ACTIVIDAD LABORATORIO NO.1
MINERÍA DE TEXTO Y PROCESAMIENTO DE LENGUAJE NATURAL

PRESENTADO POR:
ALEJANDRO DE MENDOZA

PRESENTADO AL PROFESOR:
ING ROGERIO ORLANDO BELTRAN CASTRO

FUNDACIÓN UNIVERSITARIA INTERNACIONAL DE LA RIOJA
BOGOTÁ D.C.
16 DE FEBRERO
2026

TABLA DE CONTENIDO

TABLA DE CONTENIDO **2**
INTRODUCCIÓN **3**
DESARROLLO ACTIVIDAD **3**
1. Obtención Del Corpus De Los Cuatro Libros Seleccionados 4
    Código Implementado 4
2. Quince términos más frecuentes y nube de palabras por libro 8
    Código Implementado 8
3. Cálculo del índice TF-IDF de los cuatro libros 17
    Código Implementado 17
4. Representación gráfica de los quince términos más característicos 18
    15 términos con mayor valor de TF-IDF en cada libro 18
5. Comparación entre términos frecuentes y característicos 22
    Comparación respectiva de los dos gráficos: 22
6. Asociaciones entre palabras (bigramas) 24
    Análisis de asociaciones mediante bigramas 24
    Asociaciones con “helsing” 25
    Asociaciones con “martians” 26
7. Verificación del entorno de trabajo en RStudio 28
    Imagen de la figura: 28
    Interpretación detallada de los objetos generados: 28
CONCLUSIONES DE LA ACTIVIDAD **29**
BIBLIOGRAFÍA **30**
AGRADECIMIENTO **31**

INTRODUCCIÓN

En el contexto actual del análisis de datos y la inteligencia artificial, la minería de texto y el procesamiento del lenguaje natural (PLN) se han convertido en áreas fundamentales dentro de la informática. Estas disciplinas permiten extraer información relevante a partir de grandes volúmenes de texto, facilitando tareas como la clasificación de documentos, el análisis semántico, la identificación de patrones lingüísticos y la caracterización temática de diferentes fuentes textuales.

El presente trabajo desarrolla un laboratorio práctico orientado al análisis de texto utilizando el lenguaje de programación R, junto con librerías especializadas como tidytext, gutenbergr, dplyr y ggplot2. Estas herramientas permiten aplicar técnicas modernas de procesamiento lingüístico y minería de datos de manera eficiente, así como visualizar los resultados mediante representaciones gráficas.

En esta actividad se abordó el procesamiento de un corpus compuesto por cuatro libros pertenecientes al Proyecto Gutenberg, una biblioteca digital de dominio público que ofrece acceso gratuito a miles de obras literarias. En particular, se seleccionaron las siguientes novelas clásicas:

- Dracula (Bram Stoker)
- Frankenstein (Mary Shelley)
- The Time Machine (H. G. Wells)
- The War of the Worlds (H. G. Wells)

A partir de estas obras, se realizó un análisis exploratorio basado en la frecuencia de términos y en la identificación de palabras representativas mediante el índice TF-IDF (Term Frequency - Inverse Document Frequency), el cual permite destacar aquellos términos característicos de cada libro en comparación con los demás.

El laboratorio incluyó, en primer lugar, la obtención y construcción del corpus textual, seguido por la extracción de los quince términos más frecuentes y su respectiva representación gráfica. Posteriormente, se calculó el índice TF-IDF para identificar los términos más distintivos de cada obra, lo cual permitió comparar las diferencias entre palabras comunes y palabras temáticamente representativas.

Finalmente, se realizó un análisis de asociaciones entre palabras mediante bigramas, obteniendo relaciones léxicas relevantes para términos característicos seleccionados, lo que contribuye a comprender cómo ciertos conceptos aparecen vinculados dentro del discurso narrativo de cada libro.

El objetivo principal de este laboratorio es aplicar de manera práctica los fundamentos de la minería de texto y el procesamiento del lenguaje natural, desarrollando habilidades para el análisis estadístico de documentos, la extracción de características relevantes y la interpretación de resultados a partir de herramientas computacionales modernas. En conclusión, este trabajo demuestra la utilidad del PLN como un recurso esencial para transformar texto no estructurado en información significativa dentro del ámbito académico y profesional.

DESARROLLO ACTIVIDAD

A continuación, se presenta el desarrollo paso a paso del laboratorio de Minería de Texto y Procesamiento de Lenguaje Natural utilizando el lenguaje de programación R y librerías especializadas del ecosistema tidyverse. El proceso fue abordado de manera estructurada, construyendo el análisis progresivamente desde la obtención del corpus hasta la interpretación de los resultados. En las siguientes secciones se documentan las decisiones metodológicas adoptadas, el procesamiento aplicado a los textos y los resultados obtenidos en cada etapa del análisis.

En primer lugar, se procedió a la obtención del corpus textual mediante el paquete gutenbergr, el cual permitió descargar directamente desde el Proyecto Gutenberg las cuatro obras seleccionadas. Posteriormente, los textos fueron unificados en un único conjunto de datos y sometidos a un proceso de limpieza y transformación, que incluyó la tokenización (separación del texto en palabras individuales) y la eliminación de stopwords o palabras vacías que no aportan significado relevante al análisis.

Una vez construido el corpus limpio, se realizó el cálculo de las frecuencias de términos para cada libro, identificando los quince términos más frecuentes en cada obra y representándolos gráficamente. Este paso permitió obtener una primera aproximación cuantitativa al contenido de los textos y observar patrones léxicos recurrentes dentro de cada narrativa.

Posteriormente, se implementó el cálculo del índice TF- IDF (Term Frequency - Inverse Document Frequency) con el fin de identificar los términos más característicos de cada libro. A diferencia de la frecuencia simple, esta métrica permite destacar palabras que no solo son frecuentes en un documento, sino que además resultan distintivas frente a los demás textos analizados.

Finalmente, se llevó a cabo un análisis de asociaciones entre palabras mediante la construcción de bigramas (pares consecutivos de términos). Este procedimiento permitió identificar relaciones léxicas relevantes para palabras representativas seleccionadas, evidenciando patrones de co-ocurrencia dentro de las obras literarias.

Cada etapa del laboratorio fue validada mediante la revisión de los resultados obtenidos en consola y la representación gráfica correspondiente, garantizando la correcta ejecución del procesamiento y facilitando la interpretación de los hallazgos. Este enfoque progresivo permitió consolidar la comprensión práctica de las técnicas fundamentales de minería de texto aplicadas a un corpus literario real.

1. Obtención Del Corpus De Los Cuatro Libros Seleccionados

En primer lugar, se procedió a obtener el corpus textual correspondiente a cuatro libros seleccionados del Proyecto Gutenberg. Para ello, se utilizó el paquete gutenbergr, el cual permite descargar directamente textos en formato plano desde dicha biblioteca digital.

Las obras seleccionadas para el análisis fueron:

- Dracula - Bram Stoker
- Frankenstein - Mary Shelley
- The Time Machine - H. G. Wells
- The War of the Worlds - H. G. Wells

Posteriormente, los textos fueron integrados en un único corpus para facilitar su procesamiento conjunt

Código Implementado

A continuación, la explicación de todo el código implementado en RStudio para este punto:

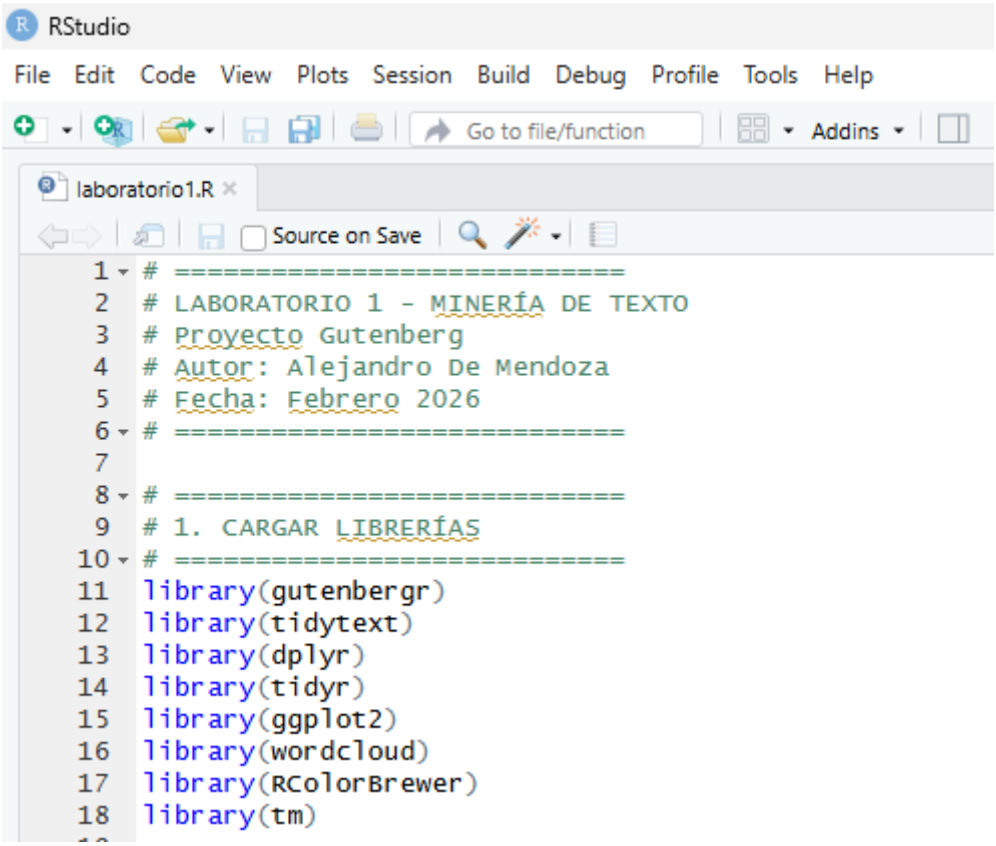
Importación de librerías:

Lo primero es importar las librerías donde la función library() permite cargar en memoria los paquetes previamente instalados en R. Estos paquetes contienen funciones especializadas que serán utilizadas durante el análisis.

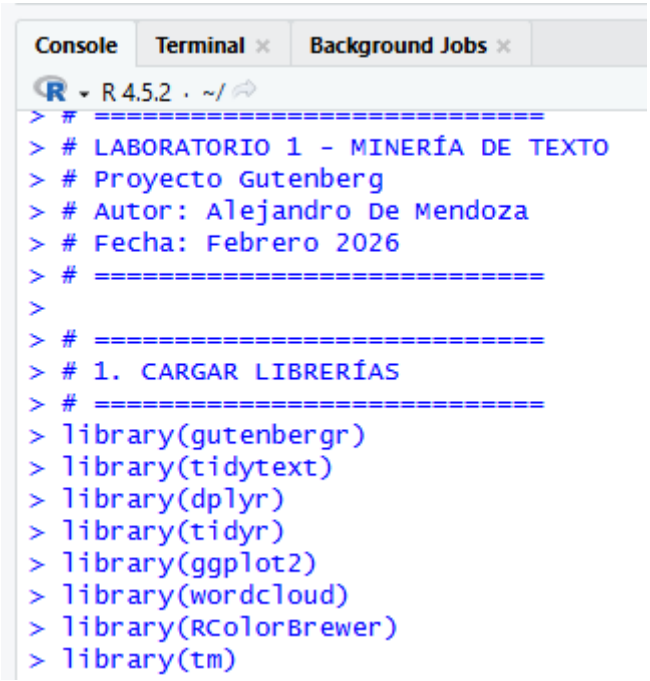
Cada librería cumple un propósito específico:

- gutenbergr → Permite descargar libros desde el Proyecto Gutenberg.
- tidytext → Proporciona herramientas para minería de texto y tokenización.
- dplyr → Facilita la manipulación y transformación de datos.
- tidyr → Permite organizar y transformar estructuras de datos (por ejemplo, separar columnas).
- ggplot2 → Se utiliza para generar gráficos estadísticos.
- wordcloud → Permite crear nubes de palabras.
- RColorBrewer → Mejora de colores.
- tm → Proporciona funciones adicionales para procesamiento de texto.

A continuación, la imagen del código:



Y la resolución en consola mostro el siguiente resultado:



Libros a descargar:

Luego de la importación de las librerías se procedió a descargar cada uno de los libros con el siguiente código, entonces este bloque descarga los textos completos de los libros seleccionados desde el Proyecto Gutenberg utilizando la función gutenberg_download() del paquete gutenbergr.

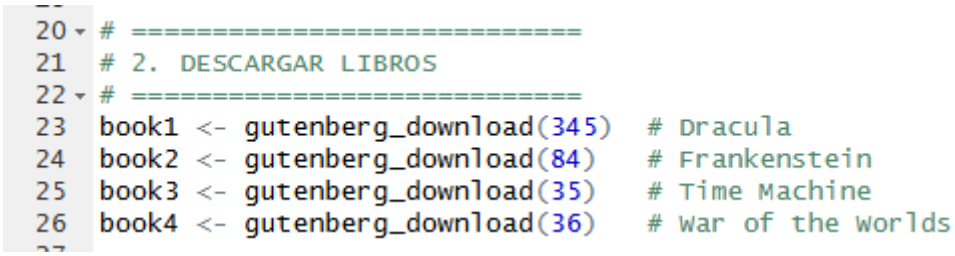
Cada número que aparece dentro del paréntesis corresponde al ID único del libro en Gutenberg. Por ejemplo:

- 345 → Dracula
- 84 → Frankenstein
- 35 → The Time Machine
- 36 → The War of the Worlds

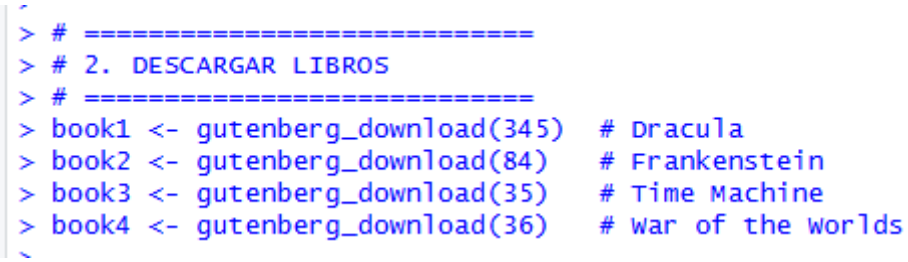
La función realiza lo siguiente:

- Se conecta a la biblioteca digital de Gutenberg.
- Descarga el texto en formato plano.
- Lo guarda en un objeto tipo tabla (data frame).
- Ese objeto se almacena en una variable (book1, book2, etc.).

A continuación, la imagen del código:



Y la resolución en consola mostró el siguiente resultado:



NOTA: Ahora, si por alguna razón en la consola aparece:

- Mirror list unavailable. Falling back to <https://aleph.pglaf.org>.

Esto no es un error. Significa que el servidor principal no respondió y el sistema utilizó automáticamente un servidor alternativo (mirror) para realizar la descarga. El libro se descargó correctamente.

Asignación de nombres:

Una ves descargos los libros de procedió a asignar una etiqueta o identificador a cada libro descargado, entonces cada objeto (book1, book2, book3, book4) contiene el texto del libro, pero inicialmente solo tiene columnas como:

- gutenberg_id
- text

Ahora para poder distinguir a qué obra pertenece cada línea de texto, se crea una nueva columna llamada book. Por ejemplo, book1$book <- "Dracula" que agrega una columna llamada book dentro de book1 y coloca el valor Dracula en todas sus filas". Lo mismo ocurre con los otros libros. A continuación, la imagen del código:

```
# ============================
# 3. ASIGNAR NOMBRES
# ============================
book1$book <- "Dracula"
book2$book <- "Frankenstein"
book3$book <- "Time Machine"
book4$book <- "War of the Worlds"
```

Y la resolución en consola mostró el siguiente resultado:

```
> # ============================
> # 3. ASIGNAR NOMBRES
> # ============================
> book1$book <- "Dracula"
> book2$book <- "Frankenstein"
> book3$book <- "Time Machine"
> book4$book <- "War of the Worlds"
```

Unión de los Corpus:

Este código une los cuatro libros descargados en un solo conjunto de datos (data frame).

Cada objeto (book1, book2, book3, book4) contiene el texto de un libro en forma de tabla.
La función bind_rows() del paquete dplyr:

- Toma varias tablas que tienen la misma estructura
- Las combina verticalmente
- Apila las filas una debajo de la otra

Como resultado:

- Se crea un nuevo objeto llamado corpus
- Este contiene el texto completo de los cuatro libros
- Cada fila mantiene la información del libro al que pertenece (gracias a la columna book que agregaste antes)

A continuación, la imagen del código:

```
# ============================
# 4. UNIR CORPUS
# ============================
corpus <- bind_rows(book1, book2, book3, book4)
cat("Corpus creado con", nrow(corpus), "líneas de texto\n")
```

Y la resolución en consola mostró el siguiente resultado:

```
> # ============================
> # 4. UNIR CORPUS
> # ============================
> corpus <- bind_rows(book1, book2, book3, book4)
> cat("Corpus creado con", nrow(corpus), "líneas de texto\n")
Corpus creado con 32370 líneas de texto
```

Por lo que de esta manera se obtuvo el corpus de los respectivos libros mencionados con sus respectivos nombres y su unificación en uno solo.

2. Quince términos más frecuentes y nube de palabras por libro

Una vez obtenido el corpus, se realizó el proceso de tokenización, es decir, la separación del texto en palabras individuales. Posteriormente, se eliminaron las palabras vacías o stopwords, las cuales no aportan significado relevante.

Con el texto limpio, se calcularon los quince términos más frecuentes en cada libro, lo cual permitió identificar las palabras con mayor repetición dentro de cada obra.

Código Implementado

A continuación, la explicación de todo el código implementado en RStudio para este punto:

Tokenización:

Este bloque toma el objeto corpus, que contiene el texto completo de los libros, y lo transforma en una estructura adecuada para el análisis lingüístico.

En particular:

- Cada fila del corpus contiene una línea de texto.
- La función unnest_tokens() separa ese texto en unidades más pequeñas llamadas tokens.
- En este caso, los tokens son palabras individuales.

A continuación, la imagen del código:

```
42 # ============================
43 # 5. TOKENIZAR
44 # ============================
45 words <- corpus %>%
46   unnest_tokens(word, text)
47 cat("Total de palabras tokenizadas:", nrow(words), "\n")
```

Y la resolución en consola mostró el siguiente resultado:

```
> # ============================
> # 5. TOKENIZAR
> # ============================
> words <- corpus %>%
+   unnest_tokens(word, text)
> cat("Total de palabras tokenizadas:", nrow(words), "\n")
Total de palabras tokenizadas: 331182
```

Eliminar StopWords

Ahora se procede a eliminar las stopwords del conjunto de palabras tokenizadas. Es por esto que las stopwords son palabras muy frecuentes en un idioma que no aportan significado relevante para el análisis, como por ejemplo:

- the
- and
- of
- in
- to

Y en este caso se utiliza data("stop_words") para cargar en memoria el conjunto de palabras vacías incluido en el paquete tidytext. Este dataset contiene una lista de stopwords en inglés. Adicionalmente utilizó anti_join(stop_words) donde la función anti_join() elimina de words todas las filas que coincidan

con la columna word en stop_words. Finalmente, el resultado se guarda en un nuevo objeto llamado words_clean.

A continuación, la imagen del código:

```
# ============================
# 6. ELIMINAR STOPWORDS
# ============================
data("stop_words")
words_clean <- words %>%
  anti_join(stop_words)
cat("Palabras después de eliminar stopwords:", nrow(words_clean), "\n")
```

Y la resolución en consola mostró el siguiente resultado:

```
> # ============================
> # 6. ELIMINAR STOPWORDS
> # ============================
> data("stop_words")
> words_clean <- words %>%
+   anti_join(stop_words)
Joining with `by = join_by(word)`
> cat("Palabras después de eliminar stopwords:", nrow(words_clean), "\n")
Palabras después de eliminar stopwords: 110323
```

Determinación de las 15 palabras frecuentes:

Este bloque calcula las 15 palabras más frecuentes en cada libro después de haber eliminado las stopwords.

Explicación paso a paso:

- words_clean %>%: Se parte del conjunto de palabras ya limpio (sin stopwords).
- count(book, word, sort = TRUE): La función count():
  - Cuenta cuántas veces aparece cada palabra (word)
  - Dentro de cada libro (book)
  - Crea una nueva columna llamada n con la frecuencia
  - sort = TRUE ordena los resultados de mayor a menor frecuencia
  - En este punto ya se sabe cuántas veces aparece cada palabra en cada libro.
- group_by(book): Agrupa los datos por libro. Esto es importante porque queremos las 15 palabras más frecuentes por cada libro, no en todo el corpus combinado.
- slice_max(n, n = 15): Selecciona las 15 palabras con mayor frecuencia (n) dentro de cada grupo (cada libro).

Y como resultado se crea el objeto top_words, que contiene:

- El nombre del libro
- La palabra
- El número de veces que aparece

Es decir, las 15 palabras más repetidas en cada obra. A continuación, la imagen del código:

```r
# ============================
# 7. TOP 15 PALABRAS FRECUENTES
# ============================
top_words <- words_clean %>%
  count(book, word, sort = TRUE) %>%
  group_by(book) %>%
  slice_max(n, n = 15)

print(top_words)
```

Y la resolución en consola mostró el siguiente resultado:

```
> # ============================
> # 7. TOP 15 PALABRAS FRECUENTES
> # ============================
> top_words <- words_clean %>%
+   count(book, word, sort = TRUE) %>%
+   group_by(book) %>%
+   slice_max(n, n = 15)
>
> print(top_words)
# A tibble: 60 x 3
# Groups:   book [4]
   book    word        n
   <chr>   <chr>   <int>
 1 Dracula time      390
 2 Dracula van       323
 3 Dracula night     310
 4 Dracula helsing   301
 5 Dracula dear      224
 6 Dracula lucy      223
 7 Dracula day       221
 8 Dracula hand      210
 9 Dracula mina      210
10 Dracula door      200
# i 50 more rows
# i Use `print(n = ...)` to see more rows
```

Gráfico de Términos Frecuentes:

Ahora se procede a generar el bloque que genera un gráfico de barras que representa las 15 palabras más frecuentes en cada uno de los cuatro libros analizados.

El objetivo es visualizar de forma clara cuáles son los términos que aparecen con mayor repetición en cada obra.

Explicación paso a paso:

- top_words %>%: Se utiliza el dataset top_words, que ya contiene las 15 palabras más frecuentes por libro.
- ggplot(aes(...)): Se inicia la construcción del gráfico con ggplot2. Dentro de aes() se definen las variables:
  - x = reorder(word, n): Ordena las palabras según su frecuencia para que aparezcan organizadas en el gráfico.
  - y = n: Representa la frecuencia de aparición de cada palabra.
  - fill = book: Asigna un color diferente según el libro.

- geom_col(show.legend = FALSE): Dibuja las barras del gráfico utilizando las frecuencias. Se desactiva la leyenda porque cada libro se mostrará por separado en paneles.
- coord_flip(): Invierte los ejes del gráfico, de manera que:
    - Las palabras queden en el eje vertical
    - Las barras sean horizontales
    - Esto mejora la legibilidad cuando hay muchas etiquetas.
- facet_wrap(~book, scales = "free"): Divide el gráfico en cuatro paneles, uno por cada libro. Esto permite comparar visualmente las palabras frecuentes en cada obra.
- labs(...): Se agregan etiquetas informativas:
    - título del gráfico
    - nombre del eje X (Palabra)
    - nombre del eje Y (Frecuencia)

Entonces este bloque construye un gráfico de barras facetado que muestra los quince términos más frecuentes de cada libro, proporcionando una primera aproximación exploratoria al contenido textual de las obras analizadas.

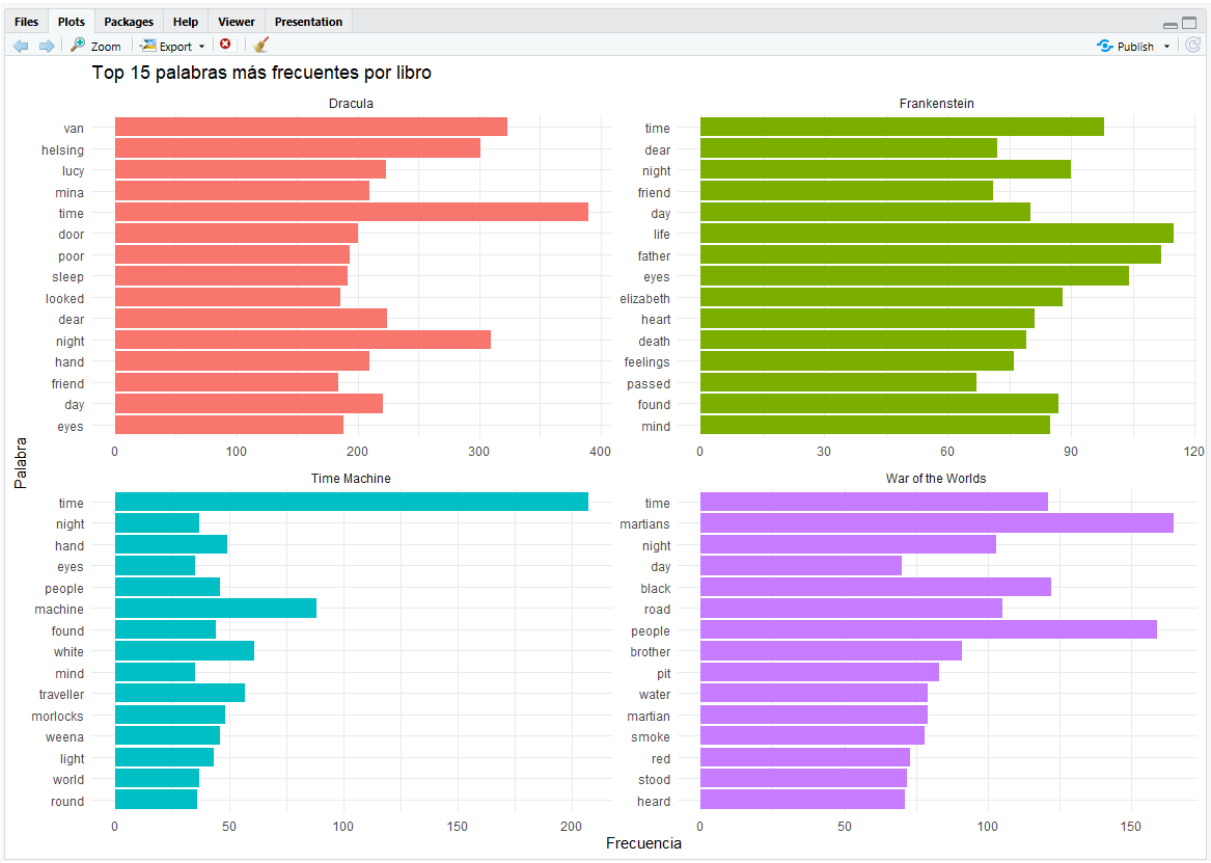
A continuación, la imagen del código:

```
67  # ============================
68  # 8. GRÁFICO DE FRECUENCIAS
69  # ============================
70  top_words %>%
71    ggplot(aes(x = reorder(word, n), y = n, fill = book)) +
72    geom_col(show.legend = FALSE) +
73    coord_flip() +
74    facet_wrap(~book, scales = "free") +
75    labs(
76      title = "Top 15 palabras más frecuentes por libro",
77      x = "Palabra",
78      y = "Frecuencia"
79    ) +
80    theme_minimal()
```

Y la resolución en consola mostró el siguiente resultado:

```
> # ============================
> # 8. GRÁFICO DE FRECUENCIAS
> # ============================
> top_words %>%
+   ggplot(aes(x = reorder(word, n), y = n, fill = book)) +
+   geom_col(show.legend = FALSE) +
+   coord_flip() +
+   facet_wrap(~book, scales = "free") +
+   labs(
+     title = "Top 15 palabras más frecuentes por libro",
+     x = "Palabra",
+     y = "Frecuencia"
+   ) +
+   theme_minimal()
```

Junto con el siguiente grafico:



Y como se puede denotar la figura presentada muestra la representación gráfica de los quince términos más frecuentes en cada una de las cuatro obras analizadas: Dracula, Frankenstein, The Time Machine y The War of the Worlds.

Este gráfico fue construido a partir del conteo de palabras luego del proceso de tokenización y eliminación de stopwords. Cada panel corresponde a un libro distinto y permite observar de manera clara cuáles son los términos con mayor repetición dentro de cada texto.

En Dracula, destacan términos como:

- van
- helsing
- lucy
- mina

Esto se debe a que la narrativa contiene frecuentes referencias a personajes principales, especialmente “Van Helsing”, así como a otros protagonistas relevantes de la historia.

Por otra parte, en Frankenstein aparecen palabras como:

- father
- life
- eyes
- death

Lo cual refleja el enfoque filosófico y emocional del texto, centrado en temas como la vida, la muerte y las relaciones humanas.

Ahora, en The Time Machine, los términos más frecuentes incluyen:

- time
- machine
- traveller
- morlocks

Estos términos están directamente relacionados con la temática central de viajes en el tiempo y las criaturas descritas en la obra.

Finalmente, en The War of the Worlds destacan palabras como:

- martians
- martian
- black
- road

Lo cual evidencia la presencia constante del conflicto extraterrestre y la descripción de escenarios asociados a la invasión.

Nubes de Palabras

Una vez obtenidas las frecuencias de términos, se procedió a generar nubes de palabras para cada uno de los cuatro libros analizados. Las nubes de palabras constituyen una representación visual alternativa donde el tamaño de cada término es proporcional a su frecuencia de aparición dentro del texto.

Código Implementado:

Este bloque genera nubes de palabras para visualizar de forma alternativa la frecuencia de términos en cada libro. A diferencia de los gráficos de barras, las nubes permiten una percepción visual más intuitiva del peso relativo de cada palabra.

Explicación paso a paso:

- par(mfrow = c(2, 2), mar = c(1, 1, 3, 1)): Configura la ventana gráfica para mostrar 4 nubes simultáneamente en formato 2x2 (2 filas, 2 columnas).
- libros <- c("Dracula", "Frankenstein", "Time Machine", "War of the Worlds"): Define el vector con los nombres de los cuatro libros a procesar.
- colores <- list(...): Crea una lista con paletas de colores diferentes para cada libro, utilizando brewer.pal() del paquete RColorBrewer:
    - Set1 para Dracula
    - Set2 para Frankenstein
    - Paired para Time Machine
    - Dark2 para War of the Worlds
- for(i in 1:length(libros)): Bucle que itera sobre cada libro.
- palabras_libro <- words_clean %>% filter(book == libros[i]) %>% count(word, sort = TRUE): Filtra las palabras del libro actual y las cuenta, ordenándolas por frecuencia.
- wordcloud(words, freq, ...): Genera la nube de palabras con los siguientes parámetros:
    - max.words = 100: Muestra hasta 100 palabras
    - min.freq = 5: Solo incluye palabras que aparecen al menos 5 veces
    - random.order = FALSE: Ordena las palabras por frecuencia (no aleatorio)
    - colors = colores[[i]]: Usa la paleta de colores correspondiente
    - scale = c(3, 0.5): Define el rango de tamaños de las palabras
- title(main = libros[i], cex.main = 1.5, font.main = 2): Añade el título de cada nube.
- par(mfrow = c(1, 1)): Restaura la configuración gráfica predeterminada.

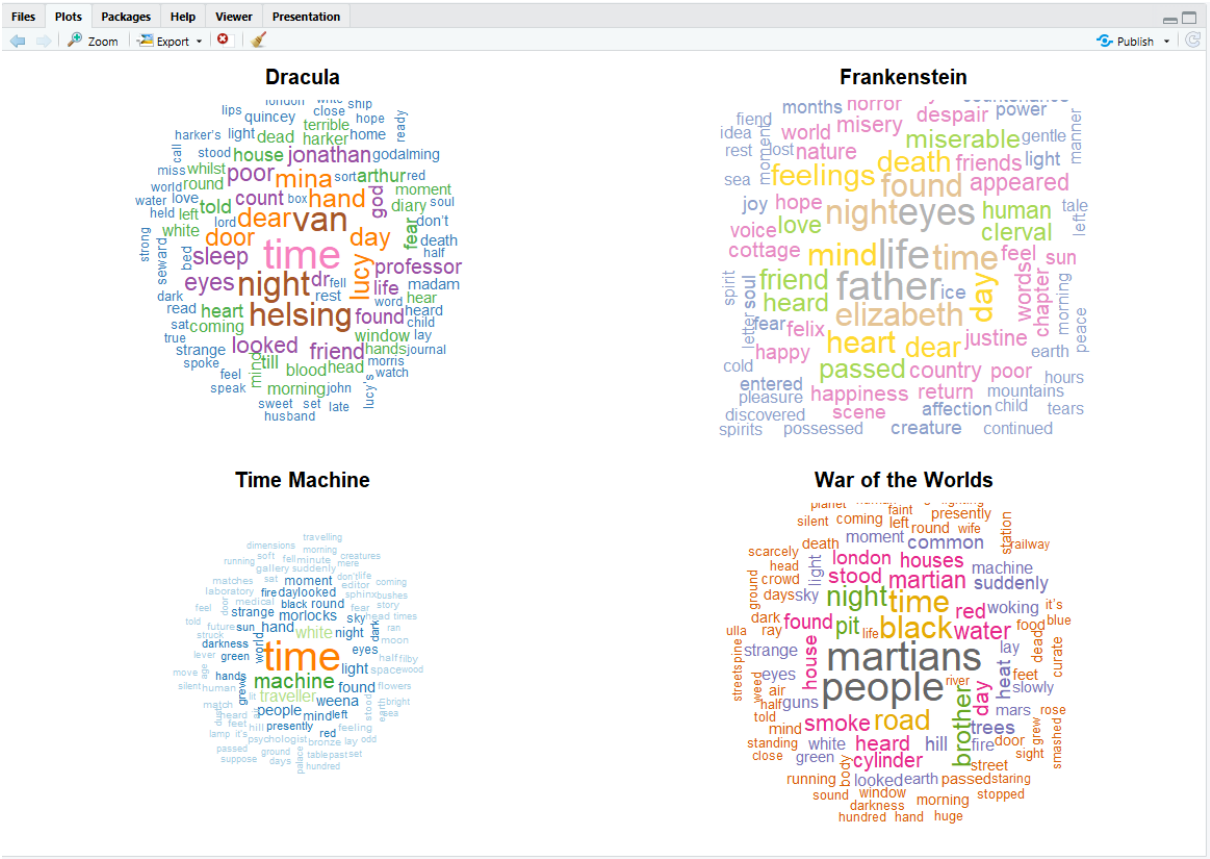
A continuación, la imagen del código:

```r
# ============================
# 9. NUBES DE PALABRAS
# ============================

cat("\n=== GENERANDO NUBES DE PALABRAS ===\n")

# Opción A: 4 nubes en una sola ventana (2x2)
par(mfrow = c(2, 2), mar = c(1, 1, 3, 1))

libros <- c("Dracula", "Frankenstein", "Time Machine", "War of the Worlds")
colores <- list(
  brewer.pal(8, "Set1"),
  brewer.pal(8, "Set2"),
  brewer.pal(8, "Paired"),
  brewer.pal(8, "Dark2")
)

for(i in 1:length(libros)) {
  palabras_libro <- words_clean %>%
    filter(book == libros[i]) %>%
    count(word, sort = TRUE)

  wordcloud(
    words = palabras_libro$word,
    freq = palabras_libro$n,
    max.words = 100,
    min.freq = 5,
    random.order = FALSE,
    colors = colores[[i]],
    scale = c(3, 0.5)
  )

  title(main = libros[i], cex.main = 1.5, font.main = 2)
}

par(mfrow = c(1, 1))  # Restaurar configuración

cat("Nubes de palabras generadas exitosamente\n")

# Opción B: Nubes individuales (descomenta si quieres verlas grandes)
# for(libro in libros) {
#   palabras_libro <- words_clean %>%
#     filter(book == libro) %>%
#     count(word, sort = TRUE)
#
#   wordcloud(
#     words = palabras_libro$word,
#     freq = palabras_libro$n,
#     max.words = 150,
#     min.freq = 5,
#     random.order = FALSE,
#     colors = brewer.pal(8, "Dark2"),
#     scale = c(4, 0.5)
#   )
#
#   title(main = paste("Nube de Palabras -", libro),
#         cex.main = 2, font.main = 2)
#
#   readline(prompt = "Presiona Enter para la siguiente nube...")
# }
```

Y la resolución en consola mostró el siguiente resultado:

```
> # ============================
> # 9. NUBES DE PALABRAS
> # ============================
>
> cat("\n=== GENERANDO NUBES DE PALABRAS ===\n")

=== GENERANDO NUBES DE PALABRAS ===
>
> # Opción A: 4 nubes en una sola ventana (2x2)
> par(mfrow = c(2, 2), mar = c(1, 1, 3, 1))
>
> libros <- c("Dracula", "Frankenstein", "Time Machine", "War of the Worlds")
> colores <- list(
+   brewer.pal(8, "Set1"),
+   brewer.pal(8, "Set2"),
+   brewer.pal(8, "Paired"),
+   brewer.pal(8, "Dark2")
+ )
>
> for(i in 1:length(libros)) {
+   palabras_libro <- words_clean %>%
+     filter(book == libros[i]) %>%
+     count(word, sort = TRUE)
+
+   wordcloud(
+     words = palabras_libro$word,
+     freq = palabras_libro$n,
+     max.words = 100,
+     min.freq = 5,
+     random.order = FALSE,
+     colors = colores[[i]],
+     scale = c(3, 0.5)
+   )
+
+   title(main = libros[i], cex.main = 1.5, font.main = 2)
+ }
Hubo 13 avisos (use warnings() para verlos)
>
> par(mfrow = c(1, 1))  # Restaurar configuración
>
> cat("Nubes de palabras generadas exitosamente\n")
Nubes de palabras generadas exitosamente
>
> # Opción B: Nubes individuales (descomenta si quieres verlas grandes)
> # for(libro in libros) {
> #   palabras_libro <- words_clean %>%
> #     filter(book == libro) %>%
> #     count(word, sort = TRUE)
> #
> #   wordcloud(
> #     words = palabras_libro$word,
> #     freq = palabras_libro$n,
> #     max.words = 150,
> #     min.freq = 5,
> #     random.order = FALSE,
> #     colors = brewer.pal(8, "Dark2"),
> #     scale = c(4, 0.5)
> #   )
> #
> #   title(main = paste("Nube de Palabras -", libro),
> #         cex.main = 2, font.main = 2)
> #
> #   readline(prompt = "Presiona Enter para la siguiente nube...")
> # }
```

Y por último la figura presentada a continuación, muestra las nubes de palabras generadas para cada obra:



La figura presentada muestra las nubes de palabras generadas para cada obra:

Interpretación por libro:

En Dracula se destacan visualmente términos como "van", "helsing", "lucy" y "mina", coincidiendo con el análisis de frecuencias previo. El tamaño prominente de "helsing" refleja su papel central en la narrativa, mientras que los nombres "lucy" y "mina" evidencian la importancia de estos personajes femeninos en el desarrollo de la trama.

En Frankenstein, las palabras más grandes incluyen "father", "life", "eyes", "death" y "night", reflejando el tono filosófico y emocional de la obra. Términos como "miserable", "feelings" y "heart" enfatizan la carga emocional del texto, mientras que nombres como "elizabeth" y "clerval" representan a los personajes secundarios clave.

En The Time Machine, destaca visualmente "time" como el concepto central, acompañado de "machine", "morlocks" y "traveller". La presencia de términos como "night", "white" y "light" sugiere la importancia de las descripciones visuales y atmosféricas en la narrativa de ciencia ficción.

En The War of the Worlds, "martians" domina claramente la nube, seguido por "people", "black", "road" y "smoke". Palabras como "cylinder", "heat" y "pit" evocan elementos tecnológicos y destructivos de la invasión extraterrestre, mientras que términos como "london" y "houses" contextualizan geográficamente la narrativa.

Valor de las nubes de palabras:

Las nubes de palabras complementan el análisis de frecuencias al ofrecer:

- Una percepción visual inmediata de los términos dominantes
- Identificación rápida de temas centrales sin necesidad de leer números
- Representación estética que facilita la comunicación de resultados
- Comparación visual entre diferentes documentos

Sin embargo, es importante recordar que las nubes de palabras muestran frecuencia absoluta, no distintividad. Por ello, el análisis TF-IDF sigue siendo necesario para identificar términos verdaderamente característicos de cada obra.

Conclusión del análisis de frecuencia

Estos gráficos proporcionan una primera aproximación exploratoria al contenido de cada libro, permitiendo identificar términos recurrentes. Sin embargo, muchas de estas palabras corresponden a nombres propios o términos comunes dentro de la narrativa, por lo que posteriormente se complementa el análisis mediante TF-IDF para obtener palabras más distintivas o características de cada obra.

3. Cálculo del índice TF-IDF de los cuatro libros

Posteriormente, se calculó el índice TF-IDF, el cual permite identificar los términos más representativos o característicos de cada documento. Esta métrica destaca palabras que son frecuentes en un libro específico, pero poco comunes en los demás.

Código Implementado

A continuación, la explicación de todo el código implementado en RStudio para este punto:

Cálculo del índice TF-IDF (Term Frequency – Inverse Document Frequency)

Este bloque calcula el índice TF-IDF (Term Frequency - Inverse Document Frequency) para cada palabra en cada libro. El objetivo es identificar los términos más característicos o distintivos de cada obra.

Explicación paso a paso:

- words_clean %>%: Se parte del conjunto de palabras ya limpio (sin stopwords).
- count(book, word, sort = TRUE): Cuenta cuántas veces aparece cada palabra dentro de cada libro y genera la frecuencia absoluta (n).
- bind_tf_idf(word, book, n): Aquí ocurre lo más importante:
    - La función bind_tf_idf() calcula tres valores:
        - TF (Term Frequency) → Frecuencia relativa de la palabra dentro del libro.
        - IDF (Inverse Document Frequency) → Penaliza palabras que aparecen en muchos libros.
        - TF-IDF → Producto de ambos valores.
    - La fórmula es: $TF{-}IDF = TF \times IDF$. Esto permite destacar palabras que:
        - Son frecuentes en un libro
        - Pero no aparecen mucho en los demás
- arrange(desc(tf_idf)): Ordena los resultados de mayor a menor valor de TF-IDF. Así, las palabras más características aparecen primero.
- head(tf_idf): Muestra las primeras filas del resultado para verificar los términos con mayor valor TF-IDF.

En resumen, este bloque calcula el índice TF-IDF para identificar los términos más representativos de cada libro, permitiendo realizar un análisis comparativo más preciso que el basado únicamente en frecuencia. A continuación, la imagen del código:

```r
# ============================
# 10. CÁLCULO TF-IDF
# ============================
cat("\n=== CALCULANDO TF-IDF ===\n")

tf_idf <- words_clean %>%
  count(book, word, sort = TRUE) %>%
  bind_tf_idf(word, book, n) %>%
  arrange(desc(tf_idf))

head(tf_idf, 10)
```

Y la resolución en consola mostró el siguiente resultado:

```
> # ============================
> # 10. CÁLCULO TF-IDF
> # ============================
> cat("\n=== CALCULANDO TF-IDF ===\n")

=== CALCULANDO TF-IDF ===
>
> tf_idf <- words_clean %>%
+   count(book, word, sort = TRUE) %>%
+   bind_tf_idf(word, book, n) %>%
+   arrange(desc(tf_idf))
>
> head(tf_idf, 10)
# A tibble: 10 x 6
   book              word          n      tf   idf  tf_idf
   <chr>             <chr>     <int>   <dbl> <dbl>   <dbl>
 1 War of the Worlds martians    165 0.00722  1.39 0.0100
 2 Dracula           van         323 0.00661  1.39 0.00916
 3 Dracula           helsing     301 0.00616  1.39 0.00854
 4 Dracula           lucy        223 0.00456  1.39 0.00633
 5 Dracula           mina        210 0.00430  1.39 0.00596
 6 Time Machine      morlocks     48 0.00426  1.39 0.00591
 7 Time Machine      weena        46 0.00408  1.39 0.00566
 8 Dracula           jonathan    181 0.00370  1.39 0.00514
 9 War of the Worlds martian      79 0.00345  1.39 0.00479
10 Dracula           dr          162 0.00332  1.39 0.00460
>
```

4. Representación gráfica de los quince términos más característicos

Con los valores TF-IDF obtenidos, se extrajeron los quince términos más característicos de cada libro, lo cual permitió diferenciar claramente los temas y elementos propios de cada obra.

15 términos con mayor valor de TF-IDF en cada libro

Este bloque selecciona los 15 términos con mayor valor de TF-IDF en cada libro, es decir, las palabras más representativas o distintivas de cada obra.

Explicación paso a paso

- tf_idf %>%: Se parte del conjunto de datos tf_idf, el cual ya contiene para cada palabra:
  - su frecuencia absoluta (n)
  - su frecuencia relativa (tf)
  - su frecuencia inversa (idf)

    - su valor final TF-IDF (tf_idf)
- group_by(book): Agrupa los datos por libro. Este paso es fundamental porque queremos obtener los términos más característicos dentro de cada obra individualmente, no del corpus completo combinado.
- slice_max(tf_idf, n = 15): Selecciona las 15 palabras con el mayor valor de TF-IDF dentro de cada grupo (cada libro). En términos simples: Para cada libro, se eligen las 15 palabras más distintivas.
- top_tf_idf: Muestra en consola la tabla resultante con:
    - Libro
    - Palabra
    - Frecuencia
    - TF-IDF

En resumen, este bloque selecciona los quince términos más característicos de cada libro según el índice TF-IDF, proporcionando una caracterización temática más precisa y comparativa entre las obras analizadas. A continuación, la imagen del código:

```
155 # ============================
156 # 11. TOP 15 TÉRMINOS CARACTERÍSTICOS
157 # ============================
158 top_tf_idf <- tf_idf %>%
159   group_by(book) %>%
160   slice_max(tf_idf, n = 15)
161
162 print(top_tf_idf)
```

Y la resolución en consola mostró el siguiente resultado:

```
> # ============================
> # 11. TOP 15 TÉRMINOS CARACTERÍSTICOS
> # ============================
> top_tf_idf <- tf_idf %>%
+   group_by(book) %>%
+   slice_max(tf_idf, n = 15)
>
> print(top_tf_idf)
# A tibble: 60 x 6
# Groups:   book [4]
   book    word          n      tf   idf  tf_idf
   <chr>   <chr>     <int>   <dbl> <dbl>   <dbl>
 1 Dracula van         323 0.00661  1.39 0.00916
 2 Dracula helsing     301 0.00616  1.39 0.00854
 3 Dracula lucy        223 0.00456  1.39 0.00633
 4 Dracula mina        210 0.00430  1.39 0.00596
 5 Dracula jonathan    181 0.00370  1.39 0.00514
 6 Dracula dr          162 0.00332  1.39 0.00460
 7 Dracula count       153 0.00313  1.39 0.00434
 8 Dracula harker      113 0.00231  1.39 0.00321
 9 Dracula diary       110 0.00225  1.39 0.00312
10 Dracula madam        93 0.00190  1.39 0.00264
# i 50 more rows
# i Use `print(n = ...)` to see more rows
>
```

Grafico de los 15 terminos

Este bloque construye un gráfico de barras que muestra los 15 términos con mayor valor TF-IDF en cada libro. El objetivo es visualizar qué palabras son más distintivas en cada obra.

Explicación paso a paso

- top_tf_idf %>%: Se utiliza el dataset top_tf_idf, que ya contiene las 15 palabras más características por libro.
- ggplot(aes(...)): Se inicia la construcción del gráfico. Dentro de aes() se definen:
  - x = reorder(word, tf_idf): Ordena las palabras según su valor TF-IDF.
  - y = tf_idf: Representa el valor del índice TF-IDF.
  - fill = book: Asigna un color diferente según el libro.
- geom_col(show.legend = FALSE): Dibuja las barras del gráfico usando los valores TF-IDF. La leyenda se desactiva porque los libros ya están separados en paneles.
- coord_flip(): Invierte los ejes para que las barras sean horizontales, lo que mejora la lectura de las palabras.
- facet_wrap(~book, scales = "free"): Divide el gráfico en cuatro paneles, uno por cada libro. La opción scales = "free" permite que cada gráfico tenga su propia escala vertical, ya que los valores TF-IDF pueden variar entre libros.
- labs(...): Agrega:
  - Título del gráfico
  - Etiqueta del eje X
  - Etiqueta del eje Y

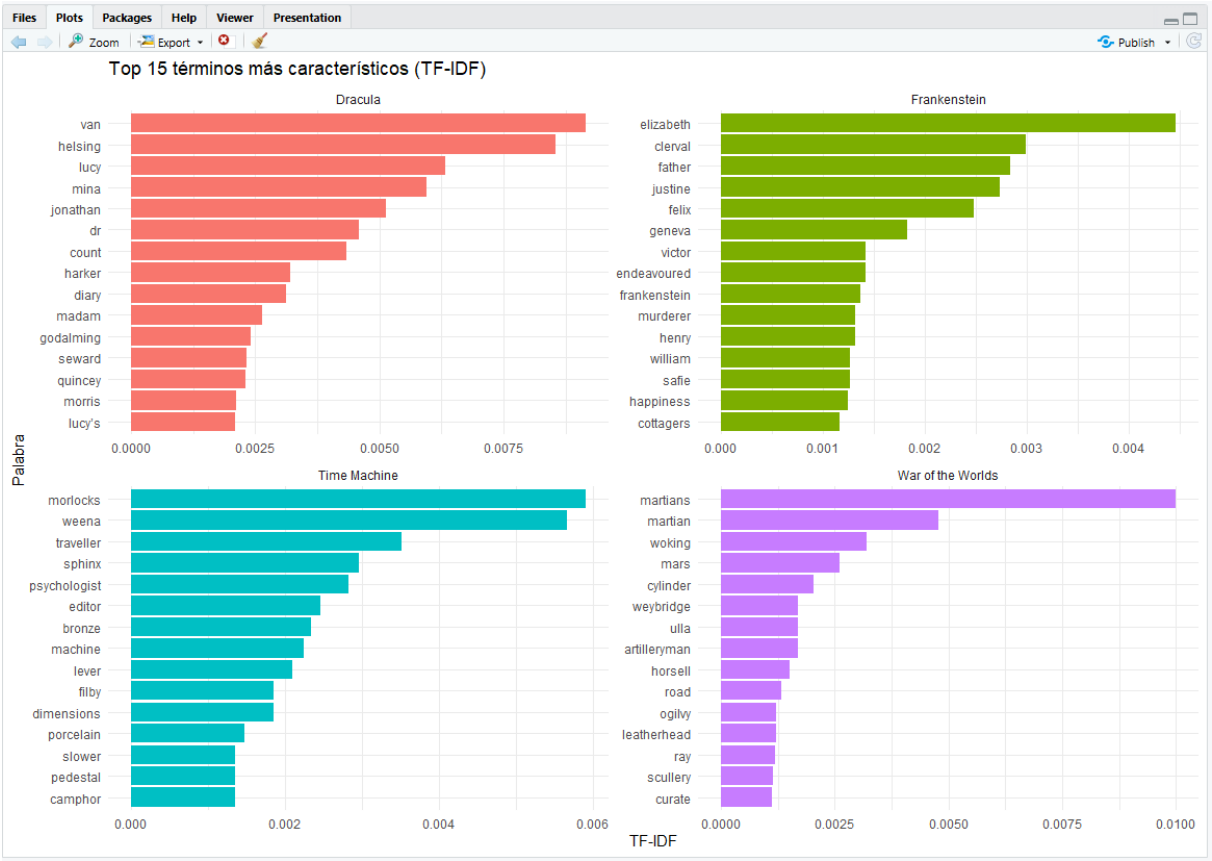
En resumen, este bloque genera una visualización comparativa de los términos más característicos según TF-IDF, permitiendo identificar de manera clara los elementos distintivos de cada obra literaria. A continuación, la imagen del código:

```
163
164 # ============================
165 # 12. GRÁFICO TF-IDF
166 # ============================
167 top_tf_idf %>%
168   ggplot(aes(x = reorder(word, tf_idf), y = tf_idf, fill = book)) +
169   geom_col(show.legend = FALSE) +
170   coord_flip() +
171   facet_wrap(~book, scales = "free") +
172   labs(
173     title = "Top 15 términos más característicos (TF-IDF)",
174     x = "Palabra",
175     y = "TF-IDF"
176   ) +
177   theme_minimal()
178
```

Y la resolución en consola mostró el siguiente resultado:

```
>
> # ============================
> # 12. GRÁFICO TF-IDF
> # ============================
> top_tf_idf %>%
+   ggplot(aes(x = reorder(word, tf_idf), y = tf_idf, fill = book)) +
+   geom_col(show.legend = FALSE) +
+   coord_flip() +
+   facet_wrap(~book, scales = "free") +
+   labs(
+     title = "Top 15 términos más característicos (TF-IDF)",
+     x = "Palabra",
+     y = "TF-IDF"
+   ) +
+   theme_minimal()
```

Junto con el siguiente grafico:



Donde la figura presentada muestra los quince términos más característicos de cada una de las cuatro obras analizadas, obtenidos mediante el índice TF-IDF. A diferencia del análisis basado únicamente en frecuencia, esta métrica permite identificar palabras que son representativas de un libro en particular y poco comunes en los demás.

Cada panel corresponde a un libro distinto y evidencia términos distintivos asociados a personajes, lugares o conceptos centrales dentro de la narrativa.

En Dracula destacan palabras como:

- helsing
- lucy
- mina
- jonathan

Estos términos corresponden principalmente a personajes principales de la obra, lo que demuestra que el TF-IDF identifica elementos propios del argumento y no solo palabras comunes.

Mientras que en Frankenstein sobresalen términos como:

- elizabeth
- clerval
- justine
- victor

Estos nombres representan personajes relevantes dentro de la historia y reflejan la estructura epistolar y emocional del texto.

Por otra parte, en The Time Machine se observan términos característicos como:

- morlocks

- weena
- traveller
- sphinx

Estos conceptos son exclusivos del universo narrativo de la obra y están directamente relacionados con su temática de viaje temporal y civilizaciones futuras.

Finalmente, en The War of the Worlds destacan términos como:

- martians
- martian
- woking
- mars

Estas palabras están vinculadas al contexto de invasión extraterrestre y a escenarios específicos descritos en la novela.

En resumen, este gráfico evidencia cómo el índice TF-IDF permite identificar vocabulario distintivo y temáticamente representativo de cada libro, proporcionando una caracterización más precisa que la frecuencia simple. Los términos obtenidos reflejan personajes, lugares y conceptos únicos de cada obra, facilitando el análisis comparativo entre textos literarios.

5. Comparación entre términos frecuentes y característicos

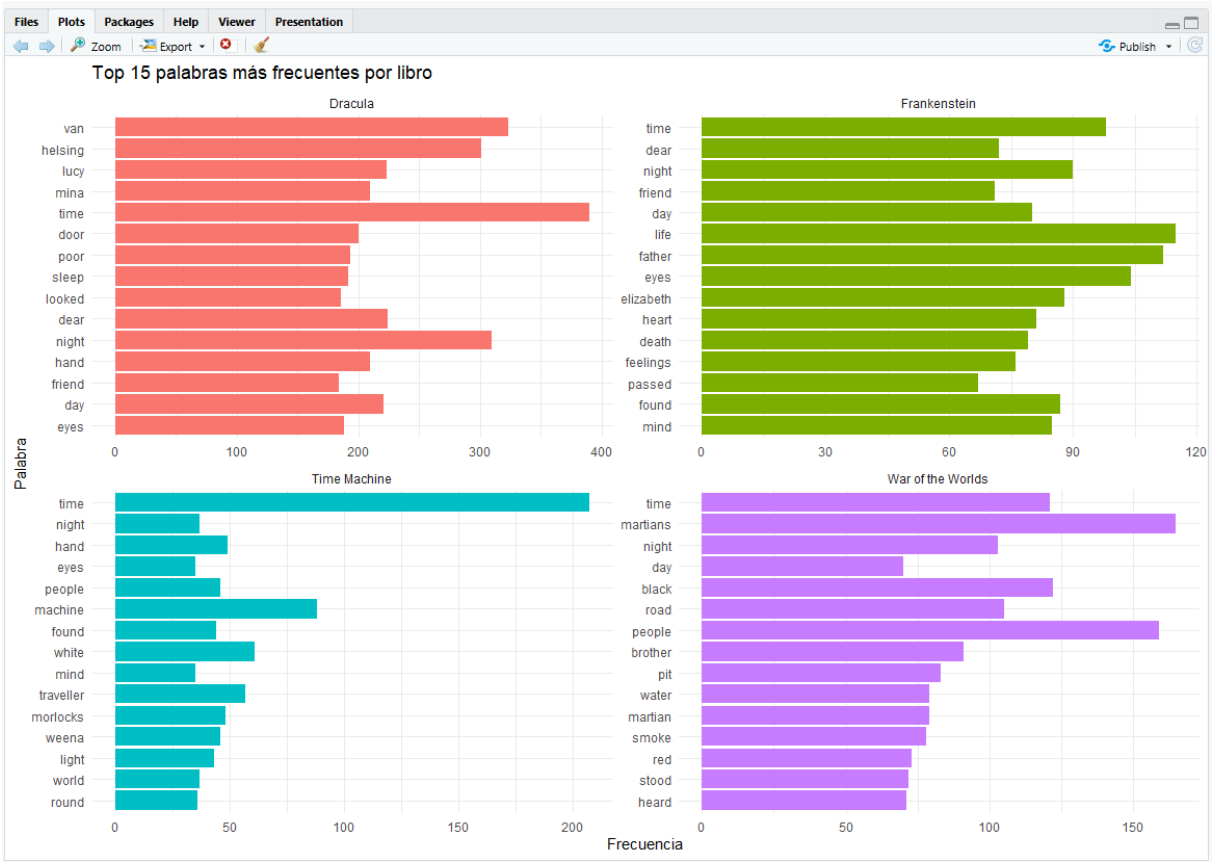
Las palabras más frecuentes reflejan términos comunes dentro del texto, pero no necesariamente permiten distinguir un libro de otro. En cambio, los términos con mayor TF-IDF resultan característicos, ya que aparecen principalmente en un solo libro.

Por ejemplo, en Dracula destacan nombres propios como “helsing” o “mina”, mientras que en War of the Worlds sobresalen términos como “martians”, directamente relacionados con la temática de invasión extraterrestre.

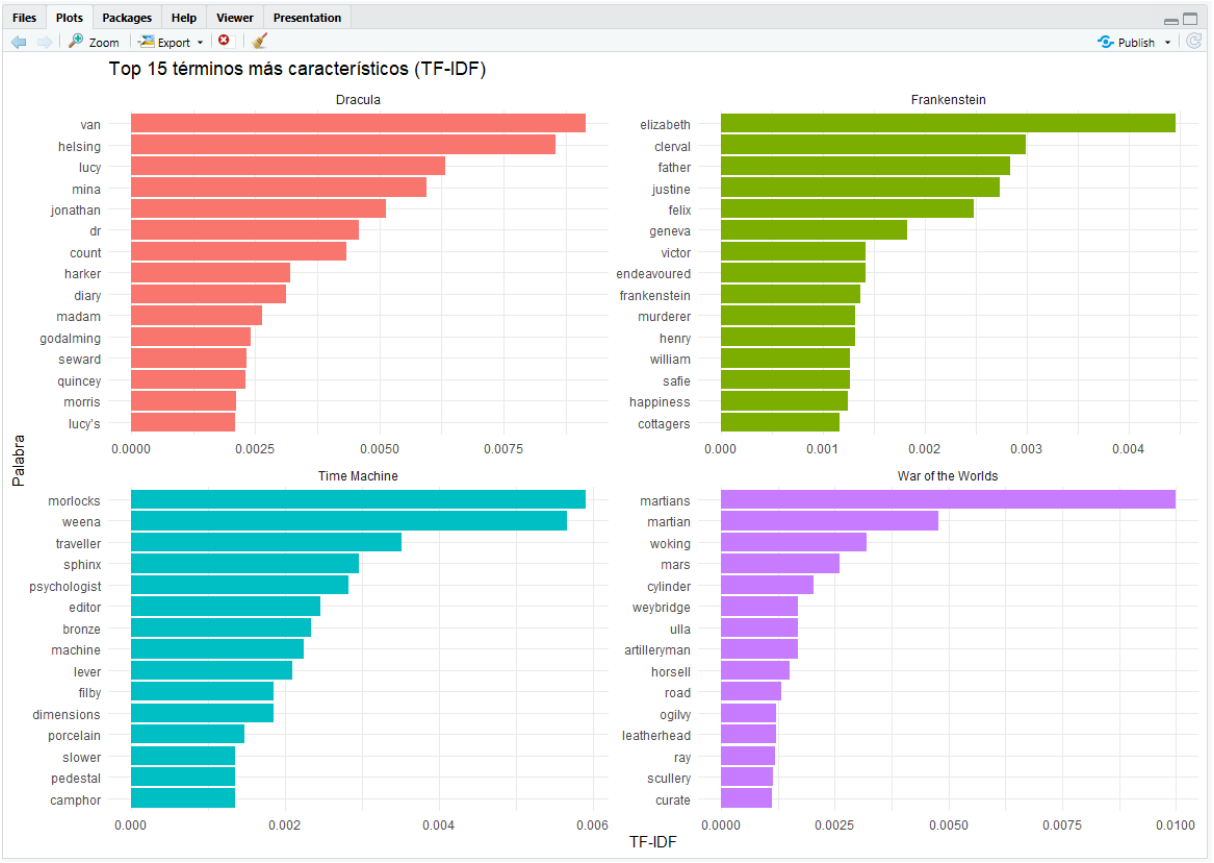
Comparación respectiva de los dos gráficos:

A continuación, y para dar una mejor explicación se adjuntan nuevamente las imágenes de los dos gráficos para dar claridad:

Primer Gráfico Frecuencia:



Segundo Grafico TF - IDF:



6. Asociaciones entre palabras (bigramas)

Finalmente, se realizó un análisis de asociaciones mediante bigramas, con el objetivo de identificar qué palabras aparecen frecuentemente junto a términos característicos.

Análisis de asociaciones mediante bigramas

Este bloque construye bigramas, es decir, pares consecutivos de palabras que aparecen juntas en el texto. El objetivo es analizar asociaciones entre términos y detectar relaciones léxicas dentro de las obras.

Explicación paso a paso

- Crear bigramas:
  - unnest_tokens(bigram, text, token = "ngrams", n = 2)
  - Aquí se indica que el texto debe dividirse en:
    - tokens de tipo "ngrams"
    - con tamaño n = 2
  - Es decir se crean combinaciones de dos palabras consecutivas.
  - Ejemplo: Si el texto contiene: van helsing opened door
    - Se generan:
    - van helsing
    - helsing opened
    - opened door
- Separar en palabra1 y palabra2:
  - separate(bigram, into = c("word1", "word2"), sep = " ")
  - Cada bigrama se divide en dos columnas:
    - word1
    - word2
  - Esto permite analizar cada componente por separado.
- Eliminar stopwords en bigramas
  - filter(!word1 %in% stop_words$word, !word2 %in% stop_words$word)
  - Se eliminan los pares donde alguna de las palabras sea una stopword.
  - Esto mejora la calidad del análisis, ya que evita asociaciones que no aportan significado relevante como:
    - the house
    - and then

En resumen, este bloque construye y limpia los bigramas del corpus, permitiendo analizar asociaciones léxicas entre palabras consecutivas dentro de las obras literarias. A continuación, la imagen del código:

```
179 # ============================
180 # 13. CREAR BIGRAMAS
181 # ============================
182 cat("\n=== GENERANDO BIGRAMAS ===\n")
183
184 bigrams <- corpus %>%
185   unnest_tokens(bigram, text, token = "ngrams", n = 2)
186
187 # Separar en palabra1 y palabra2
188 bigrams_separated <- bigrams %>%
189   separate(bigram, into = c("word1", "word2"), sep = " ")
190
191 # Eliminar stopwords
192 bigrams_filtered <- bigrams_separated %>%
193   filter(!word1 %in% stop_words$word,
194          !word2 %in% stop_words$word)
195
196 cat("Bigramas creados:", nrow(bigrams_filtered), "\n")
```

Y la resolución en consola mostró el siguiente resultado:

```
> # ============================
> # 13. CREAR BIGRAMAS
> # ============================
> cat("\n=== GENERANDO BIGRAMAS ===\n")

=== GENERANDO BIGRAMAS ===
>
> bigrams <- corpus %>%
+   unnest_tokens(bigram, text, token = "ngrams", n = 2)
>
> # Separar en palabra1 y palabra2
> bigrams_separated <- bigrams %>%
+   separate(bigram, into = c("word1", "word2"), sep = " ")
>
> # Eliminar stopwords
> bigrams_filtered <- bigrams_separated %>%
+   filter(!word1 %in% stop_words$word,
+          !word2 %in% stop_words$word)
>
> cat("Bigramas creados:", nrow(bigrams_filtered), "\n")
Bigramas creados: 29105
>
```

Asociaciones con “helsing”

Este bloque identifica las asociaciones más frecuentes del término “helsing” dentro del libro Dracula utilizando los bigramas previamente construidos. El objetivo es determinar qué palabras aparecen con mayor frecuencia junto a “helsing”.

Explicación paso a paso

- bigrams_filtered %>%: Se parte del conjunto de bigramas ya limpiado (sin stopwords).
- filter(word1 == "helsing" | word2 == "helsing"): Se seleccionan únicamente los pares de palabras donde la primera palabra sea "helsing" o la segunda palabra sea "helsing". Es decir, se buscan todas las combinaciones donde aparezca el término “helsing”.
- count(word1, word2, sort = TRUE): Se cuentan las veces que aparece cada bigrama y se ordenan de mayor a menor frecuencia. Esto permite identificar las asociaciones más fuertes.
- head(helsing_assoc, 10): Muestra las 10 asociaciones más frecuentes.

Interpretación del resultado

El resultado principal suele ser: **van helsing**

Lo cual confirma que el análisis detecta correctamente el nombre completo del personaje. Otras asociaciones como:

- helsing stood
- helsing called

Reflejan acciones narrativas vinculadas al personaje dentro del texto.

En resumen, este bloque analiza las asociaciones del término “helsing” en Dracula, identificando las palabras que co-ocurren con mayor frecuencia y evidenciando relaciones semánticas relevantes dentro de la narrativa. A continuación, la imagen del código:

```r
# ============================
# 14. ASOCIACIONES - HELSING
# ============================
cat("\n=== ASOCIACIONES CON 'HELSING' ===\n")

helsing_assoc <- bigrams_filtered %>%
  filter(word1 == "helsing" | word2 == "helsing") %>%
  count(word1, word2, sort = TRUE)

print(head(helsing_assoc, 10))
```

Y la resolución en consola mostró el siguiente resultado:

```
> # ============================
> # 14. ASOCIACIONES - HELSING
> # ============================
> cat("\n=== ASOCIACIONES CON 'HELSING' ===\n")

=== ASOCIACIONES CON 'HELSING' ===
>
> helsing_assoc <- bigrams_filtered %>%
+   filter(word1 == "helsing" | word2 == "helsing") %>%
+   count(word1, word2, sort = TRUE)
>
> print(head(helsing_assoc, 10))
# A tibble: 10 x 3
   word1   word2         n
   <chr>   <chr>     <int>
 1 van     helsing     282
 2 helsing stepped       5
 3 helsing amsterdam     3
 4 helsing raised        3
 5 helsing stood         3
 6 helsing beckoned      2
 7 helsing called        2
 8 helsing held          2
 9 helsing holding       2
10 helsing motioned      2
```

Asociaciones con “martians”

Este bloque identifica las asociaciones más frecuentes del término “martians” dentro del libro The War of the Worlds, utilizando los bigramas previamente generados y filtrados. El objetivo es analizar qué palabras aparecen con mayor frecuencia junto a “martians”.

Explicación paso a paso

- bigrams_filtered %>%: Se parte del conjunto de bigramas ya limpiado, es decir, sin palabras vacías (stopwords).
- filter(word1 == "martians" | word2 == "martians"): Se seleccionan únicamente los pares de palabras donde la primera palabra sea “martians”, o la segunda palabra sea “martians”. En otras palabras se extraen todas las combinaciones en las que aparece el término “martians”.
- count(word1, word2, sort = TRUE): Se cuentan las veces que aparece cada bigrama y se ordenan de mayor a menor frecuencia. Esto permite identificar cuáles son las asociaciones más representativas.
- head(martians_assoc, 10): Muestra las 10 asociaciones más frecuentes.

Interpretación del resultado

Las asociaciones suelen incluir combinaciones como:

- martians advancing
- martians appeared
- martians aimed

Estas combinaciones reflejan acciones y descripciones relacionadas con la invasión extraterrestre, evidenciando el carácter dinámico y conflictivo de la narrativa.

En resumen, este bloque analiza las asociaciones del término “martians” en The War of the Worlds, identificando las palabras que co-ocurren con mayor frecuencia y evidenciando relaciones semánticas propias del contexto narrativo de la obra. A continuación, la imagen del código:

```
209 # ============================
210 # 15. ASOCIACIONES - MARTIANS
211 # ============================
212 cat("\n=== ASOCIACIONES CON 'MARTIANS' ===\n")
213
214 martians_assoc <- bigrams_filtered %>%
215   filter(word1 == "martians" | word2 == "martians") %>%
216   count(word1, word2, sort = TRUE)
217
218 print(head(martians_assoc, 10))
```

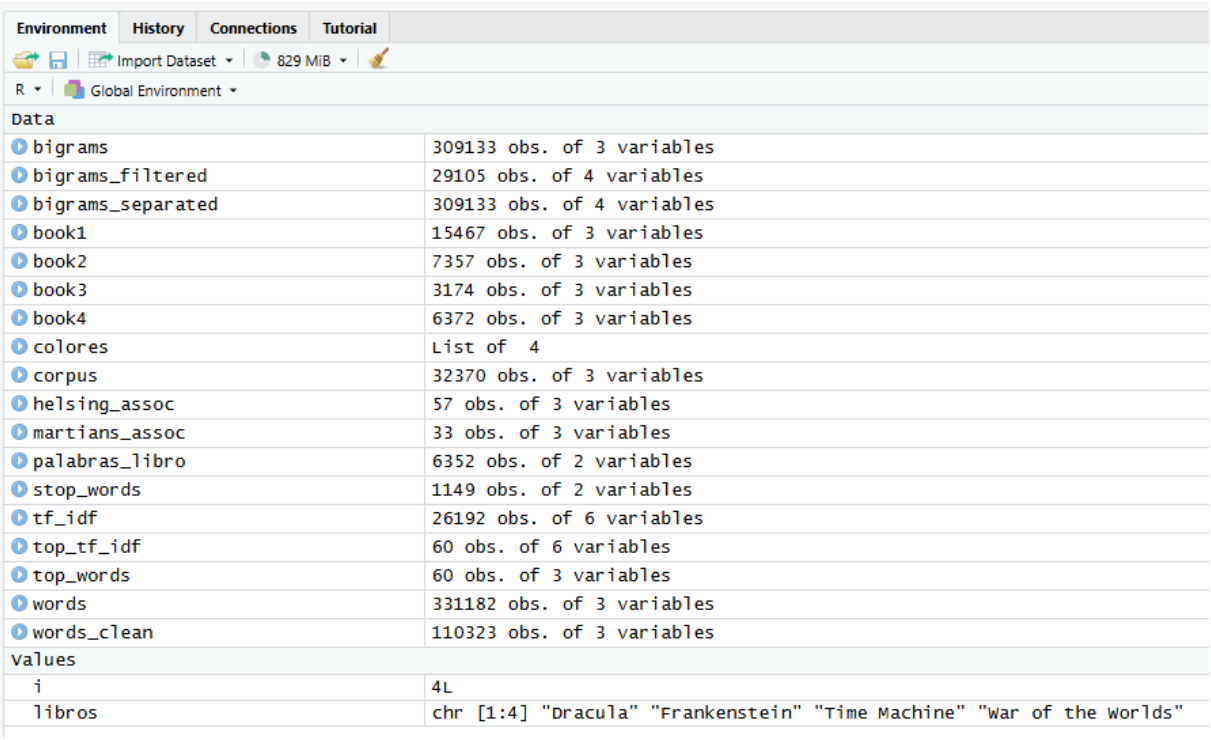
Y la resolución en consola mostró el siguiente resultado:

```
> # ============================
> # 15. ASOCIACIONES - MARTIANS
> # ============================
> cat("\n=== ASOCIACIONES CON 'MARTIANS' ===\n")

=== ASOCIACIONES CON 'MARTIANS' ===
>
> martians_assoc <- bigrams_filtered %>%
+   filter(word1 == "martians" | word2 == "martians") %>%
+   count(word1, word2, sort = TRUE)
>
> print(head(martians_assoc, 10))
# A tibble: 10 × 3
   word1      word2          n
   <chr>      <chr>      <int>
 1 actual     martians       1
 2 glittering martians       1
 3 invisible  martians       1
 4 martians   _dead_         1
 5 martians   advancing      1
 6 martians   aimed          1
 7 martians   alarmed        1
 8 martians   altogether     1
 9 martians   appeared       1
10 martians   boilers        1
>
> # ============================
> # FIN DEL LABORATORIO
> # ============================
```

7. Verificación del entorno de trabajo en RStudio

La siguiente figura corresponde al panel Environment de RStudio, el cual muestra todos los objetos generados durante la ejecución completa del laboratorio. Esta sección permite verificar que cada etapa del procesamiento se realizó correctamente y que los resultados fueron almacenados en estructuras de datos disponibles para su análisis posterior.

Imagen de la figura:



Interpretación detallada de los objetos generados:

El panel Environment muestra un total de 18 objetos de datos creados durante el laboratorio, los cuales se pueden clasificar en las siguientes categorías:

1. Libros descargados del Proyecto Gutenberg:

- book1 (15,467 obs. de 3 variables) → Dracula
- book2 (7,357 obs. de 3 variables) → Frankenstein
- book3 (3,174 obs. de 3 variables) → The Time Machine
- book4 (6,372 obs. de 3 variables) → The War of the Worlds

Cada uno de estos objetos contiene el texto completo de cada libro descargado, organizado en filas (observaciones) que representan líneas de texto, con tres columnas que incluyen el ID de Gutenberg, el texto y el nombre del libro.

2. Corpus unificado:

- corpus (32,370 obs. de 3 variables)

Este objeto representa la unión de los cuatro libros en un único conjunto de datos. El número total de observaciones (32,370) corresponde a la suma de todas las líneas de texto de las cuatro obras, confirmando que la integración del corpus fue exitosa.

3. Tokenización y limpieza:

- words (331,182 obs. de 3 variables) → Texto tokenizado completo.
- words_clean (110,323 obs. de 3 variables) → Texto después de eliminar stopwords.

La diferencia entre ambos objetos evidencia el impacto de la eliminación de stopwords: se eliminaron aproximadamente 220,859 palabras vacías (66.7% del total), dejando únicamente las palabras con significado relevante para el análisis.

4. Análisis de frecuencias:

top_words (60 obs. de 3 variables) → Las 15 palabras más frecuentes por libro (15 × 4 = 60).
stop_words (1,149 obs. de 2 variables) → Lista de stopwords en inglés utilizada.

El objeto top_words contiene exactamente 60 filas porque almacena las 15 palabras más frecuentes de cada uno de los 4 libros.

5. Análisis TF-IDF:

- tf_idf (26,192 obs. de 6 variables) → Cálculo completo de TF-IDF para todas las palabras.
- top_tf_idf (60 obs. de 6 variables) → Los 15 términos más característicos por libro.

El objeto tf_idf contiene el cálculo del índice TF-IDF para todas las palabras únicas del corpus limpio, incluyendo las columnas: libro, palabra, frecuencia (n), term frequency (tf), inverse document frequency (idf), y el valor final tf_idf. El objeto top_tf_idf filtra únicamente los 15 términos con mayor valor TF-IDF en cada libro.

6. Análisis de bigramas:

- bigrams (309,133 obs. de 3 variables) → Bigramas completos generados.
- bigrams_separated (309,133 obs. de 4 variables) → Bigramas separados en word1 y word2.
- bigrams_filtered (29,105 obs. de 4 variables) → Bigramas después de eliminar stopwords.

La generación de bigramas produjo 309,133 pares de palabras consecutivas. Al eliminar aquellos que contenían stopwords, el conjunto se redujo a 29,105 bigramas significativos (9.4% del total).

7. Asociaciones específicas:

- helsing_assoc (57 obs. de 3 variables) → Bigramas relacionados con "helsing".
- martians_assoc (33 obs. de 3 variables) → Bigramas relacionados con "martians".

Estos objetos contienen las asociaciones léxicas identificadas para términos característicos seleccionados de Dracula y The War of the Worlds, respectivamente.

8. Otros elementos:

- colores (List of 4) → Lista de paletas de colores para las visualizaciones.
- palabras_libro (6,352 obs. de 2 variables) → Último objeto temporal generado durante la creación de nubes de palabras.
- libros (chr [1:4]) → Vector con los nombres de los cuatro libros: "Dracula", "Frankenstein", "Time Machine", "War of the Worlds".

Y con este ultimo punto se da por finalizado el desarrollo de este laboratorio.

¡¡¡Mil gracias respetado profesor.!!!

CONCLUSIONES DE LA ACTIVIDAD

Para concluir este trabajo, se puede afirmar que el desarrollo del laboratorio permitió comprender de manera práctica el funcionamiento y alcance de las técnicas de minería de texto y procesamiento de lenguaje natural (PLN) aplicadas a un corpus literario real. A través del uso del lenguaje R y librerías

especializadas como tidytext y gutenbergr, se evidenció cómo es posible transformar texto no estructurado en información cuantificable y analizable mediante métodos estadísticos.

A diferencia de un enfoque meramente teórico, en el cual los conceptos de tokenización, frecuencia de términos o TF-IDF se estudian de forma abstracta, la implementación práctica permitió construir el conocimiento de manera progresiva y estructurada. El proceso comenzó con la obtención del corpus desde el Proyecto Gutenberg, seguido de su limpieza y tokenización, el cálculo de frecuencias, la identificación de términos característicos mediante TF-IDF y, finalmente, el análisis de asociaciones léxicas mediante bigramas. Esta secuencia organizada facilitó la comprensión integral de cada etapa del procesamiento textual.

En el análisis de frecuencias se observó que las palabras más repetidas en cada libro reflejan elementos centrales de la narrativa, como nombres de personajes y conceptos recurrentes. Sin embargo, este tipo de análisis presenta una limitación importante: no necesariamente distingue qué términos son realmente representativos de cada obra en comparación con las demás.

Por esta razón, el cálculo del índice TF-IDF resultó fundamental. Esta métrica permitió identificar términos distintivos de cada libro, penalizando aquellos que aparecen en múltiples documentos y destacando palabras específicas de cada narrativa. Por ejemplo, en Dracula resaltaron nombres propios como “helsing” y “mina”; en The Time Machine, términos como “morlocks”; y en The War of the Worlds, palabras como “martians”. Esto demuestra la capacidad del TF-IDF para caracterizar temáticamente los documentos de manera comparativa.

Un aspecto especialmente relevante evidenciado en este laboratorio es la diferencia entre dos enfoques dentro del análisis textual:

- Análisis basado en frecuencia: Permite identificar los términos más repetidos dentro de un documento, proporcionando una primera aproximación exploratoria del contenido.
- Análisis basado en TF-IDF: Permite identificar los términos más característicos o distintivos de cada documento en relación con un conjunto de textos, ofreciendo una caracterización más precisa y comparativa.

Ambos enfoques son complementarios: mientras la frecuencia muestra patrones generales de repetición, el TF-IDF resalta elementos semánticos únicos.

Asimismo, el análisis de asociaciones mediante bigramas evidenció que el procesamiento de lenguaje natural no se limita al estudio de palabras individuales, sino que también puede capturar relaciones contextuales entre términos. La identificación de combinaciones como “van helsing” o “martians advancing” demuestra que el análisis de co-ocurrencias permite comprender vínculos semánticos dentro de la narrativa, aportando un nivel adicional de interpretación.

En síntesis, esta actividad permitió aplicar de manera práctica los fundamentos de la minería de texto, consolidando habilidades en manipulación de datos textuales, limpieza de corpus, cálculo de métricas estadísticas y visualización gráfica de resultados. Más allá de los resultados obtenidos, el laboratorio evidenció la importancia de un flujo estructurado de procesamiento —desde la obtención del corpus hasta la interpretación de asociaciones— para garantizar un análisis textual riguroso y coherente.

En conclusión, el trabajo demuestra cómo las técnicas de procesamiento de lenguaje natural constituyen herramientas fundamentales para el análisis automatizado de documentos, permitiendo transformar grandes volúmenes de texto en conocimiento estructurado y significativo dentro del ámbito académico y profesional.

BIBLIOGRAFÍA

A continuación, la bibliografía implementada en este desarrollo:

- Tema 1. Minería de texto. Procesadores de Lenguajes (COLGII) - PER 15746 - Enero 2026.
- Tema 2. Procesamiento del lenguaje natural. Procesadores de Lenguajes (COLGII) - PER 15746 - Enero 2026.
- Clases virtuales con el profesor Ing. Rogerio Orlando Beltrán Castro.

- Hester, J., & Bryan, J. (2023). readr: Read rectangular text data. R package version 2.1.6. https://CRAN.R-project.org/package=readr
- Silge, J., & Robinson, D. (2016). tidytext: Text mining and analysis using tidy data principles in R. Journal of Open Source Software, 1(3), 37. https://doi.org/10.21105/joss.00037
- Silge, J., & Robinson, D. (2023). tidytext: Text mining using tidy data principles. R package version 0.4.3. https://CRAN.R-project.org/package=tidytext
- Wickham, H. et al. (2023). dplyr: A grammar of data manipulation. R package version 1.2.0. https://CRAN.R-project.org/package=dplyr
- Wickham, H. (2016). ggplot2: Elegant graphics for data analysis. Springer-Verlag New York.
- Wickham, H., & Bryan, J. (2023). tidyr: Tidy messy data. R package version 1.3.0. https://CRAN.R-project.org/package=tidyr
- Feinerer, I., Hornik, K., & Meyer, D. (2008). Text mining infrastructure in R. Journal of Statistical Software, 25(5), 1–54.
- Fellows, I. (2018). wordcloud: Word clouds. R package version 2.6. https://CRAN.R-project.org/package=wordcloud
- Robinson, D. (2023). gutenbergr: Download and process public domain works from Project Gutenberg. R package version 0.4.1. https://CRAN.R-project.org/package=gutenbergr
- Project Gutenberg. (2024). Project Gutenberg. https://www.gutenberg.org
- Stoker, B. (1897). Dracula. Project Gutenberg.
- Shelley, M. (1818). Frankenstein; or, The Modern Prometheus. Project Gutenberg.
- Wells, H. G. (1895). The Time Machine. Project Gutenberg.
- Wells, H. G. (1898). The War of the Worlds. Project Gutenberg.

AGRADECIMIENTO

Finalmente, deseo expresar mi más sincero agradecimiento al profesor Ing. Rogerio Orlando Beltrán Castro, por los conocimientos, orientación y acompañamiento brindados durante el desarrollo de esta actividad. Sus explicaciones y aportes en el área de procesamiento de lenguajes y análisis computacional de datos fueron fundamentales para comprender los principios de la minería de texto y su aplicación práctica mediante herramientas estadísticas.

Gracias a los conceptos compartidos en clase, fue posible desarrollar este laboratorio de manera estructurada, aplicando correctamente técnicas de tokenización, limpieza de corpus, análisis de frecuencia, cálculo del índice TF-IDF y estudio de asociaciones léxicas. La guía metodológica proporcionada permitió abordar el análisis paso a paso, garantizando una comprensión clara del flujo completo de procesamiento textual.

Sin duda, esta experiencia fortaleció significativamente mi formación académica, ampliando mi comprensión sobre cómo el lenguaje natural puede ser transformado en datos estructurados y analizados mediante herramientas computacionales modernas. Este laboratorio no solo consolidó conceptos teóricos, sino que también permitió evidenciar la relevancia práctica del procesamiento de lenguaje natural dentro del campo de la informática.

¡¡¡Mil gracias profesor!!!

Respetuosamente,

Alejandro De Mendoza