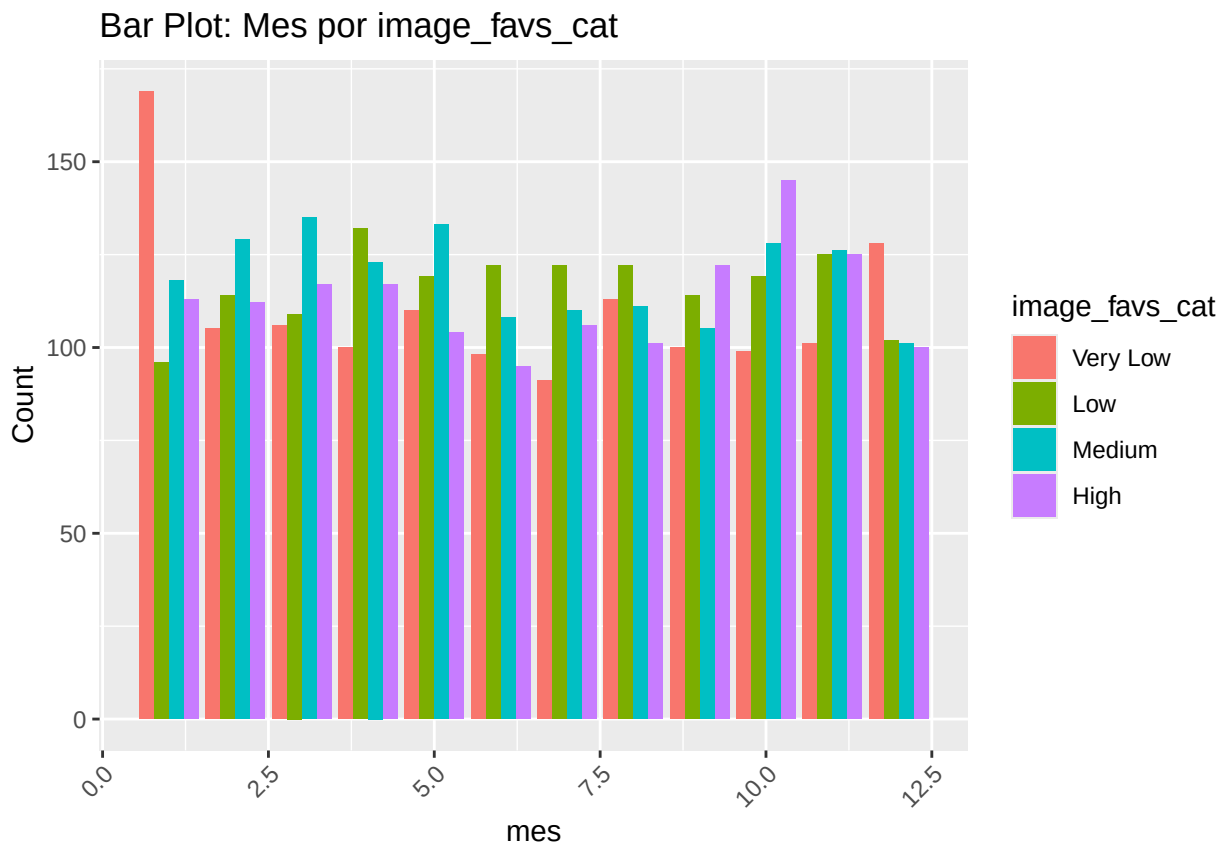

```r
print(bar_plot3)
```



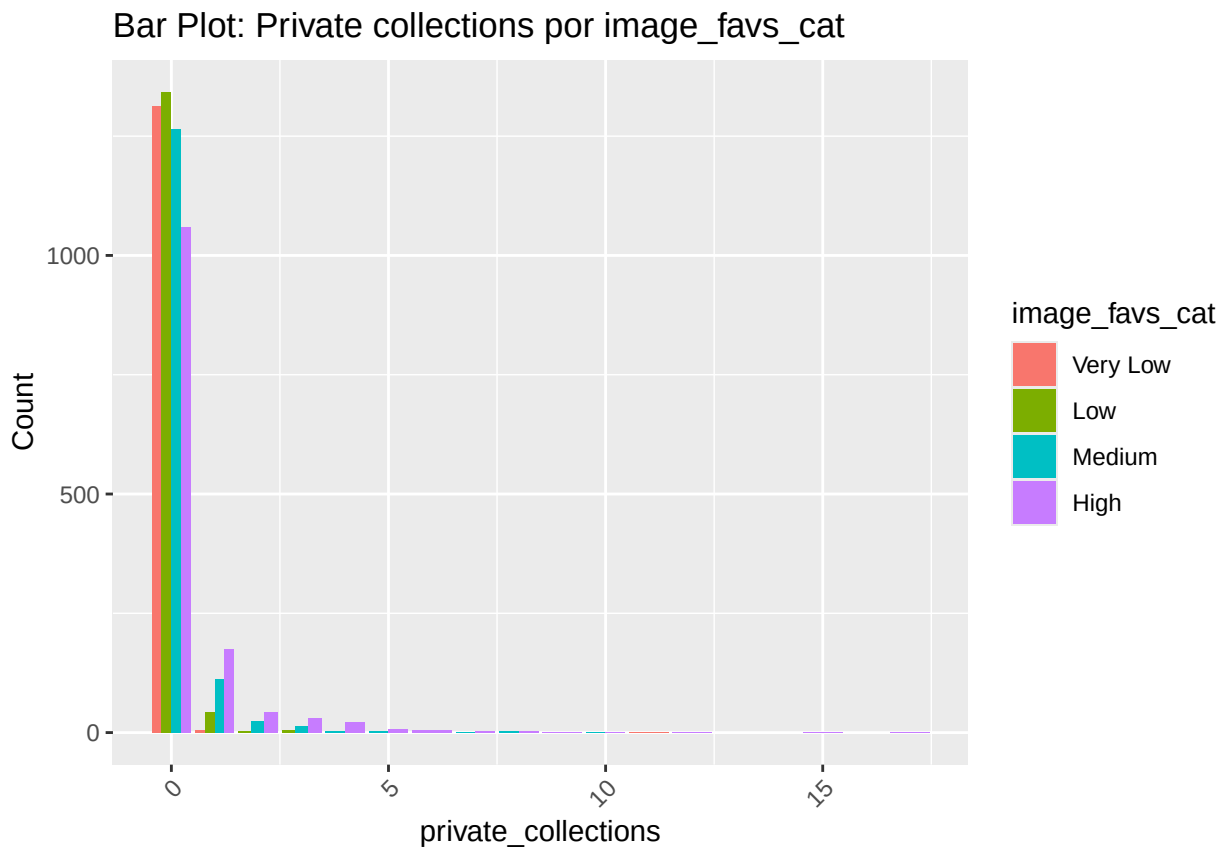

```
print(bar_plot4)
```




```
print(bar_plot5)
```




```
print(bar_plot6)
```



Al analizar los gráficos de barras de las variables categóricas en relación con la variable objetivo (`image_favs_cat`), se observan las siguientes tendencias:

1. **Variables Temporales**:
   - Las variables temporales, como el día de la semana y el momento del día, no parecen diferir significativamente en el número de imágenes con los distintos niveles de `image_favs_cat`, con la excepción de las imágenes publicadas en enero, que tienen más casos en la categoría "Very Low" que en otras.
2. **Search Topic**:
   - El tema de búsqueda parece influir en la popularidad de las imágenes. Por ejemplo, las imágenes relacionadas con "Manga Art", "Character Design Fantasy" y "Gothic Art" son las más apreciadas, ya que tienen más casos con un nivel "High" de favoritos. En contraste, las imágenes relacionadas con "Fan Arts" y "Street Art Graffiti" tienden a tener más valores "Very Low", lo que sugiere que son menos apreciadas. "Science Fiction", "Steampunk Art" y "Pixel Art" tienen más valores "Low".
3. **Private Collections**:
   - Las imágenes que han sido incluidas en al menos una colección privada tienden a tener valores "High" de favoritos, lo que sugiere que la exclusividad o el reconocimiento por parte de otros usuarios puede influir en la popularidad de las imágenes.

En resumen, el tema de búsqueda y la inclusión en colecciones privadas parecen influir en la popularidad de las imágenes, mientras que las variables temporales no muestran diferencias significativas en la distribución de los niveles de favoritos.

-> NOTA: Guardamos aquí el data_parsed como csv limpio?

4. Análisis de los datos

Creación de modelo supervisado, no supervisados y contraste de hipótesis:

Modelización utilizando random forest

Ahora que entendemos un poco qué variables separan mejor nuestros datos de acuerdo al nivel de favoritos recibidos realizaremos un random forest y analizaremos el rendimiento del modelo con los datos no vistos de test.

```
# Modelo supervisado: Random Forest
set.seed(42)
rf_model <- randomForest(image_favs_cat ~ ., data = data_train, importance = TRUE)

# Predicciones
predictions <- predict(rf_model, data_test)

# Evaluar el modelo
conf_matrix <- confusionMatrix(predictions, data_test$image_favs_cat)

# Mostrar resultados
print(conf_matrix)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Very Low Low Medium High
##   Very Low      302  21      0    0
##   Low            28 264     43    0
##   Medium          0  62    256   45
##   High            0   1     57  294
##
## Overall Statistics
##
##                Accuracy : 0.8128
##                  95% CI : (0.7912, 0.8331)
##     No Information Rate : 0.2593
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7503
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Very Low Class: Low Class: Medium Class: High
## Sensitivity                   0.9152     0.7586        0.7191      0.8673
## Specificity                   0.9799     0.9307        0.8948      0.9439
## Pos Pred Value                0.9350     0.7881        0.7052      0.8352
## Neg Pred Value                0.9733     0.9191        0.9010      0.9559
## Prevalence                    0.2403     0.2535        0.2593      0.2469
## Detection Rate                0.2200     0.1923        0.1865      0.2141
```

```
## Detection Prevalence          0.2353     0.2440        0.2644      0.2564
## Balanced Accuracy             0.9475     0.8447        0.8069      0.9056
```

Conclusiones:

- Según el dato correspondiente a la Accuracy, el modelo clasifica correctamente aproximadamente el 82.23% de las imágenes.
- El intervalo de confianza del 95% para la precisión se encuentra entre 80.1 y 84.22%. La tasa de no información, que indica la precisión que se obtendría al clasificar siempre en la clase más frecuente, sería del 25.93%.
- El p-value tan pequeño (2.2e-16) indica que la precisión del modelo es significativamente mayor que la del modelo sin información.
- El índice de Kappa (0.763) indica un buen acuerdo entre las predicciones del modelo y las clases reales.

```r
# Importancia de las variables
importance(rf_model)
```

```
##                         Very Low       Low    Medium       High
## search_topic         38.32774270 52.201009 48.513131  46.745661
## year                 33.09045433 43.885807 39.700389  39.028649
## month                 3.22023614  2.695250  3.743661   8.508784
## day_of_week          -0.02528048  3.370354 11.638528   9.217750
## moment_of_day         0.29648448  5.628556 11.580944   7.282289
## image_license         1.42288426  8.498745  5.079053  15.690725
## superficie_px2       12.59586624 17.080859 23.522245  28.059197
## image_size_kb         7.56198071 11.691849 22.436635  22.617483
## image_com            42.44208635 50.515011 44.298740 100.758743
## image_views         120.60309553 95.848094 91.040204 134.870268
## page_num              0.11000215 12.504472  1.236987  14.059194
## private_collections  17.72616883 14.355605  2.720498  22.647023
##                     MeanDecreaseAccuracy MeanDecreaseGini
## search_topic                   70.663577        482.34712
## year                           62.526556        267.32610
## month                           9.345721        128.67560
## day_of_week                    13.170196        174.22114
## moment_of_day                  13.097702         91.76296
## image_license                  16.533629         26.98699
## superficie_px2                 41.074692        228.05796
## image_size_kb                  34.656466        211.86993
## image_com                      81.367904        828.79965
## image_views                   139.571918       1466.75441
## page_num                       14.734888        153.96780
## private_collections            23.347355         57.95659
```

- El número de vistas de la imagen es la variable más importante, lo que indica que tiene una gran influencia en la clasificación del número de favoritos. Parece lógico que cuanto más vista es una imagen, más personas la podrán marcar como favorita.
- También son importantes para predecir el número de favoritos el número de comentarios, la fecha de publicación de la imagen, el tema de búsqueda y el tamaño de la imagen.
- **Precisión del Modelo**: La precisión del modelo es buena (82.23%), con una alta sensibilidad y especificidad en la mayoría de las clases.

- **Importancia de las Variables**: Las variables `image_views`, `image_com`, `published_date`, `search_topic` y `image_size` son las más influyentes en la clasificación del número de favoritos de las imágenes.

Modelo no supervisado: clustering

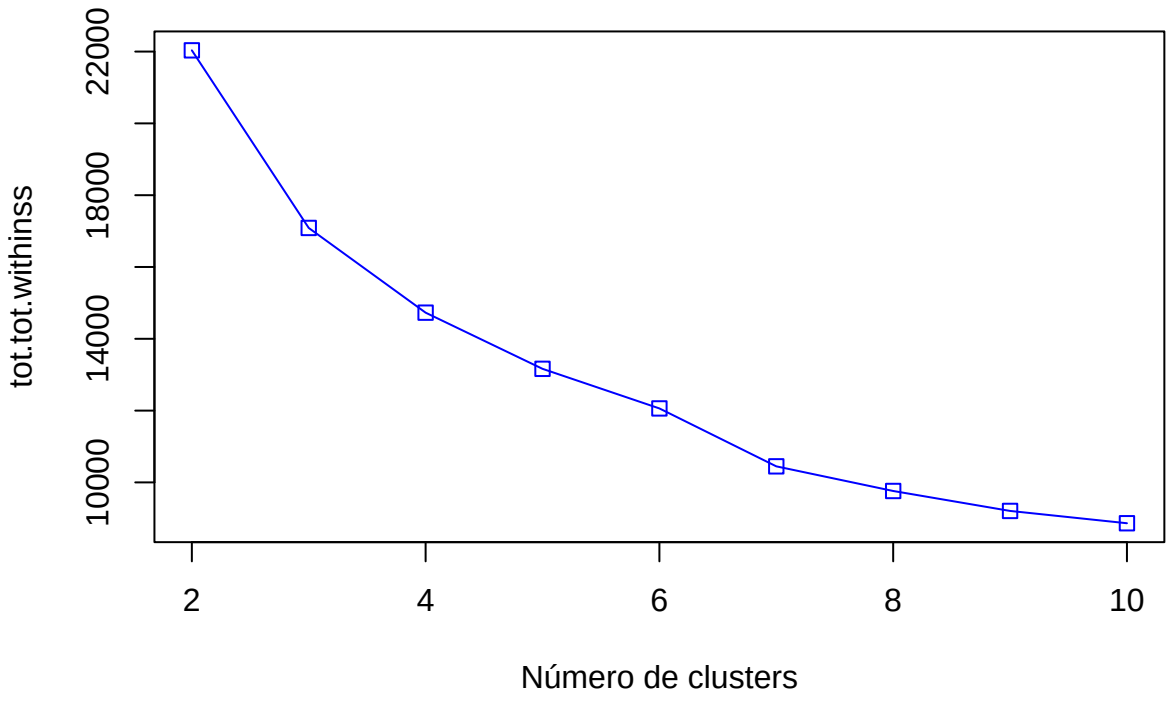
Objetivo del Análisis de Clusterización de Imágenes

El objetivo principal de nuestro análisis de clusterización de imágenes es identificar grupos naturales dentro de nuestra colección de imágenes de DevianArt, agrupándolas según características similares. Estas características incluyen el número de comentarios, vistas, favoritos, tamaño de archivo y superficie de las imágenes. El propósito de este análisis es comprender mejor las diferentes categorías de imágenes presentes en nuestra colección y descubrir relaciones subyacentes entre ellas.

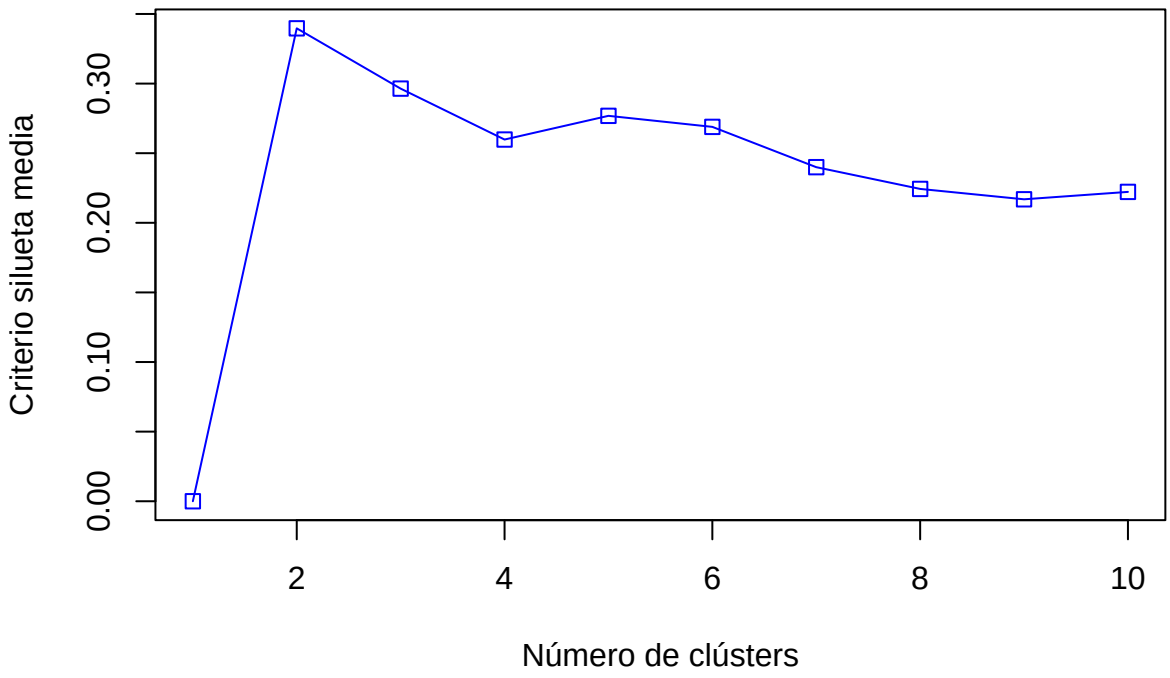
Método de Clusterización

Para lograr este objetivo, aplicamos el algoritmo de k-means a nuestras imágenes seleccionadas, con el fin de agruparlas en un número específico de clústeres. Utilizamos métricas como el método del codo y el criterio de la silueta para determinar el número óptimo de clústeres que mejor representan la estructura subyacente de nuestros datos. Una vez seleccionado el número adecuado de clústeres, asignamos cada imagen a un clúster específico en función de sus características.

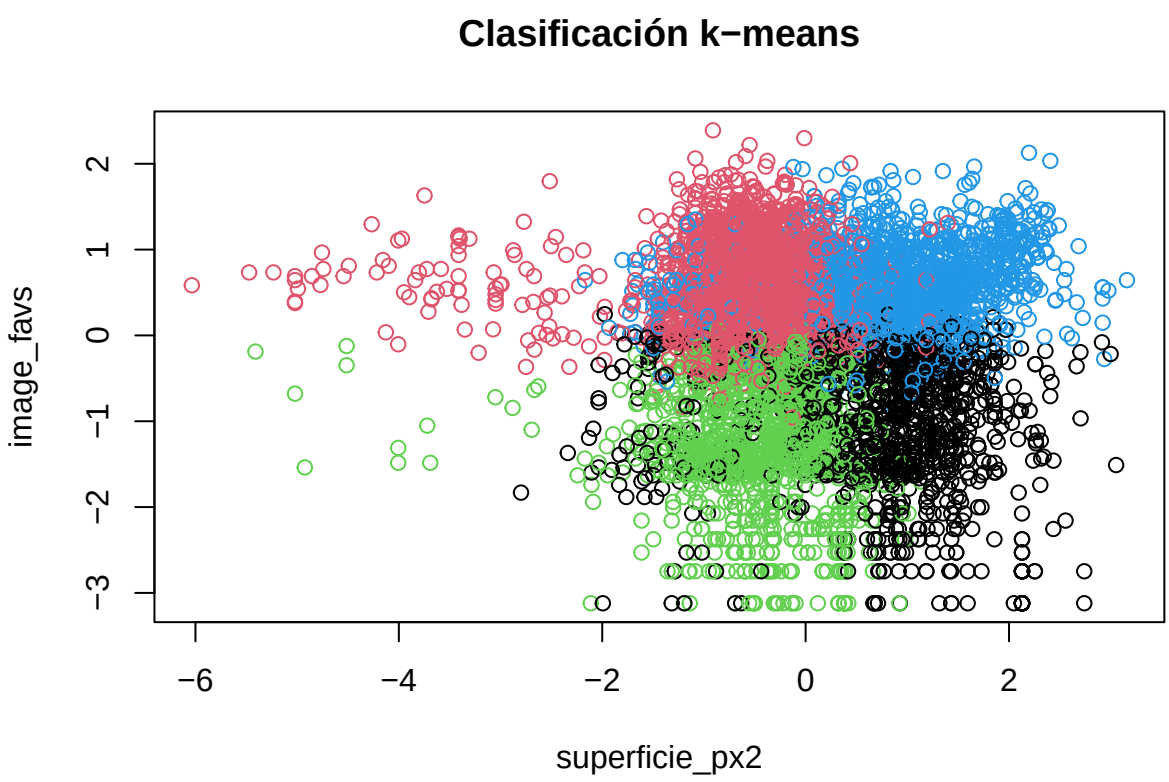
```
# modelo no supervisado usando clustering

data_kmeans <- bake(train, new_data = data_parsed)

# Seleccionamos variables numericas para encontrar grupos de imagenes basadas en sus el numero de favor


selected_vars = c(
                  'image_com', 'image_views',
                  'image_size_kb',"superficie_px2", "image_favs")

data_kmeans <- data_kmeans %>% select(all_of(selected_vars))


# Metodo del Codo
resultados <- rep(0, 10)
for (i in c(2,3,4,5,6,7,8,9,10))
{
  fit           <- kmeans(data_kmeans, i)
  resultados[i] <- fit$tot.withinss
}
plot(2:10,resultados[2:10],type="o",col="blue",pch=0,xlab="Número de clusters",ylab="tot.tot.withinss")
```




```
# Metodo de la silueta
if (!require('fpc')) install.packages('fpc')
library(fpc)
fit_asw <- kmeansruns(data_kmeans, krange = 1:10, criterion = "asw")
plot(1:10,fit_asw$crit,type="o",col="blue",pch=0,xlab="Número de clústers",
     ylab="Criterio silueta media")
```




```r
# Aplicar el algoritmo k-means
set.seed(123)  # Para reproducibilidad
k <- 4 # Número de clústeres
kmeans_res <- kmeans(data_kmeans, centers = k)


# bill_lLength y bill_depth
plot(data_kmeans[c(4,5)], col=kmeans_res$cluster, main="Clasificación k-means")
```



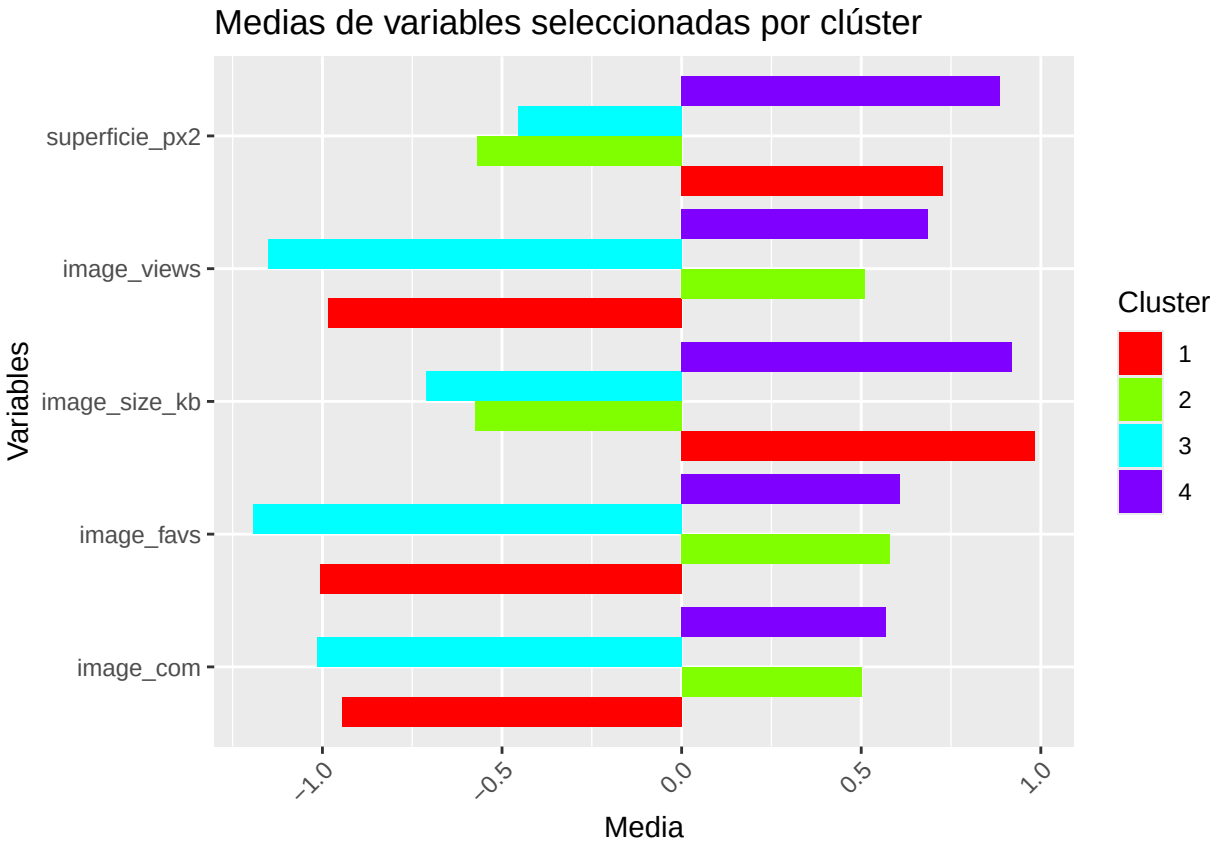
Interpretación de los Resultados

Tras completar el proceso de clusterización, examinamos los diferentes grupos resultantes para identificar patrones y características distintivas en cada uno. Por ejemplo, podemos encontrar clústeres que contienen imágenes altamente populares y ampliamente apreciadas, clústeres con imágenes menos conocidas pero de alta calidad, y clústeres con características específicas, como tamaño de archivo más pequeño o superficie más grande. Esta interpretación nos proporciona una comprensión más profunda de la diversidad de imágenes en nuestra colección y nos ayuda a organizarlas de manera más efectiva para futuros análisis y aplicaciones prácticas.

Hemos decidido analizar 4 clústeres, aunque hemos tenido resultados contrarios en el método de la silueta y del codo. El primero nos indicaba un número de 2 clusters mientras que el método del codo al menos 4 clústeres. Hemos decidido ir por 4 clústeres para una mejor separación de características.

```
# Calcular las medias de las variables seleccionadas por clúster
means_by_cluster <- aggregate(data_kmeans, by = list(kmeans_res$cluster), FUN = mean)

# Convertir el dataframe de medias por cluster en formato largo
means_by_cluster_long <- tidyr::pivot_longer(means_by_cluster, cols = -Group.1, names_to = "Variable", 

# Crear un gráfico de barras con las medias por clúster
ggplot(means_by_cluster_long, aes(x = Variable, y = Media, fill = factor(Group.1))) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Medias de variables seleccionadas por clúster",
       x = "Variables", y = "Media",
       fill = "Cluster") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_fill_manual(values = rainbow(k)) +
```

```
coord_flip()
```



- **Cluster 1**: Este clúster se caracteriza por tener imágenes con valores altos en "superficie_px2" (tamaño de la imagen en píxeles cuadrados) y "image_size_kb" (tamaño de la imagen en kilobytes), lo que sugiere que estas imágenes tienen una alta resolución y ocupan más espacio de almacenamiento. Sin embargo, tienen valores bajos en "image_views" (número de vistas), "image_com" (número de comentarios) y "image_favs" (número de favoritos), lo que indica que estas imágenes no son tan populares o no reciben tanta atención a pesar de su alta calidad visual.
- **Cluster 2**: Este clúster consiste en imágenes que son altamente apreciadas, con muchos comentarios, favoritos y vistas. Aunque estas imágenes pueden tener un tamaño relativamente pequeño en términos de "image_size_kb", son populares y generan mucha interacción. Esto sugiere que son imágenes de alta calidad que capturan la atención de los espectadores sin ocupar demasiado espacio en almacenamiento.
- **Cluster 3**: En este clúster se encuentran imágenes con exposiciones bajas y que ocupan poca memoria. Tienen valores bajos en "image_com" y "image_views", lo que indica que estas imágenes no reciben muchos comentarios ni vistas. Además, tienen valores bajos en "image_size_kb", lo que sugiere que ocupan menos espacio de almacenamiento en comparación con las imágenes de otros clústeres. Esto puede indicar que son imágenes menos populares o menos relevantes en comparación con las del clúster 2.
- **Cluster 4**: Este clúster se destaca por tener valores altos en todas las métricas evaluadas. Las imágenes en este clúster tienen altos números de comentarios ("image_com"), vistas ("image_views"), y favoritos ("image_favs"). Además, ocupan un espacio considerable en almacenamiento, como lo indican los altos valores de "image_size_kb". Aunque la superficie de estas imágenes ("superficie_px2") puede variar, en general, este clúster está compuesto por imágenes que son muy populares y atractivas para los usuarios, generando una gran interacción y ocupando una cantidad significativa de espacio en almacenamiento.

Contraste de Hipótesis

H_0: No hay asociación entre el tema de búsqueda (`search_topic`) y la categoría del número de favoritos (`image_favs_cat`).

H_1: Existe una asociación entre el tema de búsqueda (`search_topic`) y la categoría del número de favoritos (`image_favs_cat`).

```
# Crear una tabla de contingencia
contingency_table <- table(data_parsed$search_topic, data_parsed$image_favs_cat)

# Realizar el test de Chi-cuadrado
chi_square_test <- chisq.test(contingency_table)

# Resumen del test de Chi-cuadrado
print(chi_square_test)
```

```
##
##  Pearson's Chi-squared test
##
## data:  contingency_table
## X-squared = 2855.8, df = 57, p-value < 2.2e-16
```

```
contingency_table
```

```
##
##                                 Very Low Low Medium High
##   Abstract art                        82 112     85   55
##   Anime and manga art                 66  29     45  162
##   Character designs                   13  51    119  160
##   Comics and graphic novels          141  71    104   34
##   Concept art                         26  90    119  119
##   Cosplay photography                 33 126    125   75
##   Creature concepts                   40 106    111   96
##   Cyberpunk art                      117 104     72   51
##   Digital paintings                   85 146     48   58
##   Fan art (for specific fandoms)     288  31     11   11
##   Fantasy art                          5  40    133  178
##   Gothic art                          59  66     70  152
##   Horror art                          20 109    133   93
##   Landscape art                        4  52    145  150
##   Pixel art                           53 149     98   55
##   Science fiction art                 48 130    104   62
##   Steampunk art                       93 102     79   63
##   Street art and graffiti            321  29      3    3
##   Surrealism                          64  89     89   55
##   Traditional drawings                92 112     90   64
```

El resultado del test de Chi-cuadrado indica una asociación altamente significativa entre el tema de búsqueda (search_topic) y la categoría del número de favoritos (image_favs_cat). El valor del estadístico de Chi-cuadrado es muy alto (2855.8), lo que sugiere que las diferencias observadas entre las frecuencias esperadas y observadas en la tabla de contingencia son significativas. Además, el p-valor es extremadamente pequeño ($< 2.2e\text{-}16$), lo que indica que la probabilidad de obtener estos resultados bajo la hipótesis nula (que no hay asociación entre las variables) es prácticamente nula.

Esto sugiere que el tema de búsqueda influye en la popularidad de las imágenes, lo que puede tener implicaciones importantes para la optimización de la búsqueda y la promoción de contenidos en tu plataforma.

5. Representación de los resultados a partir de tablas y gráficas

Este apartado se ha respondido a lo largo de la práctica.

Se debe representar tanto el contenido del dataset para observar las proporciones y distribuciones de las diferentes variables una vez aplicada la etapa de limpieza, como los resultados obtenidos tras la etapa de análisis.

6. Resolución del problema

El objetivo de este ejercicio era responder a las siguientes preguntas, que se planteaban al inicio:

1. ¿Qué características de las imágenes (como tamaño, resolución y número de comentarios) están más correlacionadas con el número de favoritos?

El análisis de la importancia de las variables realizado mediante el modelo de Random Forest indica que las características más influyentes en la predicción del número de favoritos de las imágenes son el número de vistas (image_views), el número de comentarios (image_com), y el tamaño de la imagen en píxeles (superficie_px2).

2. ¿Cómo influyen las variables temporales (como el día de la semana y el momento del día en que se publica una imagen) en la cantidad de favoritos?

Tanto el día de la semana como el momento del día en que se publica la imagen no son relevantes a la hora de determinar el número de favs que obtendrá una imagen.

3. ¿Qué temas de búsqueda son más propensos a recibir un alto número de favoritos?

A partir del contraste de hipótesis, hemos visto que el tema de búsqueda (variable search_topic) influye en la popularidad de las imágenes.

Los temas más propensos a obtener un número significativo de favs (nivel Medium o High en la variable `image_favs_cat`) son:

- Fantasy art: 178 (High), 133 (Medium). 87.36% del total.
- Landscape art: 150 (High), 145 (Medium). 84.05 del total.
- Character designs: 160 (High), 119 (Medium). 81.34 del total.
- Concept art: 119 (High), 119 (Medium). 67.23 del total.
- Horror art: 93 (High), 133 (Medium). 63.66 del total.

4. ¿Existen patrones específicos en la comunidad de DeviantArt que puedan predecir la popularidad de una imagen?

Mediante el análisis de clústeres, hemos identificado cuatro grupos distintos de imágenes basados en sus características comunes. Por ejemplo, el clúster 1 se caracteriza por imágenes con una alta resolución (superficie_px2) y tamaño de imagen (image_size_kb), pero con baja exposición en términos de vistas, comentarios y favoritos. En contraste, el clúster 2 presenta imágenes con un alto número de comentarios, vistas y favoritos, pero con un tamaño de imagen más moderado.

Examinando la relación entre las características importantes identificadas por el modelo de Random Forest y los perfiles de los clústeres, encontramos que ciertos clústeres están asociados con características destacadas por el modelo. Por ejemplo, el clúster 2, que comprende imágenes altamente populares en términos de

comentarios, vistas y favoritos, también muestra una importancia significativa de estas características en el modelo de Random Forest.

A partir de estos análisis, podemos concluir que los resultados obtenidos permiten abordar satisfactoriamente el problema planteado de entender qué características influyen en la popularidad de las imágenes en DeviantArt. Los clústeres identificados nos ayudan a segmentar y comprender mejor la diversidad de imágenes en la plataforma, mientras que las características importantes del modelo nos proporcionan información sobre los factores que contribuyen significativamente a la popularidad de estas imágenes.