Facultad de Ciencias Económicas y Estadística

Universidad Nacional de Rosario

Tesina de grado

Licenciatura en Estadística

Clasificación supervisada de textos de ficción según género utilizando bosques aleatorios

Alumno: Santiago García Sánchez

Director: Mg. Diego Marfetán Molina

Codirector: Mg. Marcos Miguel Prunello

Año 2023

Resumen

Se llama aprendizaje automático a la disciplina que estudia el desarrollo y aplicación de algoritmos que descubren patrones presentes en datos provistos como entrada. Estos patrones pueden ser usados posteriormente para realizar predicciones sobre nuevos datos. Entre los distintos tipos de aprendizaje automático, se encuentran las técnicas de aprendizaje supervisado. En este tipo de algoritmos, se construye un modelo para predecir o estimar los valores de una variable respuesta basado en una o más variables predictoras (también llamadas atributos). Un ejemplo son los árboles de decisión, método que consiste en encontrar una serie de criterios para clasificar las observaciones dentro de determinadas clases. El algoritmo de bosques aleatorios se trata de una expansión de los árboles de decisión, en el que se construyen múltiples árboles y la categoría a la que más frecuentemente es asignada una observación es la que es tomada como clase predicha.

Los bosques aleatorios presentan un buen desempeño en tareas de clasificación. Una posible aplicación de esta técnica es la clasificación de datos textuales. Esta tesina tuvo como objetivo evaluar el rendimiento de bosques aleatorios en la tarea de clasificar textos de ficción en español, recurriendo para ello a un conjunto de más de 700 libros pertenecientes a siete géneros literarios distintos.

Existen diversas maneras de construir atributos que permitan sintetizar la información de los documentos. En este trabajo se emplearon la frecuencia de términos (frecuencia absoluta de una palabra en el documento) y la estadística TF-IDF (frecuencia absoluta de un término ponderada por la cantidad de documentos en los que aparece). Dichas medidas fueron calculadas tanto para términos individuales como para pares de palabras consecutivas, llamados bigramas. Además, se experimentó con el uso de un filtro para eliminar los nombres propios de los conjuntos de datos, con el objetivo de evitar el sobreajuste del modelo. Combinando estos elementos, en total se plantearon ocho escenarios distintos para comparar el desempeño de los bosque aleatorios en la tarea de clasificación literaria.

En general, los mejores resultados se obtuvieron al utilizar frecuencia de términos, palabras individuales y sin filtrar los nombres propios. El algoritmo permitió obtener una precisión global aceptable, aunque la precisión de la clasificación varió notablemente según el género.

Palabras clave: clasificación de textos, bosques aleatorios, géneros literarios

Índice

1. Introducción

Se denomina **aprendizaje automático** o *machine learning* a la disciplina que estudia el desarrollo y aplicación de algoritmos que descubren patrones presentes en datos provistos como entrada. Estos patrones pueden ser usados posteriormente para realizar predicciones sobre nuevos datos. Cuando se cuenta con la presencia de una variable respuesta para guiar el proceso de aprendizaje, se habla de algoritmos de aprendizaje supervisado. Por el contrario, cuando solo se dispone de un conjunto de variables explicativas y el objetivo es describir cómo se organizan o agrupan los datos, se utilizan algoritmos de aprendizaje no supervisado. Si bien existen otros tipos de aprendizaje (tales como transducción o aprendizaje semi-supervisado), los algoritmos utilizados más frecuentemente suelen ubicarse en las dos categorías ya mencionadas.

Una de las áreas de aplicación de los algoritmos de aprendizaje automático es la **clasificación de textos**, el proceso por el cual un conjunto de documentos son asignados a categorías o clases. En este contexto, un **documento** se puede definir como una unidad de datos textuales dentro de una colección, generalmente relacionado con algún documento del mundo real como un artículo, noticia, correo electrónico o libro (Feldman y Sanger, 2013). A esta colección o conjunto de documentos se la denomina **corpus**. En el proceso de clasificación, los textos de una misma clase comparten características semejantes, que pueden incluir temas, forma, idioma, autoría o popularidad, entre otros. Cuando los documentos corresponden a obras de ficción, suelen ser clasificados según su género literario.

Al trabajar con algoritmos de aprendizaje automático en la clasificación de textos hay dos enfoques posibles. El primero consiste en utilizar un conjunto de documentos sin etiquetar, es decir que no hay clases predefinidas en la muestra. El objetivo es entonces agrupar observaciones con características similares, por lo que se aplican algoritmos de aprendizaje no supervisado. El segundo enfoque radica en utilizar datos ya etiquetados dentro de clases predeterminadas, con el interés de obtener una serie de criterios para clasificar futuras observaciones. Este enfoque se aborda con algoritmos de aprendizaje supervisado y es el utilizado en el presente trabajo.

La tarea de asignar etiquetas a textos se lleva a cabo desde la antigüedad, a medida que las colecciones de libros fueron creciendo y localizar un documento específico se hizo más difícil. Atribuirle a cada texto una etiqueta pertinente a su temática permitió optimizar las tareas de búsqueda al proveer información sobre su contenido (Baeza-Yates y Ribeiro-Neto, 2011). El uso de técnicas estadísticas para la clasificación automática de textos comenzó a desarrollarse en los años 1960, siendo el trabajo de Maron (1961) uno de los primeros en explorar esta temática. Durante los años '80 surgió el enfoque conocido como “ingeniería del conocimiento”, en el cual los expertos definían manualmente un grupo de reglas para la clasificación que luego las computadoras seguían con el fin de categorizar los documentos de un

corpus (Sebastiani, 2002).

A partir de la década de 1990, gracias a los avances en computación y a la aparición de nuevas metodologías, la clasificación automática de textos cobró una mayor popularidad. Se abandonó el enfoque basado en la ingeniería del conocimiento en favor de métodos de aprendizaje automático, obteniéndose mejores resultados y reduciendo la necesidad de intervención manual por parte de expertos. Entre los primeros ejemplos, Joachims (1998) utilizó máquinas de vectores de soporte para clasificar documentos médicos según su categoría, mientras que Nigam (1999) clasificó distintos tipos de sitios *web* mediante modelos de máxima entropía. Algunas áreas de aplicación que aparecieron durante esta época incluyen filtrado de *spam*, atribución de autoría e identificación de géneros de forma automática, organización jerárquica de páginas *web* y codificación automática de encuestas (Zanasi, 2007)

En años recientes, especialmente a partir de 2016, ha habido un aumento en publicaciones vinculadas a la clasificación automática de textos (Mirończuk y Protasiewicz, 2018). Esto se debe a una combinación de avances científicos en el área y un incremento en el interés general por el potencial del aprendizaje automático, *big data* e inteligencia artificial (Falk, 2019). Entre las nuevas aplicaciones que han surgido, se encuentran el análisis de mensajes de redes sociales (Cárdenas et al., 2014) o de reseñas escritas por usuarios (Das et al., 2021).

Para esta tesina, se realizó una revisión bibliográfica de trabajos vinculados a la clasificación automática de textos, examinando publicaciones en español e inglés intentando que sean relativamente recientes y que utilicen distintas técnicas de aprendizaje supervisado. Mediante esta revisión, se encontró que los trabajos difieren en cuanto al método de aprendizaje escogido, la unidad de análisis (términos individuales o pares de términos consecutivos, conocidos como bigramas) y las variables construidas para la clasificación (entre ellas, frecuencias absolutas de cada término o bigrama o la estadística TD-IDF que representa una frecuencia absoluta ponderada).

Diversos autores intentan clasificar documentos según un sentimiento positivo, negativo o neutro. Por ejemplo, Cárdenas et al. (2014) clasificaron publicaciones de la red social Twitter (llamados comúnmente tuits) en distintos contextos, utilizando para ello redes de palabras, clasificador bayesiano ingenuo (*naive bayes*) y máquinas de vectores de soporte (SVM). Este último algoritmo proveyó los mejores resultados, con una precisión superior al 73 % en las tres categorías. De manera similar, Das et al. (2021) contrastaron seis técnicas de aprendizaje supervisado y dos escenarios (frecuencia absoluta de bigramas y TF-IDF para términos individuales) con el objetivo de clasificar reseñas. Utilizaron un conjunto extraído de IMDB (reseñas de películas) y otro de Amazon (reseñas de productos). El mejor resultado (precisión del 94 %) se obtuvo con bosques aleatorios y TF-IDF. Pranckevičius y Marcinkevičius (2017) también clasificaron reseñas, en este caso de aplicaciones para Android, pero utilizando una escala numérica

(siendo 1 el valor más bajo y 5 el mayor). Contrastaron 5 técnicas y 5 conjuntos de atributos, obteniendo precisiones por debajo del 50 % en todos los casos excepto con regresión logística, la cual alcanzó una precisión del 58,5 %.

Otro tipo de texto cuya clasificación se ha investigado exhaustivamente son los artículos periodísticos. Salinas e Izeta (2016) crearon conjuntos de datos con noticias en español y aplicaron bosques aleatorios contrastando distintos escenarios. En todos ellos, alcanzaron una precisión superior al 79 %. Yadla y Rao (2020) recurrieron a un conjunto de noticias de la BBC dividido en cinco categorías distintas. Trabajaron con la estadística TF-IDF y contrastaron siete técnicas diferentes, obteniéndose una precisión del 98 % mediante una red neuronal. Este mismo conjunto de datos fue utilizado por Shah et al. (2020), quienes también utilizaron la estadística TF-IDF y contrastaron distintas técnicas (regresión logística, k vecinos más cercanos (KNN) y bosques aleatorios). El mejor resultado se dio con regresión logística, aunque todas las técnicas tuvieron muy buen desempeño, siendo la precisión siempre superior al 86 %. Como se puede ver, los conjuntos de noticias suelen dar buenos resultados independientemente de la técnica empleada.

También se ha estudiado la clasificación de textos académicos, generalmente provenientes de universidades. Un ejemplo es el trabajo de Venegas (2007), que intentó clasificar textos de 4 carreras distintas mediante *naive bayes* y SVM. La precisión alcanzada con este último método fue del 95 % en promedio. Por su parte, Beltrán (2011) clasificó documentos de tres disciplinas distintas ajustando un modelo de regresión logística multinomial. Realizó un análisis morfológico, es decir que las variables utilizadas fueron la cantidad de verbos, sustantivos, preposiciones, etc. presentes en el documento. La precisión obtenida fue aproximadamente del 86 %.

Otros ejemplos incluyen a Ávila A. (2008), quien clasificó libros de ciencia utilizando las categorías de la Biblioteca del Congreso de los Estados Unidos. Se contrastaron diversos algoritmos y estadísticas, obteniendo precisiones muy variables según el escenario. Calvo Torres (2017) trabajó con un conjunto de 20000 mensajes provenientes de 20 grupos de noticias (semejantes a foros). Calculó frecuencias de términos y TF-IDF y utilizó SVM y regresión logística como técnicas. También contrastó distintos métodos de reducción de dimensionalidad. La mayor precisión alcanzada es 81,4 %, obtenida con regresión logística y TF-IDF sin reducción de dimensionalidad. Retno Kusumaningrum et al. (2020) compararon tuits y documentos en inglés e indonesio relacionados al retraso en el crecimiento, con el objetivo de distinguir entre aquellos escritos por expertos y aquellos escritos por personas comunes. Probaron tres estadísticas (frecuencia de términos, presencia de términos y TF-IDF) y cuatro técnicas (Regresión logística, *naive bayes*, bosques aleatorios y SVM). Los mejores resultados se dieron con SVM, alcanzando una precisión del 98 %.

Finalmente, se buscó especialmente trabajos que tuvieran como objetivo clasificar libros dentro de sus respectivos géneros. Shiroya et al. (2021) usaron dos conjuntos de sinopsis extraídas de Wikipedia de libros escritos en inglés, gujarati e hindi con el objetivo de clasificarlos según etiquetas obtenidas en otra base de datos. Utilizaron TF-IDF y contrastaron KNN, regresión logística y SVM. La mayor precisión que alcanzaron es 54 %, con SVM. Falk (2019) trabajó con un conjunto de libros obtenidos del sitio web Project Gutenberg, dividido en cuatro géneros. Utilizó términos individuales, bigramas y trigramas, evaluando escenarios con y sin lematización, normalización, reducción de dimensionalidad y eliminación de palabras vacías. Los algoritmos comparados fueron KNN y SVM, con la precisión promedio variando entre 25 % y 89 % según el escenario. Brigadoi (2021) también utilizó libros de Project Gutenberg, pero en este caso trabajó con ocho géneros posibles. Calculó frecuencias absolutas y TF-IDF, empleando términos individuales, bigramas de distintos tipos y análisis de sentimientos. Contrastó cuatro técnicas (*naive bayes*, KNN, SVM y bosques aleatorios). La mejor precisión (72 %) se dio para esta última técnica utilizando frecuencia absoluta de términos individuales. El mismo conjunto de datos fue utilizado por Yako (2021), quien calculó frecuencia de términos y TF-IDF para términos individuales y palabras que transmiten emociones (análisis de sentimientos). Contrastó KNN, SVM y bosques aleatorios, obteniéndo los mejores resultados con esta última técnica.

Mirończuk y Protasiewicz (2018) realizaron un estudio analizando trabajos recientes relacionados a la clasificación de textos. Según los autores, este proceso generalmente se resume en seis etapas básicas:

1. **Adquisición de datos**: Se recolectan los datos necesarios para responder los objetivos del problema bajo estudio, en caso de no contar ya con un conjunto de datos.

2. **Análisis de datos y etiquetado**: En el caso de aprendizaje supervisado, se le asignan una o más etiquetas a los textos (por ejemplo, el o los géneros al que pertenecen o su autor), formando así grupos de documentos. Dicho procedimiento se omite en el caso de aprendizaje no supervisado. En esta etapa se incluye también el preprocesamiento de los textos, utilizando un conjunto de herramientas que buscan eliminar elementos de poca utilidad para el análisis, separan las oraciones en unidades de menor extensión (segmentación) o identifican la raíz de cada palabra (lematización), entre otras actividades.

3. **Construcción de atributos y ponderación**: Como en todo método estadístico, se debe contar con un conjunto de variables (también llamadas atributos) que describan a las observaciones. En el análisis de textos, las variables no se observan o miden de forma directa, sino que se construyen a partir de los textos. Por ejemplo, cada palabra de nuestro idioma puede potencialmente ser tratada como una variable, con valores numéricos determinados por la cantidad de veces que la misma

aparece en cada texto o por alguna otra medida de su importancia en el documento.

4. **Reducción de dimensionalidad**: Debido a que la cantidad de atributos construida suele ser muy elevada, trabajar con todos ellos implicaría importantes requerimientos computacionales de tiempo y memoria. Por ello, en esta etapa se reduce el número de variables mediante algún criterio que permita retener aquellas de mayor importancia.

5. **Entrenamiento del modelo de clasificación**: El conjunto de datos es dividido en dos subconjuntos, denominados conjunto de entrenamiento y conjunto de prueba. Se utiliza el primero de estos subconjuntos para entrenar un algoritmo de aprendizaje automático en la tarea de clasificación deseada. De esta manera, se obtiene un modelo de clasificación.

6. **Evaluación del modelo**: Finalmente, el modelo es aplicado en el conjunto de prueba. Para evaluar su desempeño, se seleccionan uno o más indicadores de evaluación (tales como precisión global o área bajo la curva) y se estiman y comparan los valores obtenidos.

En este trabajo se construyó un clasificador que permite catalogar lo mejor posible a un conjunto de libros de ficción dentro de sus géneros literarios correspondientes. Se probaron distintas combinaciones de conjuntos de atributos y enfoques para intentar hallar el que devuelva los mejores resultados. La técnica de aprendizaje supervisado elegida fue el algoritmo de bosques aleatorios, una extensión de los métodos basados en árboles de decisión que ha mostrado muy buen desempeño en tareas de clasificación (Fernández-Delgado et al., 2014). Un resumen de los resultados obtenidos al aplicar bosques aleatorios en los trabajos de la revisión bibliográfica realizada puede encontrarse en el Anexo 1.

Esta tesina se encuentra organizada de la siguiente manera: en el Capítulo 2 se detallan más minuciosamente los objetivos. El Capítulo 3 explica la metodología utilizada y se divide en cuatro secciones. La primera de ellas presenta el conjunto de datos que se emplea, la técnica por la cual se obtuvo dicho conjunto y el proceso de etiquetado. La segunda sección está dedicada al análisis textual, detallando el preprocesamiento de los documentos, la construcción de atributos y creación de matrices de documentos-términos. La siguiente sección habla sobre el aprendizaje supervisado, desarrollando las técnicas de árboles de decisión y bosques aleatorios, como así también la evaluación e interpretación de los modelos. Finalmente, la cuarta sección describe el *software* utilizado. Los resultados del trabajo se exponen en el Capítulo 4. Finalmente, el Capítulo 5 consiste en una discusión de dichos resultados, las conclusiones finales del trabajo y sugerencias para futuras investigaciones.

2. Objetivos

El objetivo principal de esta tesina es construir un clasificador que permita categorizar automáticamente a un texto de ficción en su correspondiente género literario, eligiendo entre un conjunto predeterminado de posibles clases. Para esto se utilizan herramientas de minería de textos y de aprendizaje supervisado.

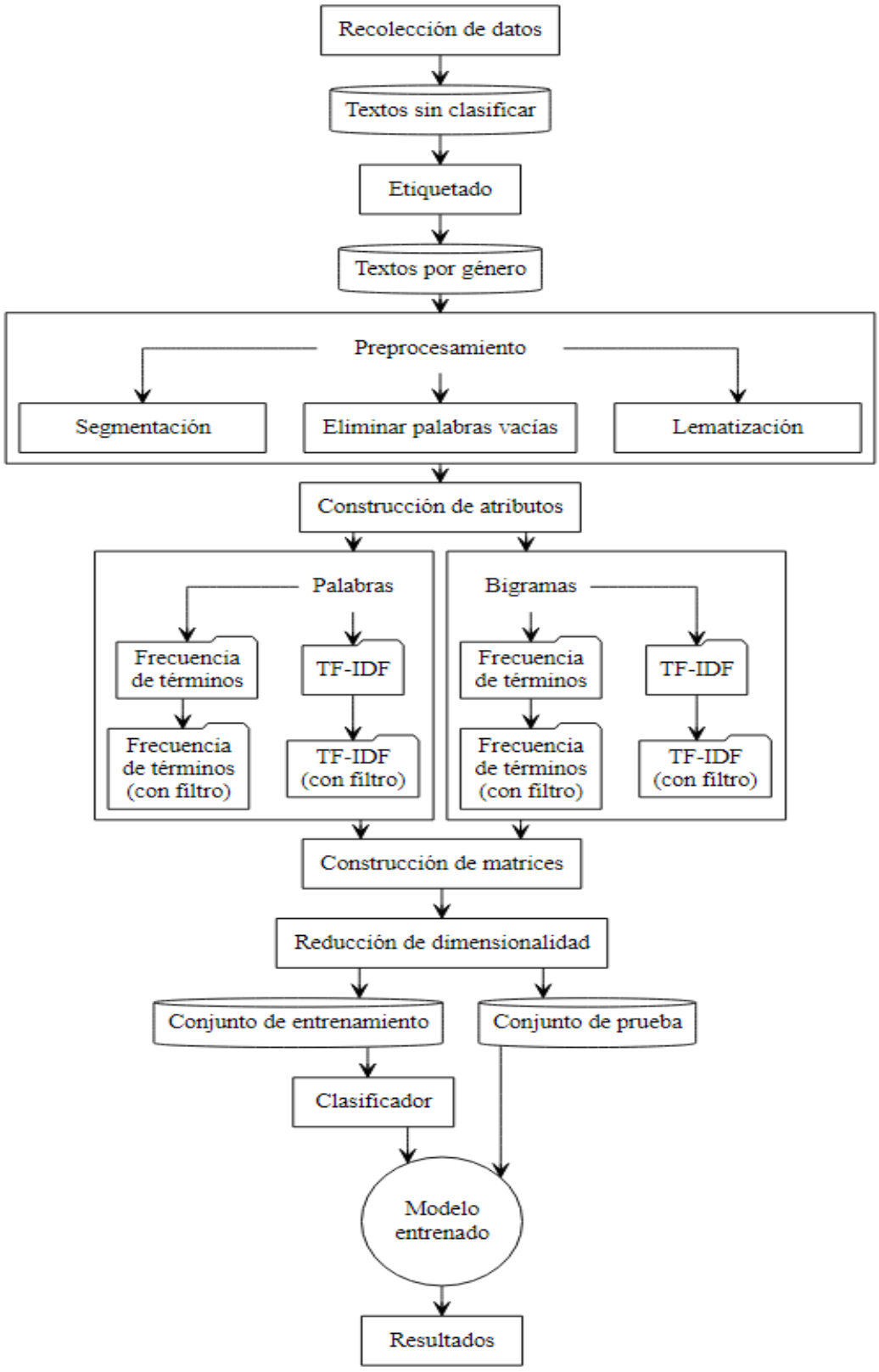
Como objetivos específicos se plantean:

- Crear un conjunto de datos conformado por libros en español pertenecientes a diversos géneros.
- Construir distintos tipos de atributos para representar a cada documento y determinar cuál de ellos produce una mejor precisión en la clasificación.
- Hallar las palabras o pares de palabras más representativas para cada género literario.
- Evaluar si el modelo planteado produce resultados satisfactorios y, de no hacerlo, explorar enfoques alternativos para obtener un mejor desempeño.
- Encontrar qué géneros resultan mejor clasificados por el algoritmo y cuáles resultan frecuentemente confundidos con otros.
- Calcular la importancia de las distintas variables empleadas para ver cuáles son más relevantes para el proceso de clasificación.

3. Metodología

En esta sección se describe la metodología utilizada a lo largo de todo el trabajo. La Figura 1 muestra un resumen en forma gráfica de las etapas que se siguieron.

Figura 1: Resumen de la metodología utilizada



3.1. Conjunto de datos

3.1.1. Selección de datos

Los datos utilizados están conformados por libros de ficción escritos en español o traducidos a dicho idioma. Puede tratarse de novelas, cuentos o recopilaciones de historias cortas. Se excluyen obras en otros idiomas o aquellos textos que no correspondan al tipo narrativo (textos periodísticos o científicos, manuales de instrucción, etc.).

Existen múltiples sitios web que permiten descargar gratuitamente libros en español pertenecientes al dominio público, entre ellos se encuentran Elejandría (https://www.elejandria.com/) y Freeditorial (https://freeditorial.com/es). Se recurrió únicamente a sitios web que afirmen explícitamente que sus textos se encuentran libres de derechos de autor, para asegurar que sea legal trabajar con sus contenidos.

3.1.2. Descarga de datos

En la mayoría de las páginas consultadas se aplicó la técnica de "raspado web" (del inglés *web scraping*) mediante el paquete *rvest* del programa R para obtener los enlaces de descarga de los textos del corpus. Esta técnica consiste en leer el código HTML de una página web y extraer algunos de los elementos deseados de ella en forma automática utilizando algún *software*. A continuación, los documentos fueron descargados desde los enlaces aplicando funciones provistas por R. Finalmente, se convirtieron en archivos .PDF a aquellos que originalmente estaban almacenados en otros formatos (por ejemplo, .DOC o .EPUB).

Hubo ciertas páginas donde aplicar raspado web no fue posible. Algunos sitios bloquean el uso de programas que aplican esta técnica, mientras que en otros no se puede obtener una única URL que contenga los enlaces de descarga. En estos casos fue necesario descargar los documentos de forma manual.

La lista completa de fuentes desde donde se descargaron los datos, así como la cantidad de textos obtenidos de cada una de ellas se presentan en la Sección 4.1.

3.1.3. Etiquetado

En base a la observación de las páginas web como las anteriormente mencionadas, se llegó a la conclusión de que la mayoría de los textos literarios podrían clasificarse en uno de los siguientes siete géneros: Aventura, Ciencia ficción, Fantasía, Ficción histórica, Policial o misterio, Romance y Terror o suspenso. Se considera que estos géneros están lo suficientemente bien definidos para poder ser más tarde clasificados automáticamente en forma satisfactoria.

La mayoría de los sitios de descargas ya presentan a las obras etiquetadas según género, pero en

ciertos casos se presentaron dificultades adicionales. Hay casos en los que distintas páginas ofrecen el mismo libro pero éste es clasificado en géneros distintos. Por ejemplo, la novela *La Caza del Meteoro* de Julio Verne es categorizado como "ciencia ficción" en ciertas fuentes, mientras que otras lo categorizan como "aventura". Por otra parte, ciertos sitios no etiquetan los textos según su género, haciendo necesario asignarle uno manualmente.

Para resolver estos inconvenientes se decidió recurrir al sitio web Goodreads (disponible en el siguiente enlace: https://www.goodreads.com/). Esta plataforma posee una base de datos de libros y permite a sus usuarios asignarles etiquetas a los mismos. Por consiguiente, el criterio utilizado para solucionar estas ambigüedades fue utilizar la etiqueta más votada por la comunidad, entendiendo que esto refleja el consenso general sobre la obra en cuestión. Por ejemplo, al ingresar en la entrada correspondiente a la novela *1984* de George Orwell, se observa que más de 11.500 usuarios lo clasificaron como "ciencia ficción", contra menos de 1.400 que lo clasificaron como "fantasía". Consecuentemente, este libro es asignado a la primera clase. De esta manera, se obtuvieron etiquetas para todos los documentos del corpus.

3.2. Análisis de textos

3.2.1. Preprocesamiento

Una vez que cada uno de los archivos estuvo transformado al formato .PDF y con una etiqueta asignada, se los guardó en distintas carpetas según su género correspondiente. Luego, los documentos fueron importados a R y convertidos en cadenas de texto para ser manipulados de manera más eficiente por el *software*. Antes de proceder con la construcción de los atributos, es necesario realizar una serie de procedimientos, que serán descriptos a continuación.

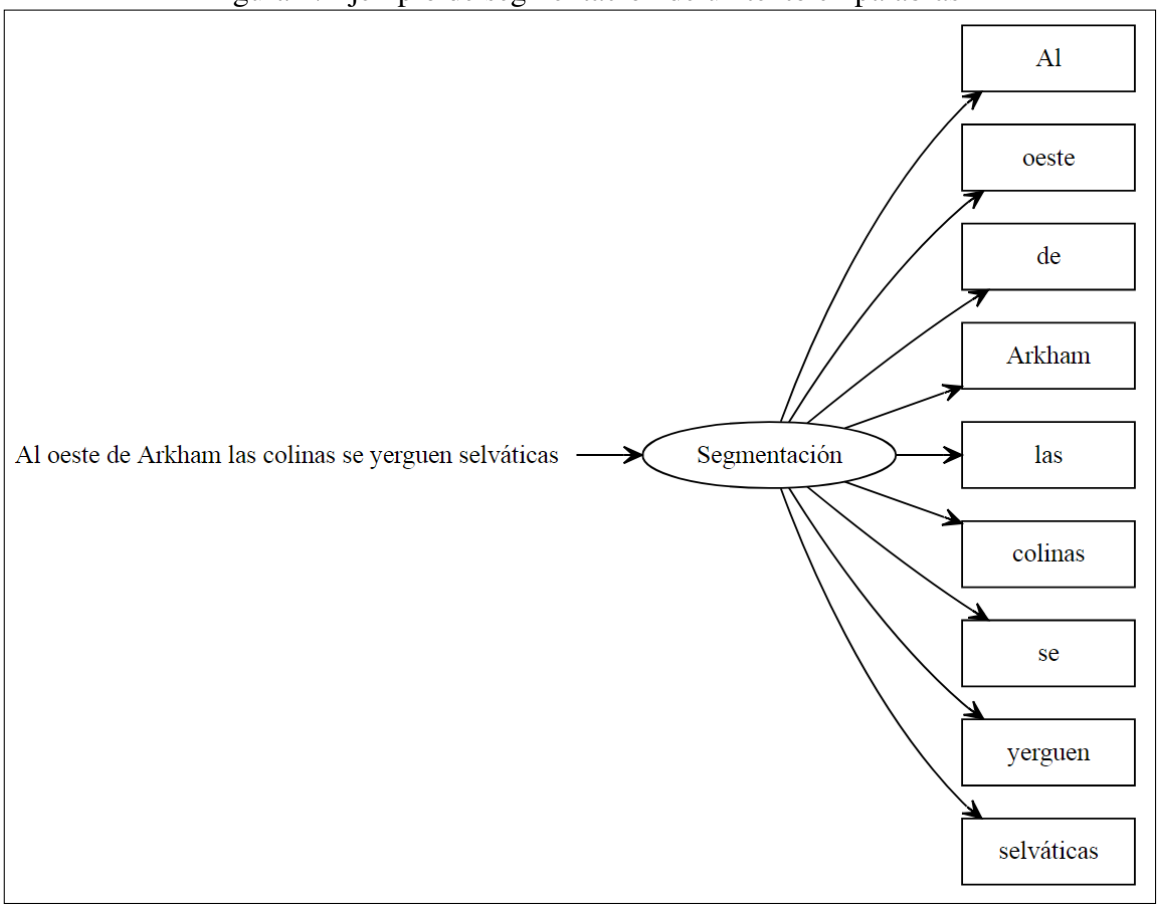
Conversión a minúsculas

Se convierten todas las mayúsculas presentes en el texto a minúscula, con el propósito de homogeneizar formatos de escritura.

Segmentación

Un **token** es una unidad significativa de texto (como, por ejemplo, una palabra) que se utiliza para el análisis (Sielge y Robinson, 2017). El proceso de dividir un texto en tókenes se denomina **tokenización** o **segmentación**. La Figura 2 muestra un ejemplo de un texto segmentado en palabras.

Figura 2: Ejemplo de segmentación de un texto en palabras



En algunos casos, resulta de interés estudiar las relaciones entre palabras, no solo entre términos individuales y su texto. Para ello, se puede considerar como token a un conjunto de palabras, por ejemplo una frase o un párrafo. En particular, el uso de **n-gramas** como tókenes puede brindar información relevante sobre un texto, al capturar más contexto sobre él. Un n-grama es una secuencia de n palabras consecutivas. Cuando $n = 2$, esta se denomina **bigrama**. El caso particular en el que $n = 1$ se denomina **unigrama** y es equivalente a trabajar con palabras individuales como tókenes. En esta tesina se opta por realizar una segmentación en unigramas y otra en bigramas, obteniendo así dos conjuntos de variables distintos que darán lugar más adelante a diferentes escenarios de análisis (Tabla 1 en la Sección 3.2.4).

Eliminación de palabras vacías

Las **palabras vacías** (también conocidas como *stopwords*) son palabras comunes que aportan poca o nula información valiosa (Hvitfeldt y Silge, 2022). Algunos ejemplos de palabras vacías en el idioma español incluyen artículos (como “el” y “la”), preposiciones (“con”, “de”) y conjunciones (“aunque”, “porque”). Éstos términos aparecen muy frecuentemente en los textos sin contribuir de manera relevante a su contenido y por ende no resultan útiles para discriminar entre categorías.

En consecuencia, las palabras vacías suelen ser removidas de los documentos antes de realizar una

tarea de clasificación de textos. Además de mejorar la capacidad discriminatoria del clasificador, este proceso permite reducir los tiempos de cómputo para la minería de textos y disminuir las dimensiones de los conjuntos de datos resultantes. Para filtrar las palabras vacías, es posible recurrir a listas de palabras previamente construidas. En esta tesina se utiliza una lista que incluye aproximadamente 700 palabras vacías.

Limpieza

Además de las palabras vacías, se eliminan otros caracteres considerados innecesarios. Estos incluyen signos de puntuación, números, símbolos, caracteres pertenecientes a otros alfabetos (por ejemplo, letras griegas) y enlaces web que se encuentran incrustados en los archivos PDF seleccionados.

Lematización

Los textos trabajados siempre presentan diferentes variantes de una misma palabra, como "organizo", "organiza" y "organizando". Si bien los tres casos vienen de la misma palabra ("organizar") y su significado es prácticamente el mismo, las computadoras los interpretan como términos distintos. Por consiguiente, antes de la construcción de las variables, es habitual buscar una forma de convertir todas las variantes de un mismo término a una única palabra "base". Otro resultado positivo de realizar esto es que se reduce el volumen total de datos, pues ahora todas las formas de una palabra contarán como distintas ocurrencias de una única variable.

El proceso de lexematización (muchas veces llamado por su nombre en inglés, *stemming*) consiste en reducir una palabra o conjunto de palabras a su lexema o raíz, es decir la parte de una familia de palabras que se mantiene invariable. Por ejemplo, el conjunto {*moderno*, *moderna*, *modernos*, *modernizará*, *modernizó*} se convierte en {*modern*, *modern*, *modern*, *modern*, *modern*}. Existen diversos métodos para llevar a cabo esta tarea, pero generalmente se utilizan algoritmos que eliminan sufijos comunes (Baeza-Yates y Ribeiro-Neto, 2011). Las raíces resultantes no suelen ser palabras reales.

Un proceso similar consiste en la **lematización**. Lematizar una palabra consiste en encontrar su lema correspondiente, es decir, la forma en la que aparece en un diccionario o enciclopedia. Por ejemplo, se escriben los sustantivos en su forma masculina y singular y los verbos en infinitivo. El conjunto mencionado anteriormente se convertiría en {*moderno*, *moderno*, *moderno*, *modernizar*, *modernizar*}. Para la lematización, se puede recurrir a modelos probabilísticos, cadenas de Markov o diccionarios (Porras, 2020). A diferencia de las raíces, los lemas siempre serán palabras reales.

Los algoritmos de lexematización suelen ser más simples y rápidos de implementar, pero también tienen peores resultados que los algoritmos de lematización en tareas de clasificación (Hvitfeldt y Silge, 2022). Para esta tesina, se experimentó con diversas metodologías, evaluando los resultados

obtenidos y los tiempos de cómputo. Finalmente, se decidió emplear un diccionario de lemas creado por el lingüista Michal Měchura y publicado en un repositorio online con licencia abierta (https://github.com/michmech).

3.2.2. Filtrado de nombres propios

Debido a que se trabaja con libros de ficción, es de esperar que entre las palabras con mayor frecuencia aparezcan nombres propios, correspondientes a personajes o lugares de las obras literarias. Esto podría llevar a que el algoritmo adopte criterios que arrojen buenos resultados en el conjunto de datos provisto, pero que resulten ineficaces en nuevos datos (efecto conocido como "sobreajuste"). Puede ser de interés evaluar si eliminar los sustantivos propios produce diferencias importantes al momento de realizar la clasificación con respecto a un conjunto de datos en el que no se eliminan.

Clark (2018) empleó una base de datos disponible online para remover los nombres de los personajes de las obras de William Shakespeare en su análisis. Sin embargo, debido a que en este trabajo se utilizan obras de numerosos autores, no existe una única base a la que se pueda recurrir. Como resultaría una tarea muy ardua evaluar cada texto por separado para detectar nombres propios, se optó por recurrir al Diccionario de la lengua española (Real Academia Española, 2014) para detectar términos que figuran en dicha publicación y eliminar los que no. Una desventaja de este enfoque es el hecho de que existen nombres propios que son además sustantivos comunes o adjetivos (por ejemplo: "Angelina", "Valentino" o "Clara"), por lo que no serán filtrados.

Este filtro se aplica tanto al conjunto de unigramas como al de bigramas mencionados anteriormente, creando dos conjuntos de variables adicionales.

3.2.3. Construcción de atributos

Luego del preprocesamiento de los datos, es necesario construir los atributos o variables que serán utilizados por el algoritmo de aprendizaje para clasificar los textos. Existen diversas maneras de sintetizar la información provista por un documento.

Frecuencia de término

Una de las estadísticas que puede utilizarse es la **frecuencia de término** (*term frequency*). Esta se trata simplemente de la frecuencia absoluta de un determinado término en un documento. La frecuencia absoluta del término t en un documento d se representa como $\mathrm{tf}_{t,d}$. Calcular este valor para cada palabra en un documento permite representarlo como un vector numérico, enfoque conocido como **bolsa de palabras** (en inglés, "*bag of words*"). No se toma en cuenta el orden exacto de las palabras en el texto,

solamente el número de ocurrencias de las mismas. Es decir que los textos "Juan es más veloz que María" y "María es más veloz que Juan" serán representados exactamente de la misma manera bajo este enfoque. Por consiguiente, se trabaja con el supuesto de que los documentos con representaciones similares son semejantes en contenido (Manning et al., 2019).

Estadística TF-IDF

Otro estrategia posible consiste en calcular la frecuencia inversa del documento (*inverse document frequency* o idf) de los términos. La idf de un término t se calcula como:

$$\text{idf}_t = \ln\left(\frac{\text{nº total de documentos}}{\text{nº de documentos que contienen el término t}}\right).$$

Por lo tanto, esta medida le asignará un valor bajo a términos muy frecuentes, mientras que un término infrecuente recibirá un valor alto. El menor valor posible se dará en el caso en el que un término aparezca en todos los documentos del corpus (Manning et al., 2019).

Al combinar la frecuencia de términos con la frecuencia inversa del documento se obtiene la estadística **frecuencia de término – frecuencia inversa de documento** (más conocida como TF-IDF por sus siglas en inglés *term frequency – inverse document frequency*). Se utiliza para medir qué tan importante es una palabra respecto a un documento dentro de un corpus (Sielge y Robinson, 2017). El peso que se le asigna al término t en el documento d es:

$$\text{tf-idf}_{t,d} = \text{tf}_{t,d} \times \text{idf}_t.$$

Esta estadística le asigna una mayor ponderación a términos que aparecen mucho en un número reducido de documentos, mientras que aquellos términos con baja frecuencia o que aparezcan en muchos textos (es decir, palabras muy comunes) recibirán menores pesos.

Para obtener las palabras más representativas de cada género, se puede calcular esta estadística considerando como "documento" al conjunto de todos los libros de cada uno de los siete géneros (en lugar de cada texto individual). Esto se debe a que en este caso el interés es obtener las palabras más representativas del conjunto y no de cada obra por separado.

Extensión para n-gramas

Las estadísticas antes mencionadas pueden ser calculadas también para n-gramas con $n > 1$, para lo cual simplemente se trata a la secuencia de n palabras como un único término y se procede como ya se describió (Sielge y Robinson, 2017).

3.2.4. Matriz de documentos-términos

Una **matriz de documentos-términos** (del inglés *document-term matrix*, también llamada matriz de términos del documento, matriz de términos por documento o simplemente DTM) se define como una matriz de dimensiones $m \times n$, siendo m el número de documentos del corpus y n el número total de términos. En otras palabras, cada fila de la matriz corresponde a un documento mientras que cada columna corresponde a un token. El valor de cada celda (i, j) corresponde entonces a $\text{tf}_{i,j}$, el número de veces que aparece el término de la columna j en el documento de la fila i. Alternativamente, se puede utilizar el valor TF-IDF en lugar de la frecuencia absoluta en cada celda (Sielge y Robinson, 2017).

En función de los tres apartados anteriores, se cuenta con 8 variantes para la construcción de la matriz, los cuales generan los 8 escenarios que serán analizados a lo largo de esta tesina (Tabla 1).

Tabla 1: Escenarios de análisis

Escenario	**N-grama**	**Estadística**	**Filtro de nombres propios**
A	Unigramas	Frecuencia de término	No
B	Bigramas	Frecuencia de término	No
C	Unigramas	TF-IDF	No
D	Bigramas	TF-IDF	No
E	Unigramas	Frecuencia de término	Sí
F	Bigramas	Frecuencia de término	Sí
G	Unigramas	TF-IDF	Sí
H	Bigramas	TF-IDF	Sí

Cada una de las 8 matrices es utilizada posteriormente para la construcción de un clasificador. Se contrastan los resultados para determinar en cuál de los escenarios el modelo presenta un mejor desempeño.

3.2.5. Reducción de dimensionalidad

Las matrices de documentos-términos suelen ser **matrices dispersas** (también conocidas como matrices ralas), es decir que presentarán grandes dimensiones y la mayoría de sus elementos serán iguales a cero (Sielge y Robinson, 2017). Esto se debe a que generalmente hay muchos tókenes que aparecen solo en uno o unos pocos documentos, haciendo que la columna correspondiente a esa variable tenga casi todas sus celdas iguales a cero. Utilizar las matrices completas para el aprendizaje automático demandaría por lo tanto una gran cantidad de memoria y elevados tiempos de cómputo. En consecuencia, antes

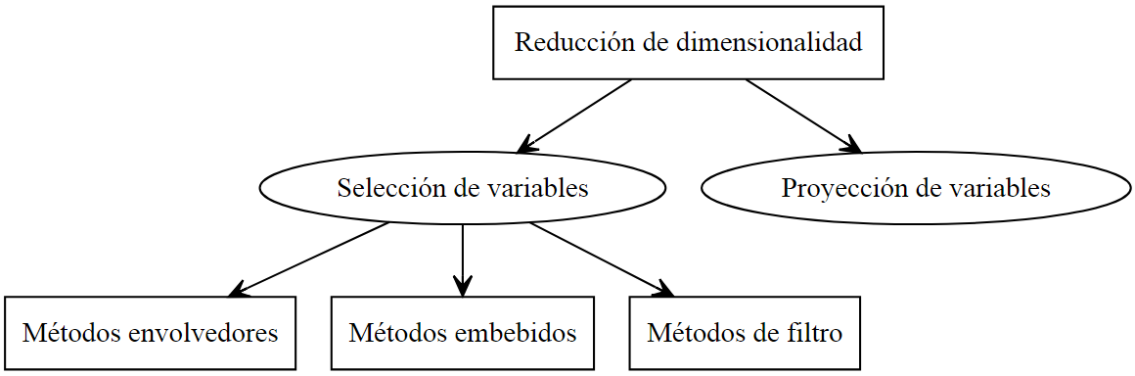
de proceder a la etapa de entrenamiento del modelo, se acostumbra a buscar una técnica de **reducción de dimensionalidad**, es decir que se intenta disminuir la cantidad de atributos y por ende el número de columnas de la matriz utilizada.

Existen diferentes tipos de reducción de dimensionalidad. El más utilizado en el contexto de clasificación de textos es la **selección de variables** (*feature selection*), definido como el proceso que busca el tamaño mínimo de atributos textuales relevantes para optimizar el error de clasificación de los textos (Kou et al., 2020). En otras palabras, las variables más descriptivas son retenidas mientras que el resto son descartadas. Una alternativa consiste en utilizar técnicas de proyección de variables (*feature projection*), tales como análisis de componentes principales o análisis discriminante lineal (Mirończuk y Protasiewicz, 2018).

Los métodos de selección de variables se pueden a su vez dividir en tres tipos (Figura 3):

- Métodos envolvedores (*wrapper*): utilizan algoritmos para encontrar y seleccionar subconjuntos de variables.
- Métodos embebidos (*embedded*): seleccionan las variables que más contribuyen a la precisión de un modelo al mismo tiempo que éste es entrenado.
- Métodos de filtro: se le asigna a cada variable una medida de su relevancia y se las ordena de mayor a menor. Luego se retienen las n primeras variables o se eliminan las n últimas, siendo el valor de n determinado empíricamente. Estos son los métodos más usados, debido a que son más simples y requieren menos tiempo de cómputo que los métodos envolvedores o embebidos (Kou et al., 2020). Para medir la relevancia de las variables pueden utilizarse coeficientes de correlación, prueba chi-cuadrado o ANOVA, entre otros.

Figura 3: Distintos tipos de reducción de dimensionalidad



En esta tesina, se aplica una metodología similar a la empleada por Yako (2021) que se describe a continuación:

1. Primero, se suman los valores de tf o TF-IDF de cada token a través de todos los géneros (obteniendo un único valor para todo el corpus en lugar de valores para cada género).
2. Se reordena este conjunto en orden descendente según la estadística utilizada.
3. Se grafican los datos (colocando el orden del término en el eje de las abscisas y el valor de su estadística en el eje de las ordenadas) y en base a su observación se selecciona una cota (cantidad de términos que al ser superada no muestre variaciones importantes en la estadística).
4. Se retienen las variables para las cuales el valor de su estadística es igual o superior a dicha cota.
5. Finalmente, se repiten los pasos anteriores para los ocho escenarios propuestos previamente.
6. Opcionalmente, se experimenta con distintos valores para la cota con el propósito de evaluar si existe una mejora importante en el desempeño del modelo.

3.3. Aprendizaje supervisado

Se denomina **aprendizaje estadístico** a una serie de herramientas que permiten interpretar conjuntos complejos de datos (James et al., 2021), intentando extraer patrones y tendencias importantes con el objetivo de entender mejor a los datos (Kuhn y Johnson, 2016). Generalmente se buscan relaciones entre una variable Y (llamada *output*, variable respuesta o variable dependiente) y variables X_j (llamadas atributos, *input*, variables predictoras o variables independientes). Los problemas de aprendizaje estadístico suelen dividirse generalmente en dos grandes categorías: **aprendizaje supervisado** y **aprendizaje no supervisado**.

Las técnicas de aprendizaje no supervisado se aplican en el caso en el que, para cada observación $i = 1...n$, se cuenta con un vector de variables predictoras $\mathbf{x}_i$ pero no se tiene una valor asociado de la variable respuesta y_i. Por consiguiente, no es posible aplicar métodos tales como regresión lineal, pues no se tiene una variable que predecir. El objetivo de estas técnicas es entonces encontrar relaciones entre las variables independientes o entre las observaciones, para describir como se organizan o agrupan los datos. Entre los métodos de aprendizaje no supervisado se encuentran distintas técnicas de análisis de conglomerados o *clusters*, análisis de componentes principales y algoritmos para la detección de anomalías.

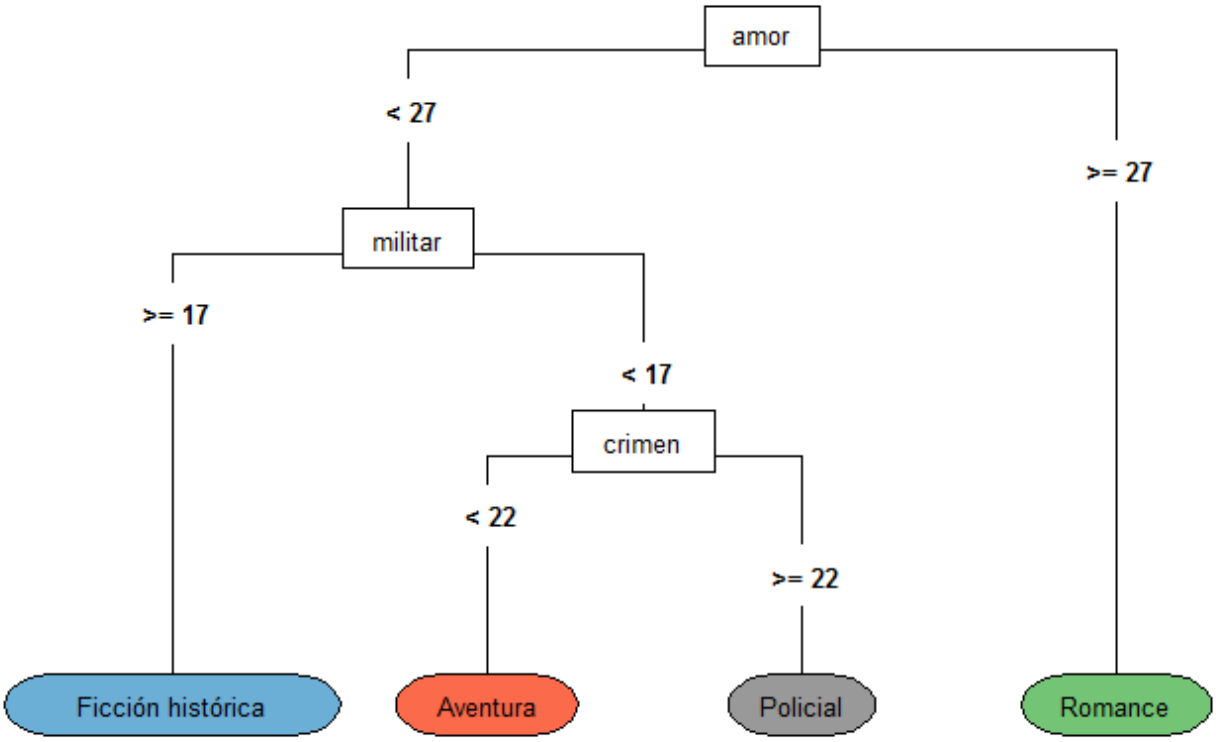
Por otro lado, en el aprendizaje supervisado se cuenta con la presencia de una medida de la variable respuesta y_i para cada una de las observaciones, lo que permite guiar el proceso de aprendizaje. El

objetivo de aplicar estos métodos es ajustar un modelo que permita predecir adecuadamente valores para las futuras observaciones o entender mejor la relación entre la variable dependiente y sus predictores. Algunos métodos pertenecientes al dominio del aprendizaje supervisado incluyen modelos de regresión lineal o logística, redes neuronales, método de k vecinos más próximos, clasificador bayesiano ingenuo y métodos basados en árboles. Estos últimos son los utilizados en el presente trabajo.

3.3.1. Árboles de decisión

Los modelos basados en árboles son una clase de algoritmos no paramétricos que funcionan particionando el espacio de las variables predictoras en subregiones no superpuestas, agrupando individuos con valores similares de la variable respuesta según una serie de reglas de partición. Para hacer una predicción para una observación determinada, generalmente se usa la media o moda de la variable respuesta calculada entre las observaciones de entrenamiento ubicadas en la misma región. Debido a que este conjunto de reglas pueden resumirse gráficamente en forma de árbol, estas técnicas se conocen como métodos de **árboles de decisión** (James et al., 2021).

Figura 4: Esquema de un árbol de decisión



La Figura 4 muestra un ejemplo de un árbol de decisión, aplicado para identificar a qué género pertenece un determinado documento considerando el número de ocurrencias de los términos “amor”,

"crimen" y "militar". Las regiones en las que se divide el espacio de predictoras se denominan **nodos terminales** o, por analogía con un árbol real, **hojas**. En el ejemplo, éstas corresponden a los distintos géneros literarios y se las representa con óvalos de colores. El **tamaño** de un nodo es la cantidad de observaciones en él. Los puntos en el árbol en donde las variables se bifurcan se conocen como **nodos internos** y en el gráfico están representados como rectángulos blancos. Cuanto más arriba aparezca el nodo, mayor es la importancia de la variable para la clasificación. Finalmente, los segmentos que conectan los nodos entre ellos son las **ramas** del árbol. El máximo número de ramas entre el primer nodo y una hoja se conoce como **profundidad** del árbol, siendo esta igual a tres en el ejemplo.

El primer algoritmo de árbol de clasificación fue publicado por Robert Messenger y Lewis Mandell en 1972, denominado THAID (del inglés *THeta Automatic Interaction Detection*). Desde entonces se han propuesto diversas metodologías con mayor poder predictivo, tales como CHAID y C4.5. Actualmente el algoritmo más popular es **CART** (del inglés *Classification and regression trees*), presentado por primera vez por Leo Breiman en 1984. Este algoritmo utiliza el Índice de Gini como criterio para realizar las particiones, medida que se calcula como:

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

donde $\hat{p}_{mk}$ es la proporción de observaciones del conjunto de entrenamiento en la m-ésima región que pertenecen a la k-ésima clase. El índice de Gini tomará entonces menores valores cuando todas las $\hat{p}_{mk}$ tengan valores cercanos a 0 o a 1, por lo que se la considera una medida de la "*pureza*" de la clase.

Para comenzar la construcción del árbol, se evalúa cada una de las variables explicativas $X_1, X_2, ..., X_p$ y se consideran todos los puntos de corte posibles. Se calcula G para cada una de las dos subregiones resultantes y ambos valores se suman ponderando por la cantidad de observaciones en cada subregión. El criterio para definir la bifurcación es entonces minimizar esta suma ponderada, lo que equivale a conseguir regiones más "puras". Este proceso se repite para cada nodo resultante hasta alcanzar algún criterio de parada, el cual usualmente consiste en que cada nodo terminal alcance un determinado tamaño mínimo. Debido a que el algoritmo busca la mejor bifurcación en cada nodo en lugar de buscar el mejor resultado final, se lo conoce como algoritmo voraz o *greedy* ("codicioso" en inglés).

Entre las ventajas de utilizar árboles de decisión se encuentra su robustez frente al ruido (variables poco importantes), bajo costo computacional y habilidad de manejar atributos redundantes. Poseen alta flexibilidad, pudiendo ser usados para tareas de clasificación o de regresión y trabajar con variables continuas y discretas. Además, son fáciles de graficar e interpretar, ya que su estructura refleja el pensamiento humano en forma más natural que otras técnicas (Jena y Dehuri, 2019).

Entre sus desventajas se encuentra el hecho de que los árboles de decisión suelen tener menor precisión predictiva que otras técnicas de aprendizaje supervisado (James et al., 2021). En ciertos casos

también pueden ser muy sensibles frente a cambios en los datos, es decir, pequeñas variaciones en el conjunto de observaciones pueden resultar en criterios de partición muy diferentes, generando árboles muy variables entre sí y haciendo las interpretaciones poco confiables. A esta característica se la denomina **inestabilidad** (Hastie et al., 2017). Para superar estas desventajas, han surgido técnicas que permiten combinar los resultados de distintos árboles y obtener resultados más precisos tales como *bagging*, bosques aleatorios y *boosting*.

3.3.2. Bosques aleatorios (*Random Forests*)

La técnica de **bagging** (abreviatura del inglés *Bootstrap aggregating*), consiste en tomar B muestras sucesivas del conjunto de datos de entrenamiento y construir un árbol distinto con cada una de ellas. A continuación, para una determinada observación, se registra la clase predicha por cada uno de los B árboles y se recurre al "voto de la mayoría": la categoría más frecuente a la cual fue asignada dicha observación es considerada como categoría final. Este método permite obtener mayor precisión y menor variabilidad respecto a los árboles de decisión individuales, con la desventaja de perder la capacidad de graficar el procedimiento y de una mayor dificultad en la interpretación (James et al., 2021).

Una mejora respecto a esta técnica se obtiene al aplicar la metodología de **bosques aleatorios** (del inglés *Random Forest*), cuyo algoritmo más utilizado fue planteado por primera vez por Leo Breiman en 2001. Así como en *bagging*, se toman B muestras sucesivas del conjunto de entrenamiento mediante *bootstrap* (método de remuestreo que utiliza muestreo simple al azar con reemplazo). La diferencia consiste en que la metodología de bosques aleatorios propone considerar solamente un subconjunto aleatorio de m variables predictoras para cada bifurcación en lugar del conjunto completo de p predictoras. Es decir que mientras que en *bagging* todos los árboles están construidos con el mismo subconjunto de variables, los bosques aleatorios utilizan distintos subconjuntos en cada uno de los B árboles.

El razonamiento detrás de esta estrategia es el siguiente: si bien *bagging* introduce cierta aleatoriedad al utilizar *bootstrap*, esta a menudo es insuficiente. Si entre las variables predictoras existe una que tenga un efecto muy relevante, dicha predictora casi siempre estará en el primer corte. Esto ocasionará que todos los árboles posean una estructura muy similar entre sí, especialmente en la parte alta, característica conocida como **correlación entre árboles**. Como solución, los bosques aleatorios introducen un nuevo elemento de aleatoriedad, al no permitirle al modelo elegir entre todas las variables predictoras disponibles para el corte sino a un subconjunto de m variables. Las estructuras de los árboles construidos serán consecuentemente más diversas pues no siempre las mismas variables aparecerán en la parte superior, eliminando así la correlación entre árboles. La reducción en la variabilidad respecto a emplear árboles individuales resulta entonces mayor con bosques aleatorios que con *bagging*, lo que deriva en menores

tasas de error y conclusiones más robustas y confiables (Kuhn y Johnson, 2016).

En síntesis, el algoritmo de bosques aleatorios para clasificación puede resumirse en los siguientes pasos (Hastie et al., 2017):

1. Tomar una muestra *bootstrap* del conjunto de entrenamiento.
2. Construir un árbol de decisión aplicando los siguientes pasos:
   a) Seleccionar al azar m variables del conjunto de p variables predictoras.
   b) Elegir el mejor predictor entre las m variables.
   c) Dividir los datos en dos nodos.
   d) Repetir los pasos a), b) y c) hasta que cada nodo terminal alcance un determinado tamaño mínimo (por defecto 1).
3. Obtener predicciones para las observaciones a partir del árbol construido.
4. Repetir los pasos 1 a 3 en B oportunidades, tomando como predicción final el "voto de la mayoría" calculado a partir de las B predicciones.

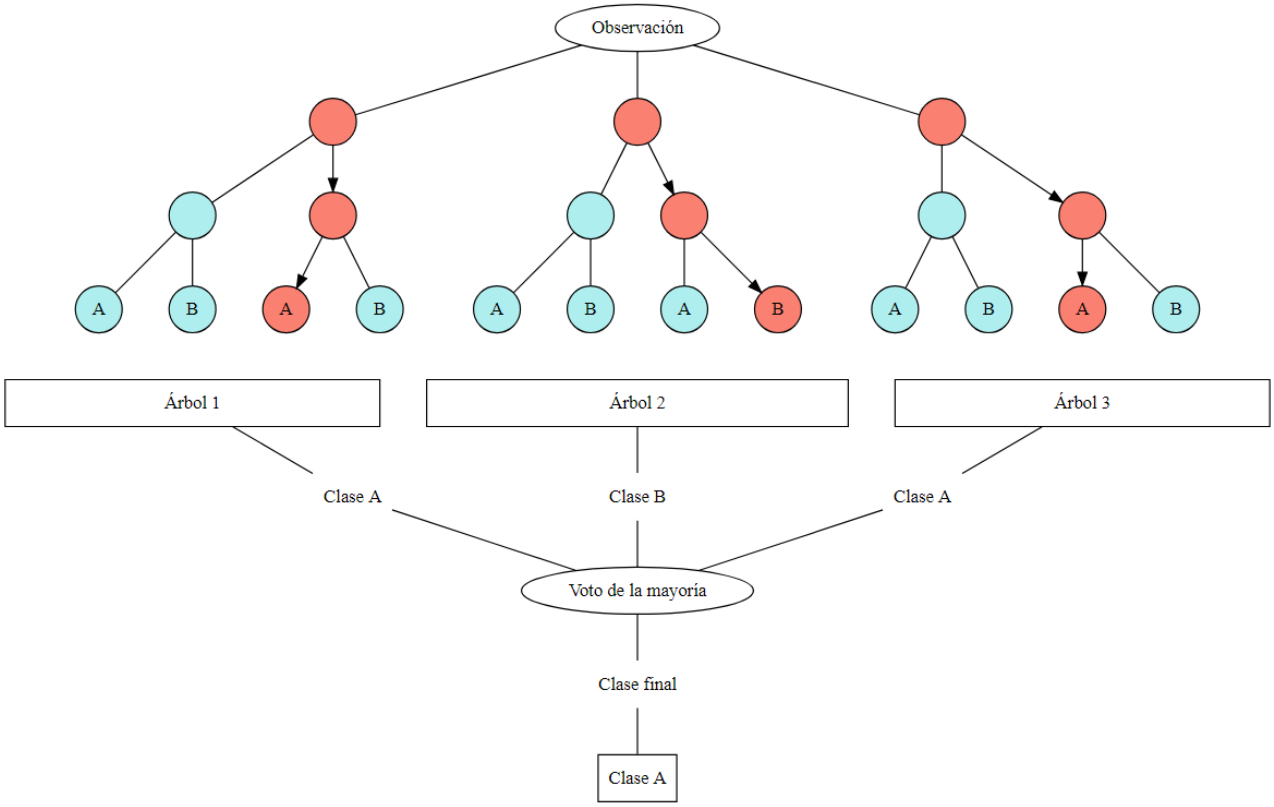
Un resumen gráfico de este procedimiento se presenta en la Figura 5.

Como se mencionó anteriormente, la técnica de bosques aleatorios provee generalmente mejores resultados predictivos que utilizar árboles individuales o *bagging*. Al igual que este último, posee la desventaja de perder interpretabilidad, dado que ahora no se cuenta con un único árbol para graficar, complicando también la determinación de la importancia de las variables (Sección 3.3.5). Asimismo, los tiempos de cómputo pueden aumentar sustancialmente, sobre todo si se cuenta con grandes conjuntos de datos. Sin embargo, gracias a que los árboles son ajustados de manera independiente (a diferencia de lo que pasa con otras técnicas como *boosting*), es posible aplicar computación paralela para optimizar estos tiempos.

Los bosques aleatorios poseen diversos hiperparámetros que pueden ser ajustados o tuneados (del inglés *tuning*) por el usuario. Probst et al. (2019) identifican seis hiperparámetros, aunque indican que los valores provistos por defecto por los distintos *software* suelen dar resultados satisfactorios. Sin embargo, múltiples autores afirman que hay dos parámetros principales que se recomienda tunear puesto que resulta en un mejor desempeño del modelo. Estos son:

- El **número B de árboles** que se construyen (llamado generalmente *ntree* en programas estadísticos): valores más altos de este hiperparámetro producen una disminución en el error de clasificación hasta cierto punto, a partir del cual éste se estabiliza y deja de disminuir. Seguir aumentando

Figura 5: Esquema de un algoritmo de bosques aleatorios



este parámetro no produce sobreajuste, pero sí ocasiona tiempos de cómputo innecesariamente más altos. Los valores más usuales para este hiperparámetro son $B = 500$ o $B = 1000$.

- La **cantidad** m **de predictores** que se consideran en cada división (llamado generalmente *mtry* en programas estadísticos): elegir valores bajos de este parámetro produce árboles más diversos y menos correlacionados, lo que resulta en mejor estabilidad como se mencionó en la Sección 3.3.2. Sin embargo, también puede ocasionar que el rendimiento del modelo empeore, pues al elegir entre menos variables, pueden ser seleccionadas algunas con una importancia muy baja, llevando a árboles con menor precisión en promedio. Se debe entonces lidiar con un dilema entre la estabilidad y la precisión de los árboles individuales.

  Por otra parte, si se trabaja con conjuntos de datos con muchas variables en los cuales solo unas pocas son importantes, es preferible elegir valores grandes de m pues es más probable que las variables relevantes sean seleccionadas. En el caso contrario (muchas variables de gran importancia), es preferible elegir valores pequeños para permitir que las variables de influencia moderada tengan posibilidades de ser seleccionadas y su efecto no sea enmascarado por las que tengan mayor influencia. En el caso de árboles de clasificación, el valor que suele tomarse por defecto es $m = \sqrt{p}$.

En esta tesina, se probaron distintas combinaciones de ambos hiperpárametros, incluyendo los valores por defecto previamente mencionados (Tabla 2). Para los demás hiperparámetros, se mantienen los valores predeterminados por la función *randomForest* del paquete de R del mismo nombre. Se utilizó validación cruzada de K iteraciones (*K-fold cross-validation*) para determinar la mejor combinación de B y m en cada uno de los escenarios.

Tabla 2: Valores de B y m que se contrastan

Hiperparámetro	Valores
Número de árboles (B)	$\{500, 1000\}$
Cantidad de predictores por división (m)	$\{100, 500, 1000, 1500, 2000, 3000, 3500\}$

3.3.3. *Boosting*

La técnica de *Boosting* es una extensión de *bagging*, alternativa a bosques aleatorios. Así como este, se basa en la construcción de varios árboles de decisión. Pero a diferencia de lo que ocurre en bosques aleatorios, en *boosting* los árboles son construidos de manera secuencial y no independiente. Los árboles "aprenden" de los errores de los anteriores al asignarle un peso a cada observación y modificarlo en cada paso. Las observaciones correctamente clasificadas mantienen su peso, mientras que el de aquellas clasificadas erróneamente se modifica. Existe evidencia de que al aplicar *boosting* se puede alcanzar una mayor precisión que al utilizar bosques aleatorios (James et al., 2021), razón por la cual se decidió estudiar su desempeño en la presente tarea de clasificación.

Se han propuesto diversos algoritmos de *boosting*, los cuales con el tiempo han agregado parámetros y se han vuelto más complejos, pero que han logrado mejores resultados. El más popular actualmente y el que se describe en esta sección es *XGBoost* (Chen y Guestrin, 2016). Los hiperparámetros de *XGBoost* que se suelen tunear son los siguientes:

- **nrounds** (valor por defecto igual a 100): número de árboles construidos. Valores muy grandes llevan a sobreajuste, a diferencia de lo que pasa con bosques aleatorios.
- **eta** (valor entre 0 y 1, por defecto igual a 0,3): define en cuánto disminuyen los pesos de las variables en cada paso, se usa para prevenir sobreajuste y hacer el modelo más robusto. Menores valores deben estar acompañados por un aumento en nrounds.
- **max_depth** (por defecto 6): profundidad máxima del árbol, mayores valores hacen al modelo más complejo y pueden llevar a sobreajuste.

- **gamma** (por defecto 0): mínima reducción en la función de pérdida necesaria para seguir particionando los nodos. Un valor mayor de gamma resulta en un algoritmo más conservador. Produce mejores resultados cuando max_depth es bajo.

- **min_child_weight** (por defecto 1): si una partición resulta en un nodo con la suma de los pesos de las observaciones menor que este parámetro, se detiene el proceso de partición. Es decir que se utiliza para prevenir relaciones muy específicas que pueden llevar a sobreajuste, pero valores muy altos producen un mal ajuste.

- **subsample** (valores entre 0 y 1, por defecto igual a 1): fracción de las observaciones que serán seleccionadas al azar para cada árbol.

- **colsample_bytree** (valores entre 0 y 1, por defecto igual a 1): fracción de las columnas que serán seleccionadas para la construcción de cada árbol del bosque.

En esta tesina, se partió de los valores por defecto de cada hiperparámetro y se fueron probando en forma secuencial distintos valores hasta obtener los mejores resultados en el conjunto de prueba. Los hiperparámetros que se probaron se presentan en la Tabla 3.

Tabla 3: Selección de hiperparámetros para *boosting*

Hiperparámetro	Valores
nrounds	{50,100,200,500,600,800,1000}
max_depth	{1,3,6,50,200,300}
eta	{0.01,0.05,0.1,0.3,0.4}
gamma	{0,0.01,1,5}
colsample_bytree	{0.5,0.6,0.8,0.9}
subsample	{0.5,0.75,0.8,1}
min_child_weight	{0,1,5}

3.3.4. Evaluación del modelo

Frecuentemente se describe el desempeño de un modelo mediante una **matriz de confusión**, como se ve en la Tabla 4. En este tipo de matrices, las celdas diagonales indican el número de observaciones que fueron correctamente clasificadas, mientras que los elementos fuera de la diagonal señalan las ocurrencias de cada tipo de error posible (Kuhn y Johnson, 2016).

Tabla 4: Esquema de una matriz de confusión

		Predicción del modelo		
		Clase 1	**Clase 2**	**Clase 3**
Clase observada	**Clase 1**	Bien clasificado	Mal clasificado	Mal clasificado
	Clase 2	Mal clasificado	Bien clasificado	Mal clasificado
	Clase 3	Mal clasificado	Mal clasificado	Bien clasificado

A partir de esta matriz pueden calcularse diversas métricas. La más simple de éstas es la **precisión global**, calculada como el porcentaje de observaciones correctamente clasificadas (esto es, la suma del total de los elementos diagonales dividido por el total de observaciones). Una desventaja de utilizar esta métrica es que no tiene en cuenta las frecuencias relativas de las clases en el conjunto de datos. Es decir que si hay una clase mucho más frecuente que las demás, la precisión de un modelo podría aumentar simplemente al asignar casi todas las observaciones a esta clase.

Una métrica que considera las frecuencias relativas de cada clase es la **estadística Kappa**, presentada por Jacob Cohen en 1960 y originalmente utilizada para medir el grado de acuerdo entre dos observadores. En el contexto de los modelos de clasificación, se utiliza para evaluar el acuerdo entre las predicciones del modelo y los datos realmente observados. El coeficiente Kappa se calcula como:

$$K = \frac{O - E}{1 - E}$$

donde O es la precisión observada y E es la precisión esperada según los totales marginales de la matriz de confusión, en otras palabras, la probabilidad esperada de aciertos que se observaría si el modelo clasificara las unidades al azar. Esta estadística puede variar entre -1 y 1, aunque muy pocas veces se observan valores negativos. Un valor de 0 indica una desacuerdo total entre las categorías predichas por el modelo y las observadas en el conjunto de pruebas, mientras que un valor equivalente a 1 indica un acuerdo perfecto. Si bien no existe un consenso acerca de qué valores de esta estadística son considerados aceptables, generalmente se concluye que existe un acuerdo al menos moderado entre las predicciones del modelo y los datos observados cuando K está entre $0,3$ y $0,5$ (James et al., 2021).

3.3.5. Importancia de las variables

Como se mencionó anteriormente, una desventaja de usar técnicas como *bagging* o bosques aleatorios es que se pierde la facilidad en la interpretación que proveen los árboles de decisión. En particular,

no es clara la determinación de la importancia de las variables en el procedimiento, pues ya no se puede representar el procedimiento en un único árbol (James et al., 2021).

Es posible obtener una medida de la importancia de una variable empleando el índice de Gini (presentado en la Sección 3.3.1). Para ello se calcula la reducción de G en cada bifurcación de esa variable en el bosque, luego se suman dichos valores y se promedia por la cantidad B de árboles. Cuanto mayor sea el valor resultante, mayor es la importancia de la variable para la clasificación. Posteriormente, las medidas de importancia pueden ser representadas gráficamente (por ejemplo, en un gráfico de barras) con el fin de obtener una interpretación más sencilla.

3.3.6. Otros enfoques para la clasificación de textos

La tarea de clasificar textos de ficción ha tenido menor desarrollo que la de clasificar otros tipos de texto, debido a que es considerada más compleja (Ward y Saarti, 2018). Las razones de esta dificultad se deben a que los textos de ficción suelen contener una amplia variedad de temas y conceptos, mientras que otros documentos tratan por lo general un único tema bien definido. En otras palabras, no existe un conjunto objetivo de reglas que permita clasificar a una obra dentro de un único género, por lo que una persona puede considerar como correcta a una clasificación que otra persona consideraría incorrecta (Jordan, 2014). Además de esto, los textos de ficción a menudo presentan sus ideas de manera intencionalmente compleja, vaga y abierta a diversas interpretaciones (Falk, 2019). Es decir que para comprender el contenido de una obra, es necesario realizar un esfuerzo de interpretación y pensamiento abstracto por parte del lector.

Las computadoras no son capaces de realizar dichas interpretaciones, lo cual sumado a la subjetividad en la determinación de un género, hacen especialmente difícil que puedan aprender a clasificar apropiadamente documentos de ficción. Por ello, los modelos suelen mostrar un peor desempeño en comparación al obtenido en la clasificación de textos que no sean de ficción, como se pudo observar en la revisión bibliográfica en la introducción. Entre las múltiples soluciones que se han planteado para mejorar la clasificación de textos de ficción, una de las más aceptadas es utilizar un enfoque multidimensional. Esto es, en lugar de asignarle una única etiqueta al documento, es posible asignarle más de una en función del contenido del mismo y de las necesidades del usuario.

Según Wang y Chiang (2011), los problemas de clasificación de textos pueden dividirse en "problemas de una etiqueta" (cuando el documento solo puede pertenecer a una clase) y "problemas multietiqueta" (cuando se le asignan múltiples etiquetas al mismo documento). Esta tesina fue abordada como un problema de una etiqueta, pues cada texto solo podía pertenecer a un género. Sin embargo, dadas las dificultades intrínsecas de trabajar con textos de ficción mencionadas en esta sección, se decidió

experimentar con dos enfoques más flexibles para investigar si se producía una mejora en los resultados.

Enfoque multi-etiqueta

Se le asignan dos etiquetas a cada texto en lugar de una sola, utilizando nuevamente información del sitio web Goodreads para ello. A cada texto se le asignan las dos etiquetas más votadas en los casos donde sea posible. Algunas obras solo presentaron una etiqueta en el sitio web, por lo que no se les pudo asignar una segunda. Se aplican los bosques aleatorios calculados anteriormente para el enfoque de una etiqueta, pero esta vez la clasificación es considerada "correcta" si la clase predicha por el modelo corresponde a cualquiera de las dos asignadas al documento (en lugar de solo la primera).

Múltiples predicciones

En lugar de extraer del modelo una única clase predicha por observación, se pueden calcular las probabilidades por clase. La clase con mayor probabilidad es la que anteriormente era tomada como única clase predicha. Ahora, se considera que una observación está bien clasificada si su clase observada corresponde con alguna de las dos primeras clases predichas por el modelo. Así como antes, se aplican los bosques aleatorios construidos previamente y se calcula la precisión global.

3.4. Software

En el presente trabajo se utilizó el software R (versión 4.1.3). Se recurrió a la colección de paquetes *tidyverse* (Wickham et al., 2019) para el tratamiento, transformación y visualización de los datos.

Los textos fueron descargados e importados al software mediante los paquetes *rvest* (Wickham, 2022), *pdftools* (Ooms, 2022) y *gutenbergr* (Robinson, 2021). El análisis textual se realizó con los paquetes *tidytext* (Sielge, 2016), *corpus* (Perry, 2021), *tm* (Feinerer y Hornik, 2020), *stopwords* (Benoit, Muhr y Watanabe, 2021) y *stringi* (Perry, 2021).

Se construyeron las matrices de términos con el paquete *quanteda* (Benoit et al., 2018). Para la programación de los clasificadores, los paquetes usados fueron *rpart* (Therneau y Atkinson, 2022) y *randomForest* (Liaw y Wiener, 2002); a su vez, se utilizó *caret* (Kuhn, 2022) para la validación cruzada y el ajuste de hiperparámetros. Se agilizaron estos procesos mediante computación paralela, utilizando el paquete *doParallel* (Microsoft Corporation y Weston, 2022).

Finalmente, para los gráficos presentes en el trabajo se utilizaron los paquetes *ggplot2* (Wickham, 2016), *RColorBrewer* (Neuwirth, 2022), *ggwordcloud* (Le Pennec y Slowikowski, 2019), *igraph* (Csardi y Nepusz, 2006), *ggraph* (Pedersen, 2021), *rpart.plot* (Milborrow, 2021), *plotly* (Sievert, 2020), *DiagrammeR* (Iannone, 2022) y *cowplot* (Wilke, 2020). El código utilizado en este trabajo puede consultarse en el siguiente enlace: https://github.com/SGS2000/tesina-bosques-aleatorios.

4. Resultados

4.1. Análisis descriptivo

En esta sección se lleva a cabo un análisis descriptivo del conjunto de datos construido. Visitando nueve páginas distintas, se logró adquirir 746 documentos. La distribución de libros según sitio web se presenta en la Tabla 5. Luego de la recolección de datos, se procedió a etiquetar los textos.

Tabla 5: Cantidad de documentos obtenidos por sitio web

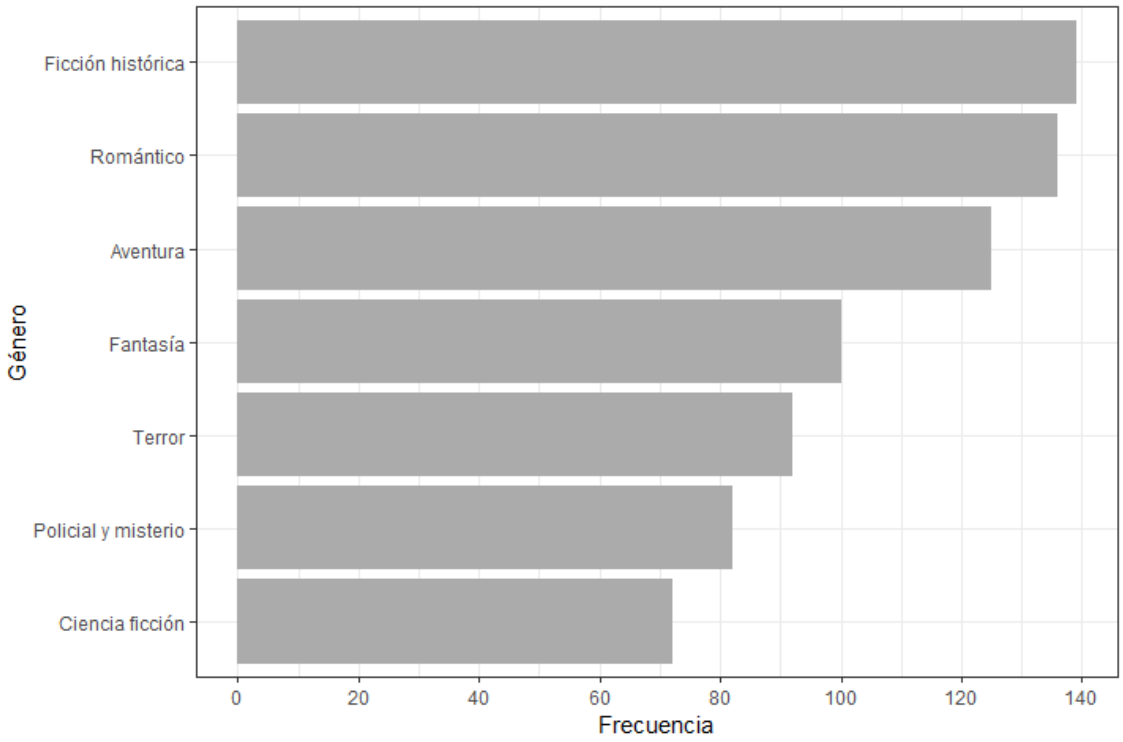
Sitio web	**URL**	**Nº de textos**
Elejandria	elejandria.com	311
Freeditorial	freeditorial.com/es	223
Ganso y Pulpo	gansoypulpo.com	129
Ciudad Seva	ciudadseva.com	30
Project Gutenberg	gutenberg.org/browse/languages/es	18
textos.info	textos.info	13
One More Library	onemorelibrary.com/index.php/en/languages/spanish	9
Dominio Público	dominiopublico.org.es	7
PlanetaLibro	planetalibro.net	6
Total		**746**

La categoría más frecuente en el conjunto de datos fue “ficción histórica” (139 textos o 18,6 % del corpus), seguida de “romance” (136 textos o 18,2 % del corpus). La categoría menos frecuente fue “ciencia ficción” (72 textos o 9 % del corpus), seguida de “policial y misterio” (82 textos o 10 % del corpus). La distribución completa de los textos según clase se presenta en la Tabla 6.

Tabla 6: Distribución de textos según género

Género	**Frecuencia absoluta**	**Frecuencia relativa**
Aventura	125	16,76 %
Ciencia ficción	72	9,65 %
Fantasía	100	13,40 %
Ficción histórica	139	18,63 %
Policial y misterio	82	10,99 %
Romántico	136	18,23 %
Terror	92	12,33 %
Corpus completo	**746**	**100 %**

Figura 6: Número de textos según género



A continuación, se aproximó el número de palabras de los documentos .PDF, tomando en cuenta solo los caracteres de texto e ignorando números y símbolos. Se contó la cantidad de palabras por género y luego se calcularon promedios, medianas, desvíos estándar, mínimos y máximos, presentados en la Tabla 7. Se puede apreciar que "ficción histórica" fue el género con más palabras, lo cual se debe en parte a que también es la categoría con más textos. Sin embargo, a pesar de no ser la clase con menor cantidad de documentos, "terror" fue la clase que menos palabras presentó. Es decir que, a pesar de haber

menos libros de ciencia ficción y misterio, estos fueron en promedio más extensos que los de terror, cómo se puede ver en la Figura 7.

Figura 7: Número promedio de palabras (en miles) según género en el corpus

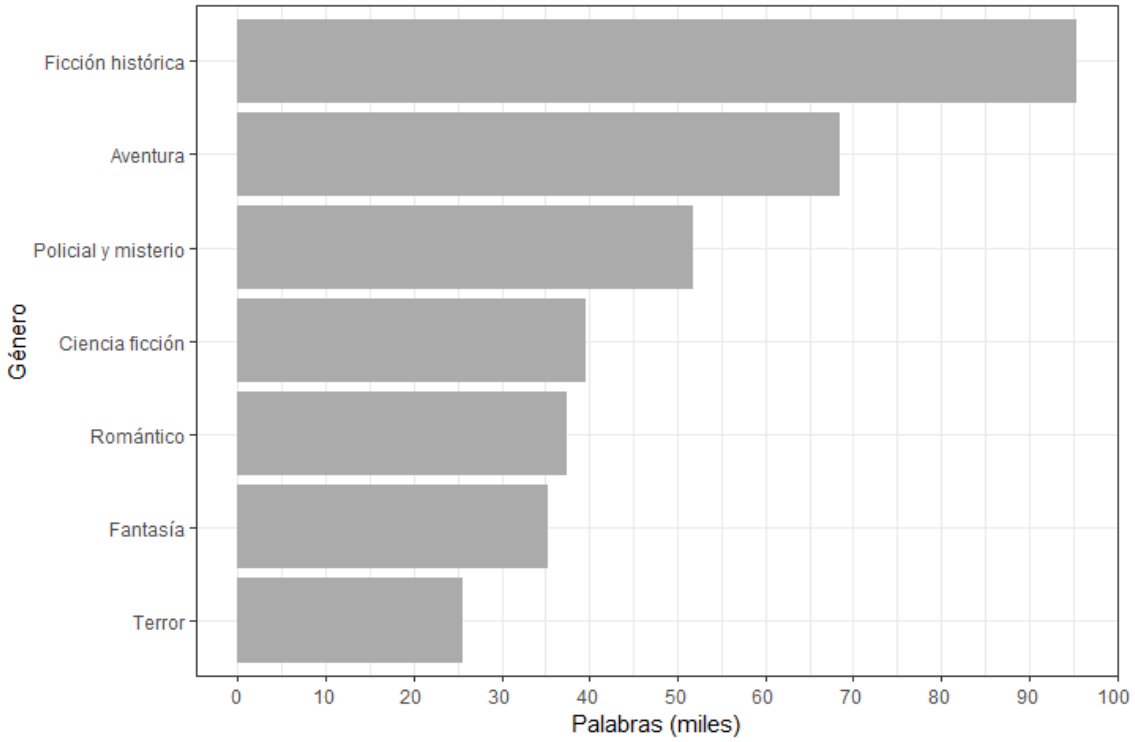


Tabla 7: Estadísticas descriptivas por género

Género	**Nº de textos**	**Palabras**					
		Nº	Media	D.E.	Mediana	Mínimo	Máximo
Aventura	125	8.548.283	68.386	50.625	57.143	1.270	381.315
Ciencia ficción	72	2.851.668	39.607	37.519	38.384	799	144.676
Fantasía	100	3.525.387	35.254	59.674	12.786	452	475.752
Ficción histórica	139	13.249.705	95.322	71.153	75.042	2.736	445.705
Policial y misterio	82	4.252.101	51.855	52.947	46.743	1.236	234.513
Romántico	136	5.079.807	37.352	53.138	3.058	620	222.491
Terror	92	2.346.928	25.510	42.289	6.671	870	280.387
Corpus completo	746	39.853.879	53.423	59.818	43.094	452	475.752

4.2. Análisis de los atributos construidos

En esta sección se examinan las variables que se construyeron para servir como predictoras en el clasificador. El análisis se realiza por separado para los escenarios con unigramas y para aquellos con bigramas (ver Tabla 1).

Unigramas (escenarios A, C, E y G)

Se obtuvieron 394.357 términos individuales distintos, siendo este número reducido a 172.934 en los escenarios donde se aplicó el filtro de nombres propios (E y G). La Tabla 8 muestra la cantidad de palabras únicas por género, tanto para los escenarios sin filtrar (A y C) como en aquellos filtrados. La suma de estas valores no resulta igual al total, debido a que hay palabras que se repiten entre los géneros.

Tabla 8: Cantidad de términos únicos por género, para escenarios con y sin filtrar

Género	Nº de términos (sin filtro)	Nº de términos (con filtro)	% de reducción
Aventura	75.054	27.767	63,0 %
Ciencia ficción	47.550	23.276	51,0 %
Fantasía	44.973	22.620	49,7 %
Ficción histórica	94.721	31.981	66,2 %
Policial y misterio	44.177	22.477	49,1 %
Romántico	52.738	25.248	52,1 %
Terror	35.144	19.565	44,3 %

Se construyeron nubes de palabras para los términos más representativos de cada escenario. Para los escenarios A y E se seleccionan aquellas palabras con mayor frecuencia absoluta, representando con mayor tamaño a los términos más frecuentes. Para los escenarios C y G se utilizan los términos con mayor valor de la estadística TF-IDF. En estos casos, la intensidad de los colores corresponden a la frecuencia del término dentro del género (mayor intensidad indica mayor frecuencia), mientras que su tamaño ahora indica un mayor valor de TF-IDF. En la Figura 8 se presentan las nubes para los libros de ciencia ficción a modo de ejemplo. Las nubes correspondientes al resto de los géneros se encuentran en el Anexo 2.

Figura 8: Nubes de palabras correspondientes a escenarios con unigramas para documentos pertenecientes al género ciencia ficción

Se observa que los escenarios A y E parecen tener las mismas palabras. Esto se debe a que entre los términos con mayor frecuencia no había ningún nombre propio y por lo tanto no fue filtrado, sería necesario graficar más palabras para apreciar las diferencias entre escenarios. Por su parte, las palabras del escenario C parecen ser en su mayoría nombres o apellidos correspondientes a los personajes de las obras, como “Winston” o “Rick”. Dichos términos son filtrados en el escenario G, permitiendo ahora visualizar palabras distintivas del género de la ciencia ficción como “galaxia” o “marciano”.

Bigramas (escenarios B, D, F y H)

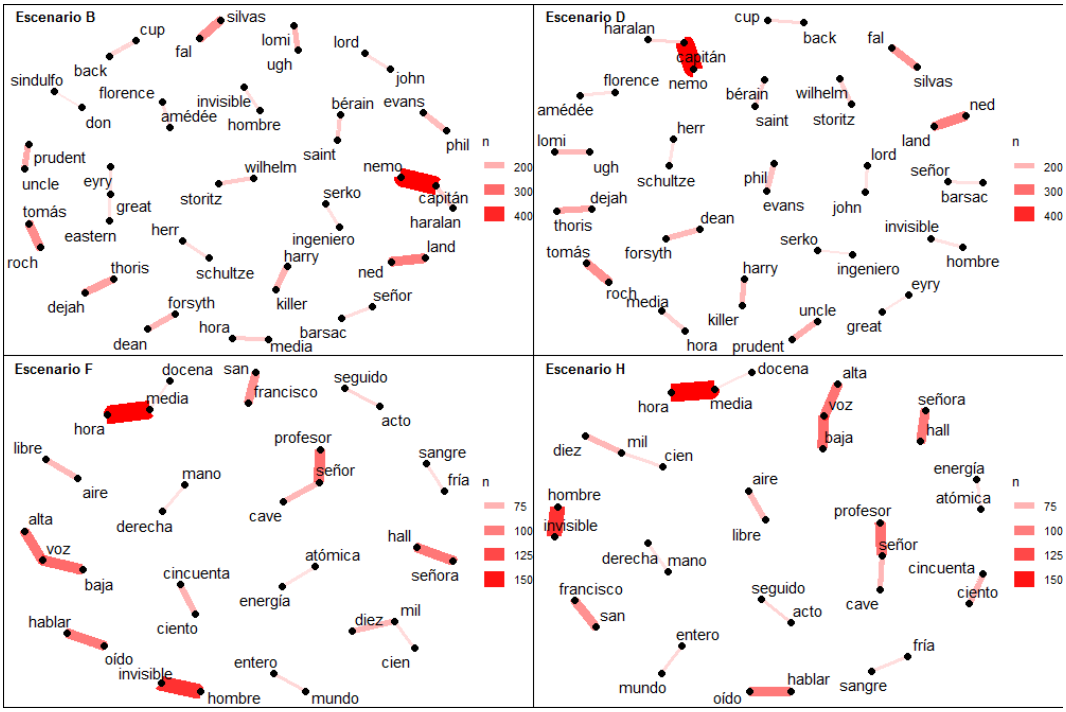
Se obtuvieron 334.634 bigramas distintos, siendo este número reducido a 202.202 en los escenarios donde se aplicó el filtro de nombre propios (F y H). La Tabla 9 muestra la cantidad de bigramas únicos por género, tanto para los escenarios sin filtrar (B y D) como en aquellos filtrados.

Tabla 9: Cantidad de bigramas únicos por género, para escenarios con y sin filtrar

Género	Nº de bigramas (sin filtro)	Nº de bigramas (con filtro)	% de reducción
Aventura	88.783	53.338	39,9 %
Ciencia ficción	22.089	13.330	39,6 %
Fantasía	32.836	20.057	38,9 %
Ficción histórica	88.258	51.883	41,2 %
Policial y misterio	40.634	25.329	37,6 %
Romántico	43.042	26.108	39,3 %
Terror	18.992	12.157	35,9 %

Para representar los bigramas más representativos por género, se recurrió a diagramas de red. Cada par de palabras unida por una línea roja es un bigrama presente en el conjunto de datos. Una misma palabra puede pertenecer a múltiples bigramas. El ancho de las líneas indica la frecuencia absoluta del bigrama (mayores frecuencias se indican con líneas más gruesas). Análogamente a lo efectuado con los unigramas, se seleccionan los bigramas con mayor frecuencia para los escenarios B y D y aquellos con mayor TF-IDF para los escenarios F y H. Los diagramas de red para los textos de ciencia ficción se presentan en la Figura 9. Los diagramas correspondientes a los demás géneros se encuentran en el Anexo 3.

Figura 9: Diagramas de red correspondientes a escenarios con bigramas para documentos pertenecientes al género ciencia ficción



A diferencia de lo ocurrido con los unigramas, ahora se aprecian diferencias más notorias entre los escenarios A y E. Por ejemplo, el bigrama “Capitán Nemo” desaparece en el segundo caso. Por el contrario, las diferencias entre los escenarios con frecuencia de términos y con TF-IDF son menos distinguibles que lo que ocurría en los escenarios con términos individuales. Aún así, es posible visualizar bigramas propios del género tales como “hombre invisible” o “energía atómica”.

4.3. Bosques aleatorios

Se probaron las combinaciones de hiperparámetros mencionadas en la Sección 3.3.2 para cada uno de los ocho escenarios y se utilizó validación cruzada para determinar la que resultara en una mayor precisión. La ejecución del código para seleccionar los mejores hiperparámetros demandó entre 4 y 14 horas dependiendo de cada escenario, utilizando un procesador Intel i7 de 8 núcleos y 16 GB de RAM.

Siguiendo el procedimiento indicado en la Sección 3.2.5, se buscó gráficamente una cota para poder reducir la dimensionalidad de las matrices. En todos los casos, se comenzó reteniendo los primeros 10.000 atributos, para luego probar con menores valores. Los únicos dos escenarios para los cuales tomar un menor número de variables resultó en una mejora en la precisión fueron los escenarios B y H, siendo 5.000 la cota seleccionada. Los resultados se presentan en la Tabla 10.

Tabla 10: Mejores combinaciones de hiperparámetros para cada escenario, utilizando validación cruzada

Escenario	**Nº total de variables (n)**	**Nº de árboles (B)**	**Nº de variables por bifurcación (m)**
A	10.000	1.000	1.500
B	5.000	1.000	500
C	10.000	1.000	1.500
D	10.000	1.000	1.500
E	10.000	500	3.000
F	10.000	500	500
G	10.000	500	500
H	5.000	500	3.500

Se aprecia que en los escenarios donde no se aplicó el filtro (A al D) fue necesario construir un mayor número de árboles, en comparación con aquellos escenarios donde se aplicó el filtro. Esto podría deberse a que, al no filtrar los términos, el efecto del sobreajuste es mayor y por lo tanto son necesarios más árboles para alcanzar una precisión más alta. El número de variables por bifurcación y el número de variables retenidas no parecen mostrar ningún patrón entre los escenarios.

Se construyeron los bosques aleatorios utilizando los hiperparámetros de la sección anterior. La ejecución del código para ajustar los árboles aleatorios demandó entre 11 minutos y 2 horas, dependiendo de cada escenario, con un tiempo promedio de 44 minutos. La precisión global y el coeficiente Kappa resultantes se muestran en la Tabla 11. Se presentan también los resultados obtenidos al ajustar un único árbol de decisión para cada escenario, para poder evaluar la mejora obtenida al aplicar bosques aleatorios.

Tabla 11: Precisión global y coeficiente Kappa observados para cada escenario, utilizando árboles de decisión y bosques aleatorios

	Árboles de decisión		Bosques aleatorios	
Escenario	Precisión	Kappa	Precisión	Kappa
A	43 %	0,33	64 %	0,57
B	41 %	0,30	51 %	0,41
C	30 %	0,16	59 %	0,51
D	37 %	0,24	50 %	0,39
E	36 %	0,23	64 %	0,58
F	35 %	0,21	48 %	0,38
G	43 %	0,33	52 %	0,42
H	37 %	0,24	47 %	0,36

Se observa que emplear bosques aleatorios resultó en un mejor desempeño respecto a utilizar árboles de decisión individuales. La precisión promedio con árboles de decisión fue 37,75 %, mientras que con bosques aleatorios fue 54,37 %. El coeficiente Kappa promedio también aumentó de 0,25 a 0,45.

Al aplicar bosques aleatorios, se observa que en seis de los ocho casos la precisión obtenida fue superior al 50 %. El coeficiente Kappa fue mayor a 0,3 en todos los casos, siendo superior al 0,5 en tres escenarios. Los mejores resultados se obtuvieron para el escenario E (unigramas filtrados y frecuencia de términos), seguido muy de cerca por el escenario A (unigramas no filtrados y frecuencia de términos). El escenario H (bigramas filtrados y TF-IDF) presentó los peores resultados.

Con la excepción de los escenarios A y E, en todos los demás casos se observa un peor desempeño en los escenarios donde se aplicó el filtro, respecto al escenario homólogo sin filtrar. Los escenarios con bigramas consistentemente presentaron peores resultados que aquellos con unigramas, la misma relación se observa para los escenarios con TF-IDF comparado a aquellos con frecuencia de términos. Se puede advertir también que la diferencia entre utilizar TF-IDF y frecuencia de términos es menor dentro de los

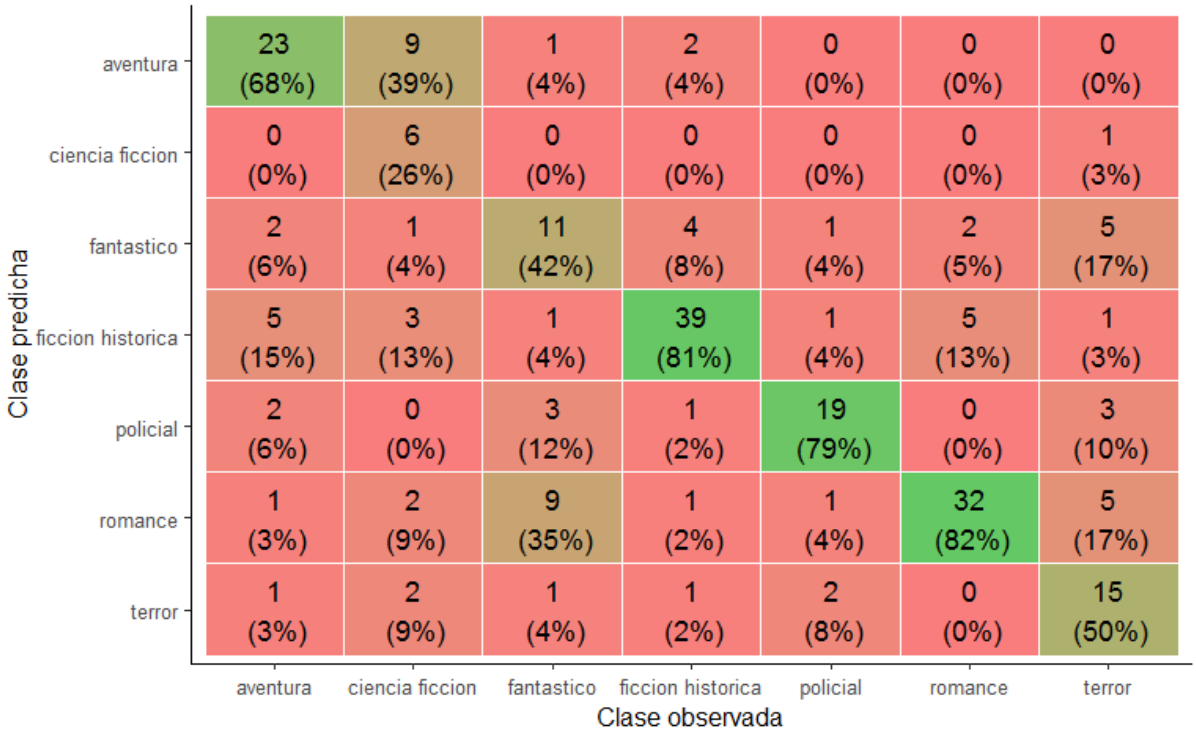
escenarios con bigramas (por ejemplo, entre los escenarios B y D) que entre los escenarios con unigramas (por ejemplo, los escenarios A y C). Para visualizar estas relaciones con mayor facilidad, se presentan las precisiones alcanzadas por escenario con bosques aleatorios en la Tabla 12.

Tabla 12: Precisión global según escenario utilizando bosques aleatorios

		Unigramas	Bigramas
Sin filtro	TF	64 %	51 %
	TF-IDF	59 %	50 %
Con filtro	TF	64 %	48 %
	TF-IDF	52 %	47 %

La Figura 10 muestra la matriz de confusión para el escenario que mostró los mejores resultados (escenario E). Las celdas muestran la frecuencia absoluta de cada combinación de categoría observada y categoría predicha por el modelo, con los porcentajes correspondientes a la clase observada (columnas). Los porcentajes de los elementos diagonales indican la precisión alcanzada por clase. Las matrices para los demás escenarios se presentan en el Anexo 4.

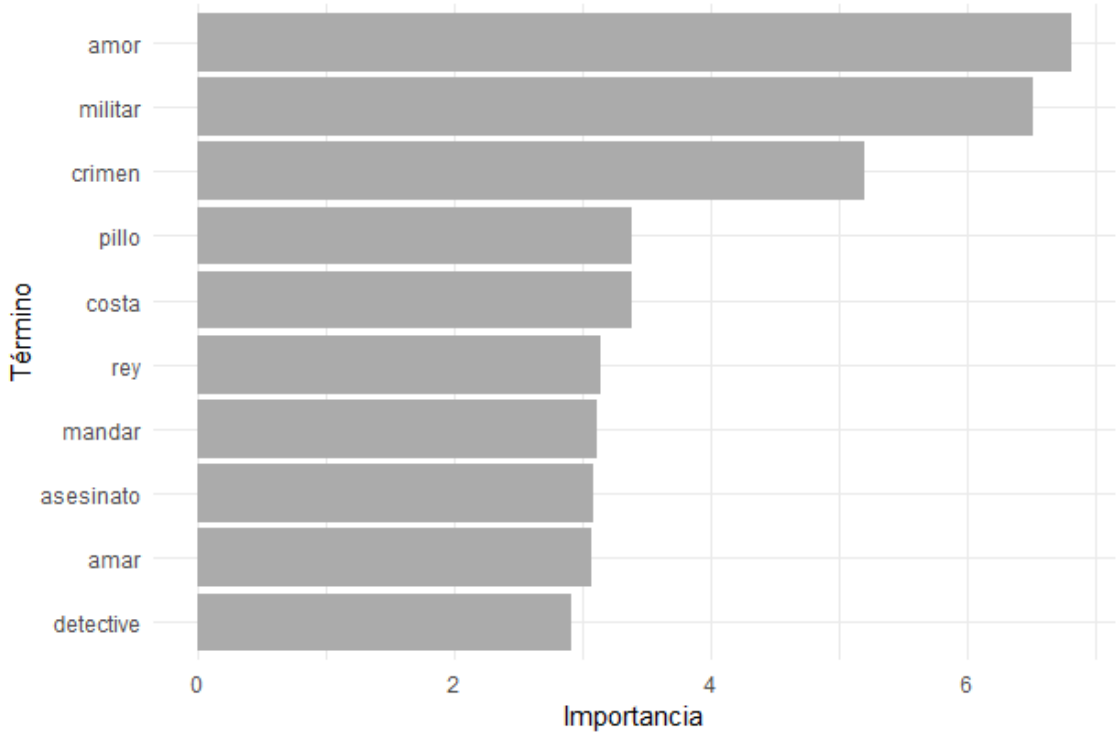
Figura 10: Matriz de confusión para el escenario E



Se aprecia que para las categorías “aventura”, “ficción histórica”, “policial y misterio” y “romance”, la precisión fue elevada, siendo superior al 68 % en todos los casos. La categoría con la menor precisión fue “ciencia ficción”, dentro de la cual solo 6 documentos (26 % del género) fueron correctamente clasificados. Las categorías más frecuentemente confundidas por el modelo fueron “ciencia ficción” con “aventura” (39 % de los libros de ciencia ficción) y “fantástico” con “romance” (35 % de los libros del género fantástico).

La Figura 11 muestra los diez términos más influyentes en el algoritmo. La longitud de las barras indica la reducción en el índice de Gini para el término respectivo. Los resultados obtenidos parecen ser bastante intuitivos: la palabra “detective”, por ejemplo, probablemente sea más frecuente en libros del género policial que en libros de otros géneros, por lo tanto es razonable que esté entre los términos de mayor importancia.

Figura 11: Términos más importantes para el modelo en el escenario E



4.4. *Boosting*

Se comenzó realizando el ajuste de los hiperparámetros para el escenario C. Dicho procedimiento requirió de una considerable cantidad de tiempo computacional debido a la gran cantidad de parámetros a ajustar y al hecho de que algunos de ellos interactúan entre sí. Los valores que produjeron un mejor resultado se presentan en la Tabla 13.

Se aplicó el algoritmo en el conjunto de prueba, obteniéndose los resultados mostrados en la Tabla 14.

Como se puede observar, la mejora en el desempeño no parece ser sustancial. Debido a la elevada carga de tiempo que conlleva el tuneo de los hiperparámetros para *boosting*, se decidió no seguir aplicando esta técnica para los siguientes escenarios. En su lugar, se buscó obtener mejores resultados recurriendo a los enfoques alternativos mencionados en la Sección 3.3.6.

Tabla 13: Selección de hiperparámetros para *boosting* con el escenario C

Hiperparámetro	Valor
nrounds	800
max_depth	200
eta	0.1
gamma	0
colsample_bytree	0.6
subsample	0.5
min_child_weight	0

Tabla 14: Comparación de la precisión y el coeficiente Kappa obtenidos al aplicar bosques aleatorios y *boosting* en el escenario C

Escenario	Bosques aleatorios		Boosting	
	Precisión	Kappa	Precisión	Kappa
C	59 %	0,51	61 %	0,53

4.5. Enfoques alternativos

Enfoque multi-etiqueta

Los resultados obtenidos bajo el enfoque multi-etiqueta se presentan en la Tabla 15. Debido a la forma en la que se calcula la proporción de observaciones clasificadas correctamente, en este caso no fue posible crear la matriz de confusión, empleándose como única medida de evaluación del desempeño a la precisión global. Se observa que la precisión promedio pasó de 54 % a 63 %. Los escenarios que presentaron un mayor porcentaje de mejora fueron el escenario F y el escenario D. Los que presentaron menor porcentaje de mejora fueron los escenarios H y C.

Tabla 15: Precisión obtenida con enfoque de una etiqueta y con enfoque multi-etiqueta por escenario

Escenario	Precisión con una etiqueta	Precisión multi-etiqueta	Porcentaje de mejora
A	64 %	74 %	15,6 %
B	51 %	58 %	13,7 %
C	59 %	66 %	11,8 %
D	50 %	61 %	22,0 %
E	64 %	74 %	15,6 %
F	48 %	59 %	22,9 %
G	52 %	60 %	15,3 %
H	47 %	52 %	10,6 %
Promedio	**54 %**	**63 %**	**15,9 %**

Múltiples predicciones

Los resultados obtenidos bajo este enfoque se presentan en la Tabla 16. Se observa que la precisión promedio pasó de 54 % a 63 %, similar a lo visto previamente en la Tabla 15. Los escenarios que presentaron un mayor porcentaje de mejora fueron el escenario G y el escenario H. Los que presentaron menor porcentaje de mejora fueron los escenarios D y E.

Tabla 16: Precisión obtenida por escenario tomando una única predicción y dos predicciones

Escenario	Precisión con una predicción	Precisión con dos predicciones	Porcentaje de mejora
A	64 %	74 %	15,6 %
B	51 %	56 %	9,8 %
C	59 %	67 %	13,5 %
D	50 %	55 %	10,0 %
E	64 %	71 %	10,9 %
F	48 %	57 %	18,7 %
G	52 %	67 %	28,8 %
H	47 %	58 %	23,4 %
Promedio	**54 %**	**63 %**	**16,1 %**

Como se puede apreciar en las Tablas 15 y 16, utilizar criterios más flexibles llevó a modelos con mejor desempeño. Todos los escenarios mostraron mejoras de al menos un 9 % respecto a la precisión

obtenida anteriormente, siendo ésta superior al 52 % en todos los casos. Esto podría indicar que recurrir a enfoques menos estrictos para la clasificación automática de textos de ficción puede producir resultados más satisfactorios, lo cual es consistente con recomendaciones realizadas por expertos.

5. Discusión y conclusiones

En este trabajo se propuso aplicar una técnica de clasificación automática con el objetivo de clasificar textos de ficción dentro de siete géneros predeterminados. Dicha técnica fue aplicada en ocho escenarios distintos, permitiendo contrastarlos y determinar qué metodología produce mejores resultados.

Se obtuvieron 746 documentos, correspondientes a obras de ficción en español o traducidas a dicho idioma, que fueron descargados de forma gratuita de diversos sitios web. Tras su limpieza, los textos fueron importados al programa R y se procedió a la construcción de atributos. Se calculó la frecuencia de términos y la estadística TF-IDF tanto para palabras individuales como para bigramas. Adicionalmente, para cada una de las cuatro combinaciones resultantes, se creó una versión filtrada con el propósito de eliminar los nombres propios presentes y disminuir el sobreajsute.

El algoritmo con el que se trabajó fue bosques aleatorios, el cual ha demostrado en estudios previos tener muy buen desempeño en la tarea de clasificar textos. Para obtener los mejores resultados posibles, los hiperparámetros del algoritmo fueron tuneados para cada escenario aplicando validación cruzada. Asimismo, se experimentó con distintos valores a la hora de elegir la cantidad de variables incluidas en el ajuste.

Los modelos entrenados fueron evaluados calculando la precisión global y la estadística Kappa. Los mejores resultados se obtuvieron al utilizar unigramas y frecuencia de términos con filtro, alcanzándose una precisión del 64 % y un valor Kappa de 0,58. Los escenarios con bigramas y con TF-IDF presentaron consistentemente un peor rendimiento que sus contrapartes con unigramas y frecuencia de términos. Si bien el mejor escenario fue uno en el que se aplicó el filtro, para todos los demás casos el uso de este mostró un desempeño inferior que el escenario análogo sin filtrar. Al evaluar el mejor escenario, se observó que la clase con mayor precisión fue “romance” (82 %), mientras que la menor precisión se observó para la clase “ciencia ficción” (26 %).

Si bien los resultados observados fueron aceptables, no fueron tan satisfactorios como los alcanzados en otros estudios. Una causa probable de estos resultados son las dificultades intrínsecas de trabajar con textos de ficción. En vista de ello, se decidió evaluar los modelos utilizando dos enfoques alternos. Primero se utilizó un enfoque multi-etiqueta, asignándole más de un género a cada texto, y en segundo lugar se consideraron las dos primeras clases predichas en lugar de solo la primera. En ambos casos se observó un aumento considerable en la precisión de cada escenario, siendo ésta superior al 52 % en todos los casos, por lo que se concluye que utilizar criterios más flexibles puede ser más recomendable en la clasificación automática de textos de ficción.

A pesar de que en este trabajo se compararon distintos escenarios de análisis, existe una enorme variedad de aspectos que podrían ser abordados y contrastados. A continuación, se describirán algunas

áreas que podrían ser investigadas en futuros estudios.

Además de frecuencia de términos y TF-IDF, existen otras estadísticas que se podrían calcular para cada escenario y contrastar sus resultados. Por ejemplo, se pueden llevar a cabo un análisis de sentimientos: cada palabra es asociada con una o más emociones y en función de ellas se evalúa qué sentimientos predominan en el texto (Sielge y Robinson, 2017). Además es posible crear nuevos conjuntos combinando escenarios, por ejemplo unigramas y bigramas (Pranckevičius y Marcinkevičius, 2017).

Otro tema que podría profundizarse es la lematización. En los cuatro escenarios con unigramas, se aplicó lematización mediante un diccionario de lemas, mientras que en los bigramas no se hizo. Se podrían contrastar los resultados obtenidos si no se hubiera realizado este proceso, o bien utilizar metodologías alternativas como lexematización.

En esta tesina no se aplica estandarización para los atributos calculados. Esto se decidió debido a que hay evidencia de que podría influir en un peor desempeño del modelo (Falk, 2019). Sin embargo, como se observó en el análisis descriptivo, las clases del conjunto de datos resultaron muy desbalanceadas, por lo que es posible que normalizar las variables derive en mejores resultados.

Finalmente, existen otros algoritmos basados en árboles que pueden emplearse para comparar sus resultados con los alcanzados con bosques aleatorios. Durante esta tesina ya se experimentó con la técnica de *boosting*, pero no se profundizó en ella debido a los altos tiempos de cómputo. Otra posibilidad es aplicar la metodología de bosques aleatorios condicionales, la cual es muy similar a bosques aleatorios, pero en lugar de utilizar árboles construidas con el algoritmo CART, los construye con el algoritmo CTREE (Jena y Dehuri, 2019).

6. Referencias bilbiográficas

Ávila Argüelles, R. (2008). *Clasificación bibliotecaria automática usando identificación simple de términos con métodos lógico-combinatorios a partir de información escasa*. [Tesis de maestría, Instituto Politécnico Nacional] http://tesis.ipn.mx/handle/123456789/3728

Baeza-Yates, R. y Ribeiro-Neto B. (2011). *Modern information retrieval: The concepts and technology behind search*. Addison-Wesley: Harlow.

Beltrán, C. (2011). Aplicación del análisis de regresión logística multinomial en la clasificación de textos académicos: Biometría, Filosofía y Lingüística informática. *INFOSUR, 5*. http://hdl.handle.net/2133/14278

Benoit, K., Muhr, D., y Watanabe, K. (2021). stopwords: Multilingual stopword lists (Versión 2.3) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=stopwords

Benoit, K., Watanabe, K., Wang, H., Nulty, P., Obeng, A., Müller, S., y Matsuo, A. (2021). quanteda: An R package for the quantitative analysis of textual data. *Journal of Open Source Software, 3(30), 774*. https://joss.theoj.org/papers/10.21105/joss.00774

Boehmke, B., y Greenwell, B. (2020). *Hands-on machine learning with R*. CRC Press.

Breiman, L. (2001). Random Forests. *Machine Learning, 45*(1), 5–32. https://doi.org/10.1023/a:1010933404324

Brigadoi, I (2021). *Genre classification using syntactic features*. [Tesis de maestría, Universidad de Upsala] http://uu.diva-portal.org/smash/get/diva2:1599825/FULLTEXT01.pdf

Calvo Torres, M. (2017). *Text Analytics para Procesado Semántico*. [Tesis de maestría, Universidad de Santiago de Compostela] http://eio.usc.es/pub/mte/descargas/ProyectosFinMaster/Proyecto_1475.pdf

Cárdenas, J., Olivares, G., y Alfaro, R. (2014). Clasificación automática de Textos usando Redes de Palabras. *Revista Signos, 47*(86), 346–364. https://doi.org/10.4067/s0718-09342014000300001

Chen, T., y Guestrin, C. (2016). XGBoost. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. https://doi.org/10.1145/2939672.2939785

Clark, M. (2018) *An Introduction to Text Processing and Analysis with R*. Universidad de Michigan. https://m-clark.github.io/text-analysis-with-R/

Csardi G. y Nepusz T. (2006). The igraph software package for complex network research, *InterJournal, Complex Systems, 1695*. https://igraph.org

Das, M., Kamalanathan, S., y Alphonse, P. (2021). *A Comparative Study on TF-IDF Feature Weighting Method and Its Analysis Using Unstructured Dataset*. COLINS.

Falk, O. (2019). *Automated fiction classification - an explorative study of fiction classification using machine-learning techniques*. [Tesis de maestría, Universidad de Borås] http://www.diva-portal.org/smash/get/diva2:1395257/FULLTEXT01.pdf

Feinerer, I., y Hornik, K. (2020). tm: Text mining package (Versión 0.7-8) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=tm

Feldman, R., y Sanger, J. (2013). *The text Mining Handbook: Advanced Approaches in analyzing unstructured data*. Cambridge University Press.

Fernández-Delgado, M., Cernadas, E., Barro, S., y Amorim, D.G. (2014). Do we need hundreds of classifiers to solve real world classification problems. *Journal of Machine Learning Research, 15*, 3133-3181.

Fradejas Rueda, J. M. (2020). *Cuentapalabras Estilometría y análisis de texto con R para filólogos*. Laboratorio de Innovación en Humanidades Digitales. http://www.aic.uva.es/cuentapalabras/

Gagolewski M. (2022). stringi: Fast and portable character string processing in R. *Journal of Statistical Software, 103(2)*, 1-59. https://doi.org/10.18637/jss.v103.i02

Hastie, T., Friedman, J., y Tisbshirani, R. (2017). *The elements of Statistical Learning: Data Mining, Inference, and prediction*. Springer.

Hvitfeldt, E., y Silge, J. (2022). *Supervised machine learning for text analysis in R*. CRC Press, Taylor & Francis Group.

Iannone, R. (2022). DiagrammeR: Graph/network visualization (Versión 1.0.9) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=DiagrammeR

James, G., Witten, D., Hastie, T., y Tibshirani, R. (2021). *An introduction to statistical learning: With applications in R*. Springer.

Jena, M., y Dehuri, S. (2020). Decision tree for classification and regression: A state-of-the art review. *Informatica, 44*(4). https://doi.org/10.31449/inf.v44i4.3023

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. *Machine Learning: ECML-98*, 137–142. https://doi.org/10.1007/bfb0026683

Jordan, E. (2014). *Automated Genre Classification In Literature*. [Tesis de maestría, Universidad Estatal de Kansas] https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.864.9862&rep=rep1&type=pdf

Kou, G., Yang, P., Peng, Y., Xiao, F., Chen, Y., y Alsaadi, F. E. (2020). Evaluation of feature selection methods for text classification with small datasets using multiple criteria decision-making methods. *Applied Soft Computing, 86*, 105836. https://doi.org/10.1016/j.asoc.2019.105836

Kuhn, M. (2022). caret: Classification and regression training (Versión 6.0-93) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=caret

Kuhn, M., y Johnson, K. (2016). *Applied predictive modeling*. Springer.

Kusumaningrum, R., Indihatmoko, T. A., Juwita, S. R., Hanifah, A. F., Khadijah, K., y Surarso, B. (2020). Benchmarking of multi-class algorithms for classifying documents related to stunting. *Applied Sciences, 10(23)*, 8621. https://doi.org/10.3390/app10238621

Liaw, A., y Wiener, M. (2002). Classification and Regression by randomForest. *R News 2(3)*, 18-22.

Manning, C. D., Raghavan, P., y Schutze H. (2019) *Introduction to information retrieval*. Cambridge University Press.

Maron, M. E. (1961). Automatic indexing: An experimental inquiry. *Journal of the ACM, 8*(3), 404–417. https://doi.org/10.1145/321075.321084

Microsoft Corporation y Weston, S. (2022). doParallel: Foreach parallel adaptor for the 'parallel' package (Versión 1.0.17) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=doParallel

Milborrow, S. (2022). rpart.plot: Plot 'rpart' models: an enhanced version of 'plot.rpart' (Versión 3.1.1) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=rpart.plot

Mirończuk, M. M., y Protasiewicz, J. (2018). A recent overview of the state-of-the-art elements of text classification. *Expert Systems with Applications, 106*, 36–54. https://doi.org/10.1016/j.eswa.2018.03.058

Neuwirth, E. (2022). RColorBrewer: Colorbrewer palettes (Versión 1.1-3) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=RColorBrewer

Nigam, K., Lafferty, J. y McCallum, A. (1999) Using Maximum Entropy for Text Classification. *IJCAI-99 Workshop on Machine Learning for Information Filtering*, 61-67.

Ooms, J. (2022). pdftools: Text extraction, rendering and converting of pdf documents (Versión 3.3.0) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=pdftools

Pedersen, T. L. (2022). ggraph: An implementation of grammar of graphics for graphs and networks (Versión 2.0.6) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=ggraph

Pennec, E. L., y Slowikowski, K. (2019). ggwordcloud: A word cloud geom for 'ggplot2' (Versión 0.5.0) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=ggwordcloud

Perry, P. O. (2021). corpus: Text corpus analysis (Versión 0.10.2) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=corpus

Pranckevičius, T., y Marcinkevičius, V. (2017). Comparison of naive bayes, Random Forest, decision tree, support vector machines, and logistic regression classifiers for text reviews classification. *Baltic Journal of Modern Computing, 5(2)*. https://doi.org/10.22364/bjmc.2017.5.2.05

Probst, P., Wright, M. N., y Boulesteix, A. L. (2019). Hyperparameters and tuning strategies for Random Forest. *WIREs Data Mining and Knowledge Discovery, 9(3)*. https://doi.org/10.1002/widm.1301

Porras, H. (2020). *Análisis de comportamiento en redes sociales usando Procesamiento del Lenguaje Natural*. Rpubs. https://rpubs.com/hugoporras/nlp_capitulo1

R Core Team (2022). *R: A language and environment for statistical computing*. The Comprehensive R Archive Network. https://www.R-project.org/

Real Academia Española. (2014). *Diccionario de la lengua española* (23ª ed.).

Robinson, D. (2021). gutenbergr: Download and process public domain works from project gutenberg (Versión 0.2.1) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=gutenbergr

Salinas, J., y Izetta, J. (2016, octubre). *Clasificación automática de textos periodísticos usando Random Forest* [Conferencia]. XXII Congreso Argentino de Ciencias de la Computación, San Luis, Argentina. http://sedici.unlp.edu.ar/handle/10915/55733

Sebastiani, F. (2002). *Machine learning in automated text categorization. ACM Computing Surveys, 34*(1), 1–47. https://doi.org/10.1145/505282.505283

Shah, K., Patel, H., Sanghvi, D., y Shah, M. (2020). A comparative analysis of logistic regression, Random Forest and KNN models for the text classification. *Augmented Human Research, 5*(1). https://doi.org/10.1007/s41133-020-00032-0

Shiroya, P. (2021). Book genre categorization using machine learning algorithms (K-nearest neighbor, support Vector Machine and logistic regression) using customized dataset. *International Journal of Computer Science and Mobile Computing, 10(3)*, 14–25. https://doi.org/10.47760/ijcsmc.2021.v10i03.002

Sievert, C. (2020). *Interactive Web-Based Data Visualization with R, plotly, and shiny*. Chapman and Hall/CRC Florida.

Silge, J., y Robinson, D. (2017). *Text mining with R: A tidy approach*. O'Reilly.

Silge, J., y Robinson, D. (2016). tidytext: Text Mining and Analysis Using Tidy Data Principles in R. *Journal of Open Source Software, 1(3)*, 37. http://dx.doi.org/10.21105/joss.00037

Therneau, T., y Atkinson, B. (2022). rpart: Recursive partitioning and regression trees (Versión 4.1.16) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=rpart

Venegas, R. (2007). Clasificación de Textos Académicos en función de su contenido léxico-semántico. *Revista Signos, 40*(63). https://doi.org/10.4067/s0718-09342007000100012

Ward, M., y Saarti, J. (2018). Reviewing, rebutting, and reimagining fiction classification. *Cataloging & Classification Quarterly, 56*(4), 317–329. https://doi.org/10.1080/01639374.2017.1411414

Wang, T.-Y., y Chiang, H.-M. (2011). Solving multi-label text categorization problem using support vector machine approach with membership function. *Neurocomputing, 74*(17), 3682–3689. https://doi.org/10.1016/j.neucom.2011.07.001

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J.,

Robinson, D., Seidel, D. P., Spinu, V., Takahashi, K., Vaughan, D., Wilke, C., Woo, K., y Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software, 4(43), 1686*. https://joss.theoj.org/papers/10.21105/joss.01686.

Wickham, H. (2022). rvest: Easily harvest (scrape) web pages (Versión 1.0.3) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=rvest

Wilke, C. O. (2020). cowplot: Streamlined plot theme and plot annotations for 'ggplot2' (Versión 1.1.1) [Software de computador]. The Comprehensive R Archive Network. https://CRAN.R-project.org/package=cowplot

Yadla, H. K., y Rao, P. (2020). Machine Learning Based Text Classifier Centered On TF-IDF Vectoriser. *International Journal of Scientific & Technology Research, 9(3)*, 583-586.

Yako, M. (2021). *Emotional Content in Novels for Literary Genre Prediction And Impact of Feature Selection on Text Classification Models*. [Tesis de maestría, Universidad de Upsala] http://uu.diva-portal.org/smash/get/diva2:1572545/FULLTEXT01.pdf

Zanasi, A. (2007). *Text mining and its applications to intelligence, Crm and Knowledge Management*. WIT Press.

Anexo 1: Bosques aleatorios en la bilbiografía

Tabla 17: Resumen de los resultados obtenidos con bosques aleatorios en la bibliografía revisada

Referencia	**Idioma/s**	**Corpus**	**Nº de categorías**	**Mejor variable**	**Mayor precisión**
Salinas e Izeta (2016)	Español	Noticias (dos conjuntos)	2 y 3	TF-IDF	99 % y 93 %
Pranckevičius y Marcinkevičius (2017)	Inglés	Reseñas	5	Combinación de uni/bi/tri-gramas	43,93 %
Kusumaningrum et al. (2020)	Inglés / Indonesio	Tuits / Documentos	2	Frecuencia de términos	76 %
Shah et al. (2020)	Inglés	Noticias	5	TF-IDF	93 % (promedio)
Yadla y Rao (2020)	Inglés	Noticias	5	TF-IDF	95,53 %
Brigadoi (2021)	Inglés	Libros de ficción	8	Frecuencia	72 %
Das et al. (2021)	Inglés	Reseñas	2	TF-IDF	93,80 %
Yako (2021)	Inglés	Libros de ficción	8	Análisis de sentimientos	No se reporta

Anexo 2: Nubes de palabras

Figura 12: Nubes de palabras correspondientes a escenarios con unigramas para documentos pertenecientes al género aventura

Figura 13: Nubes de palabras correspondientes a escenarios con unigramas para documentos pertenecientes al género fantasía

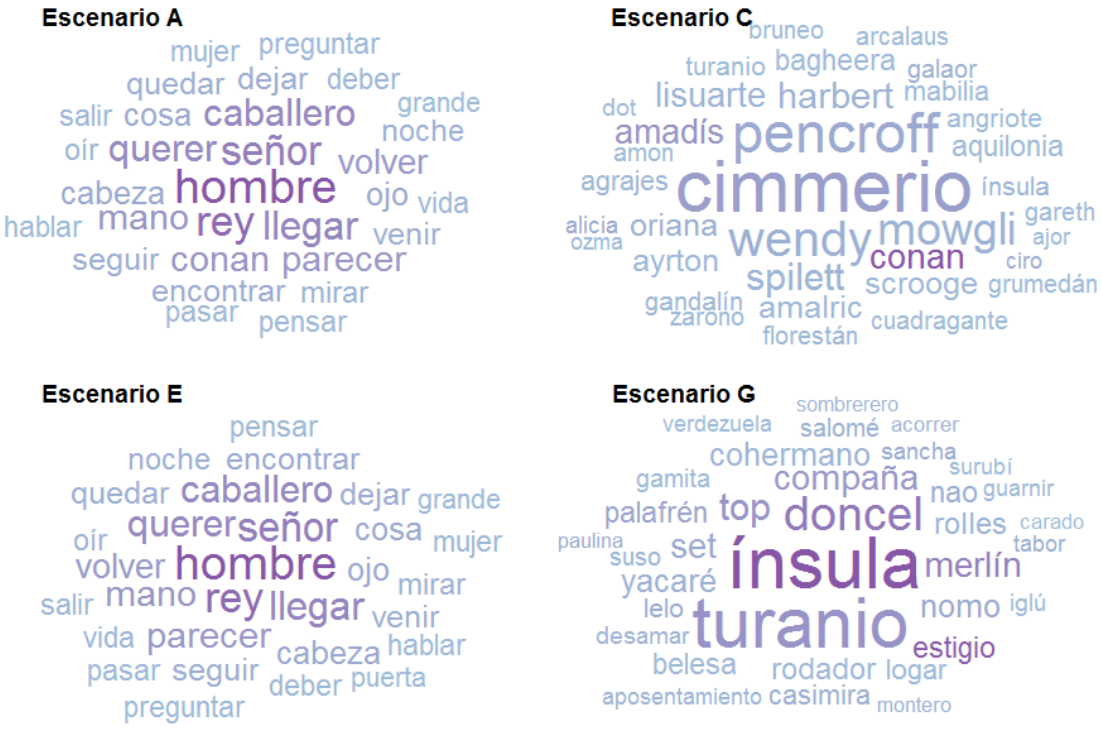


Figura 14: Nubes de palabras correspondientes a escenarios con unigramas para documentos pertenecientes al género ficción histórica

Figura 15: Nubes de palabras correspondientes a escenarios con unigramas para documentos pertenecientes al género policial o misterio

Figura 16: Nubes de palabras correspondientes a escenarios con unigramas para documentos pertenecientes al género romance

Figura 17: Nubes de palabras correspondientes a escenarios con unigramas para documentos pertenecientes al género terror

Anexo 3: Diagramas de red

Figura 18: Diagramas de red correspondientes a escenarios con bigramas para documentos pertenecientes al género aventura

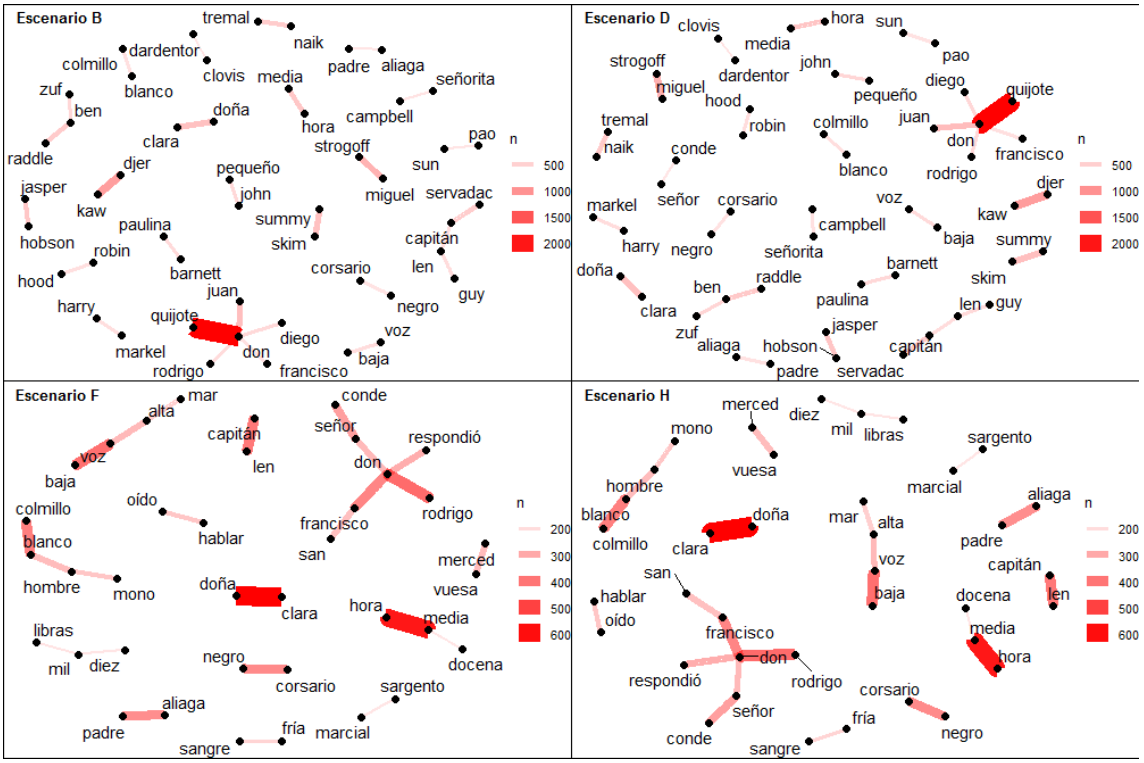


Figura 19: Diagramas de red correspondientes a escenarios con bigramas para documentos pertenecientes al género fantasía

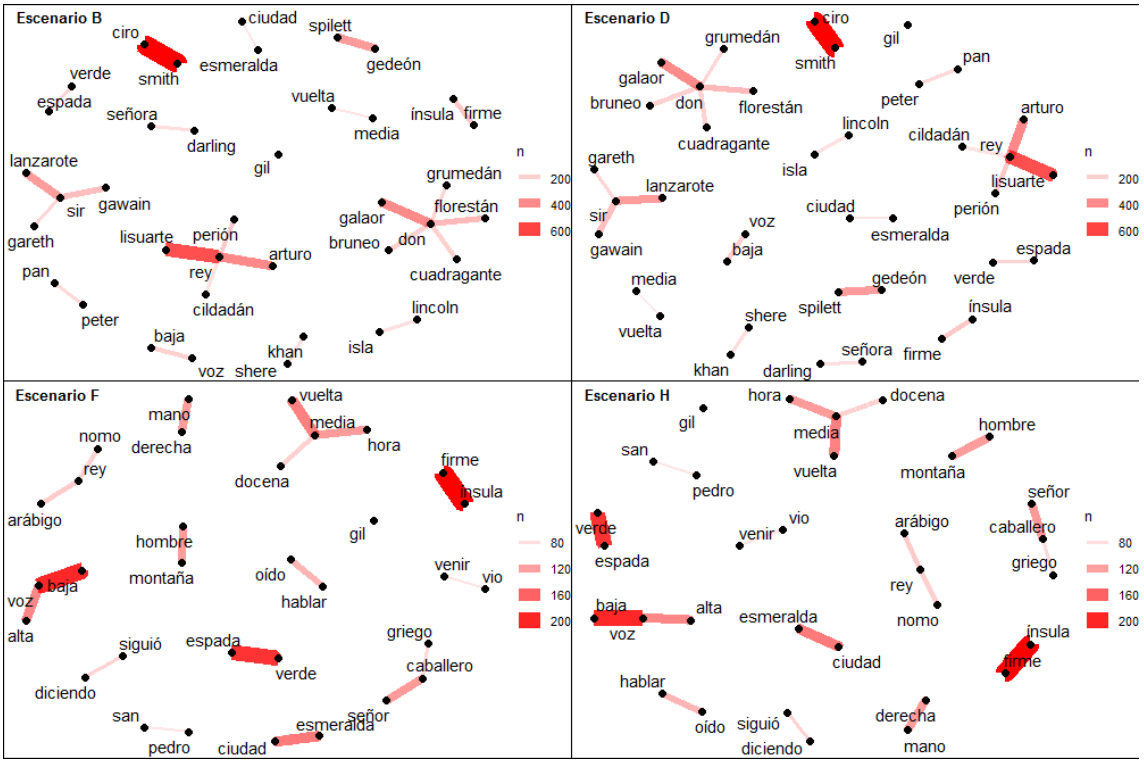


Figura 20: Diagramas de red correspondientes a escenarios con bigramas para documentos pertenecientes al género ficción histórica

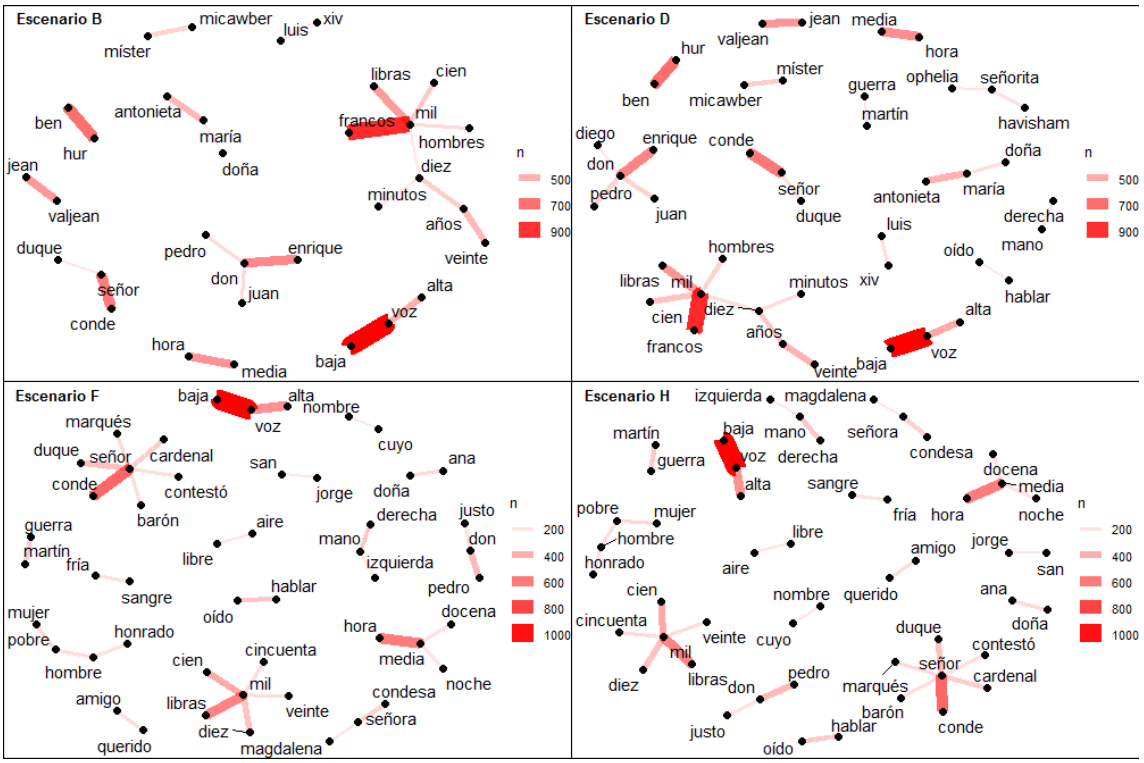


Figura 21: Diagramas de red correspondientes a escenarios con bigramas para documentos pertenecientes al género policial o misterio

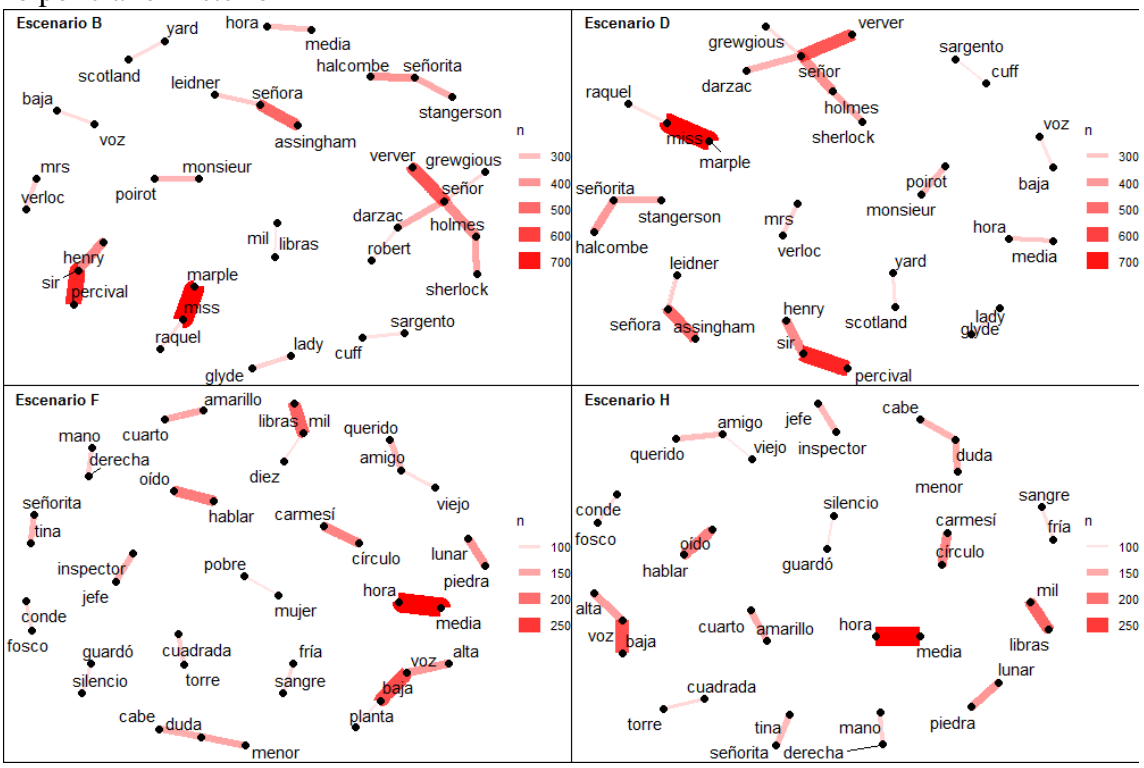


Figura 22: Diagramas de red correspondientes a escenarios con bigramas para documentos pertenecientes al género romance

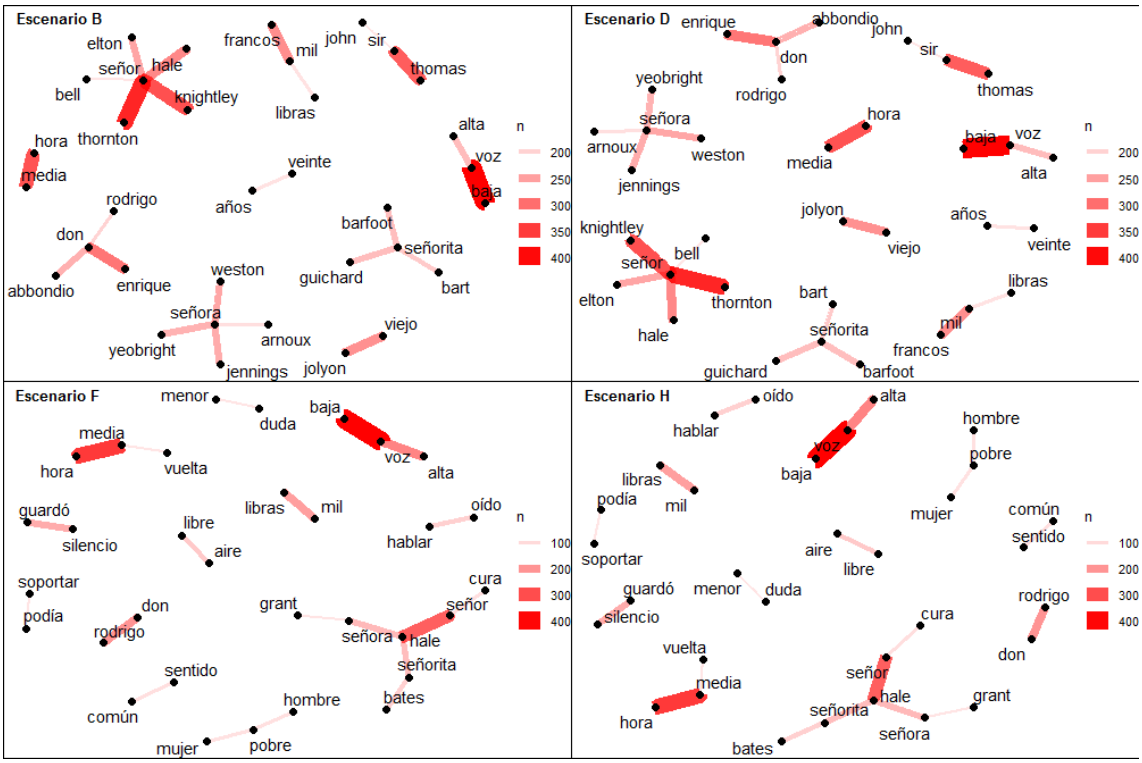
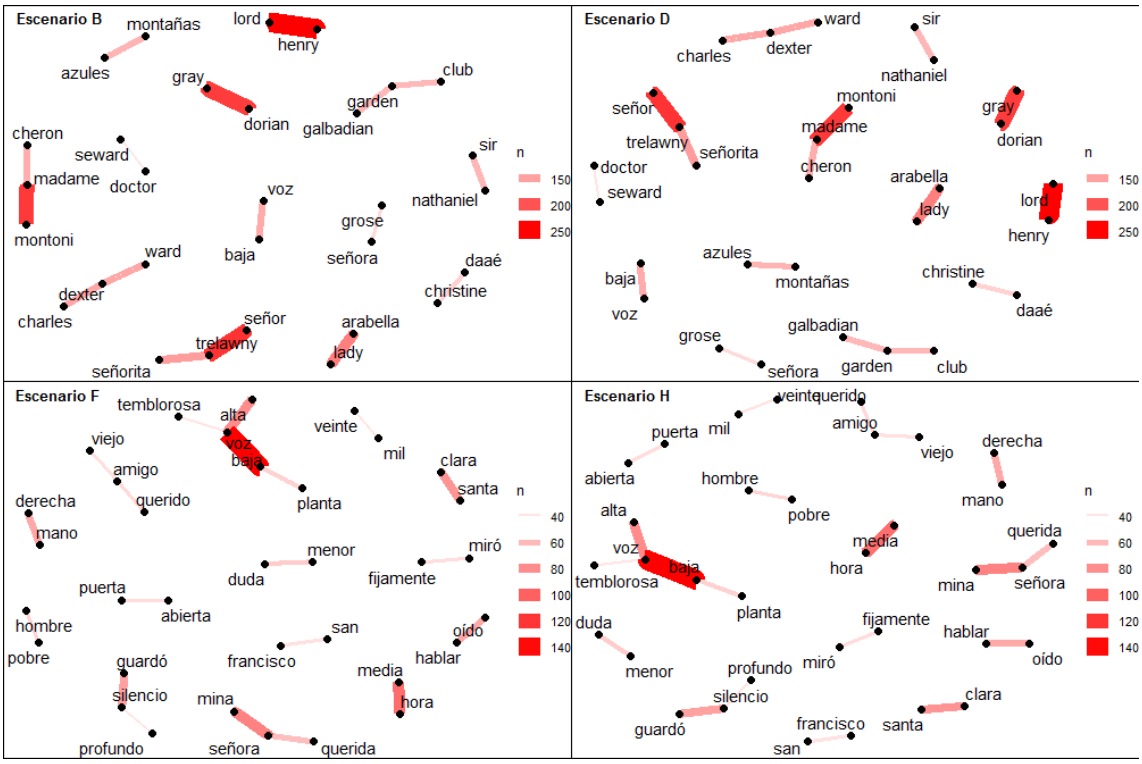


Figura 23: Diagramas de red correspondientes a escenarios con bigramas para documentos pertenecientes al género terror



Anexo 4: Matrices de confusión

Figura 24: Matriz de confusión para el escenario A

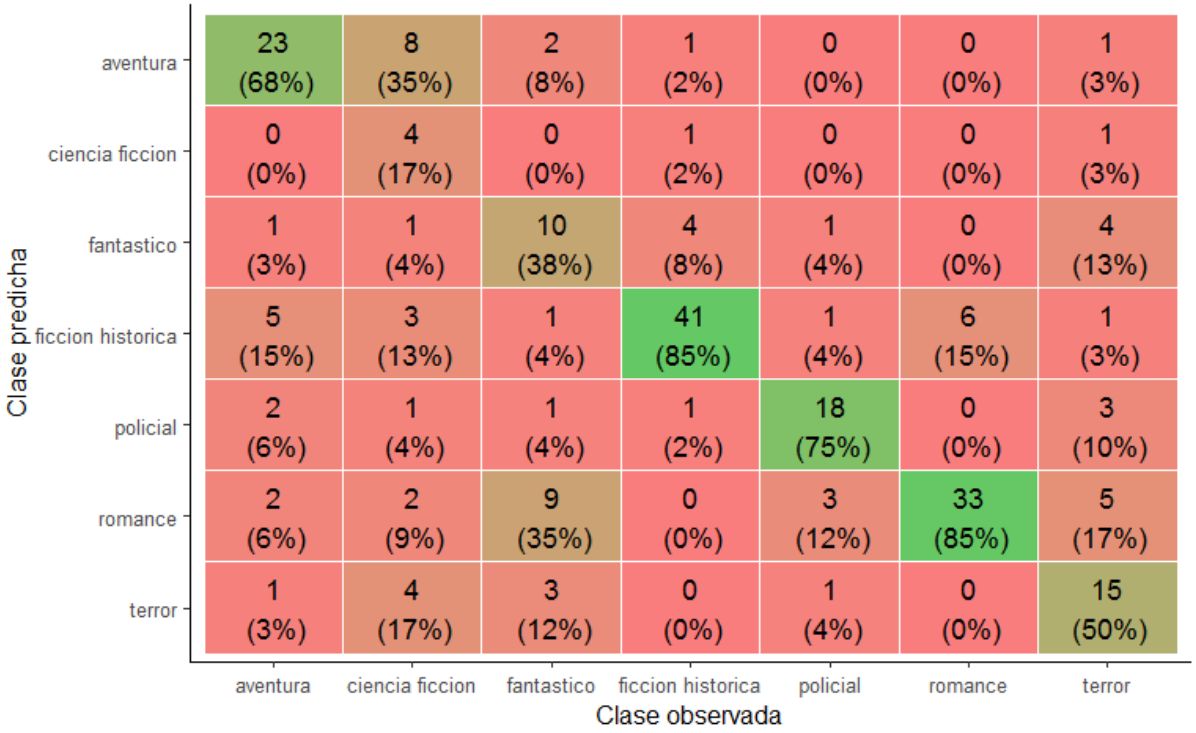


Figura 25: Matriz de confusión para el escenario B

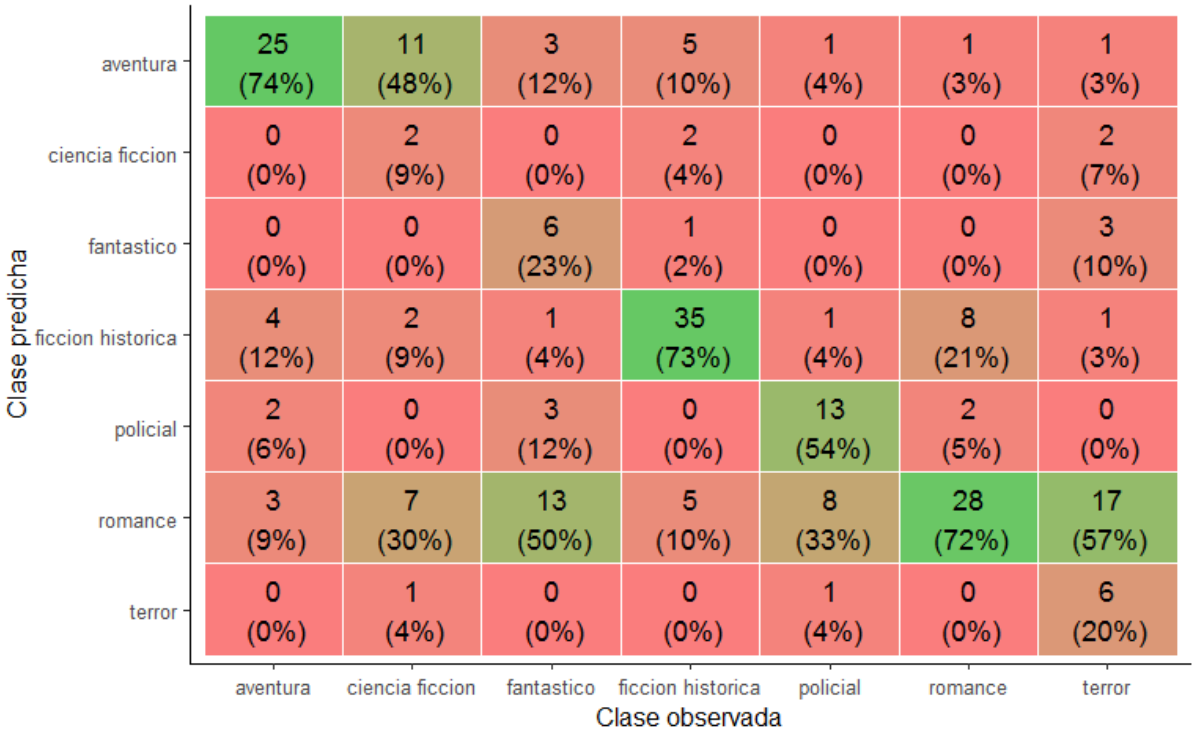


Figura 26: Matriz de confusión para el escenario C

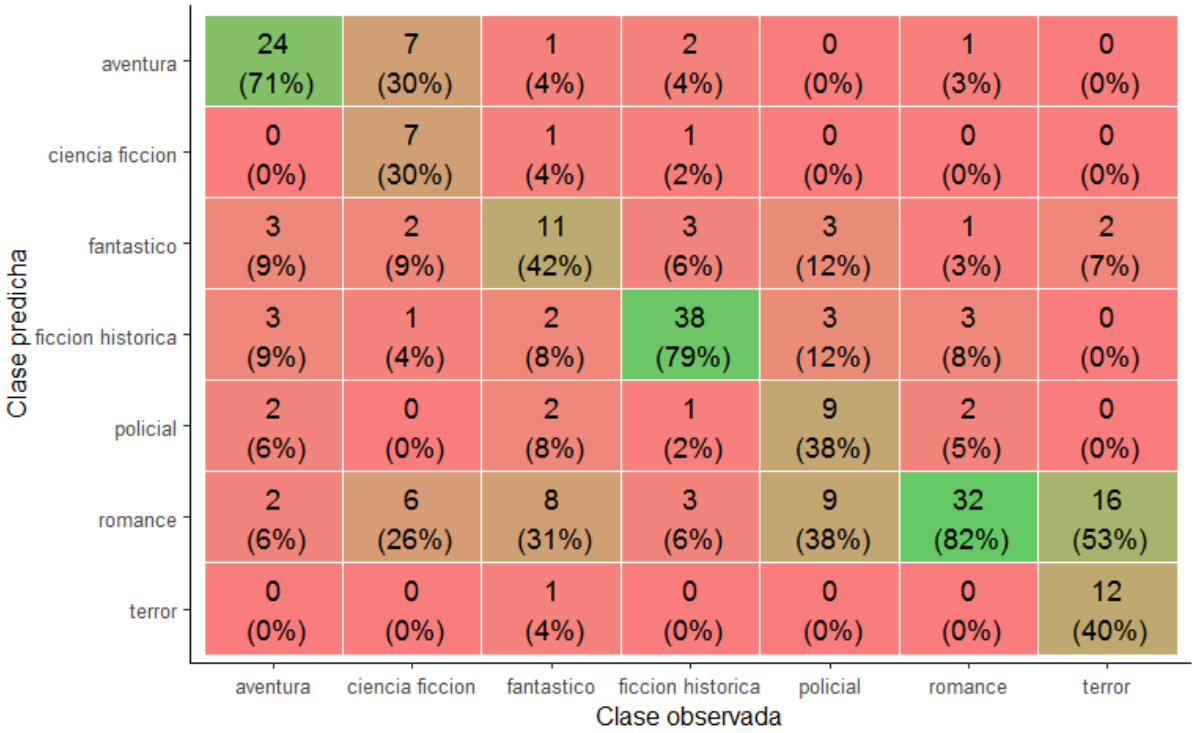


Figura 27: Matriz de confusión para el escenario D

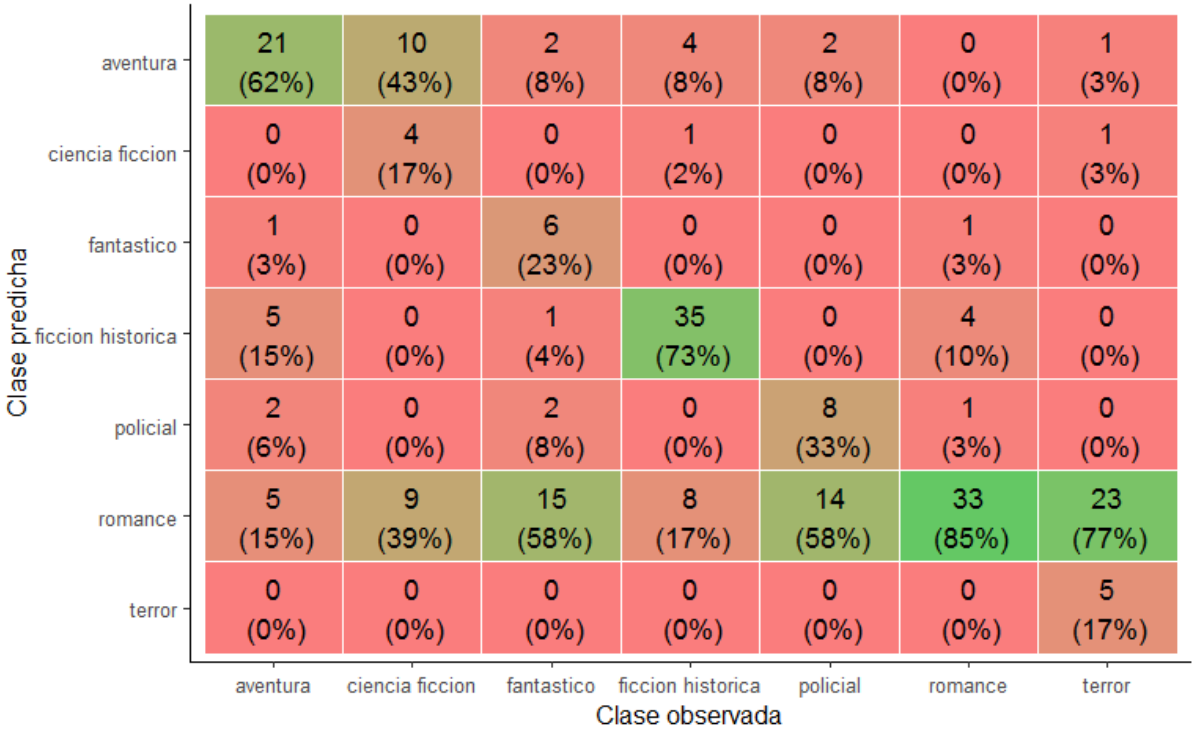


Figura 28: Matriz de confusión para el escenario F

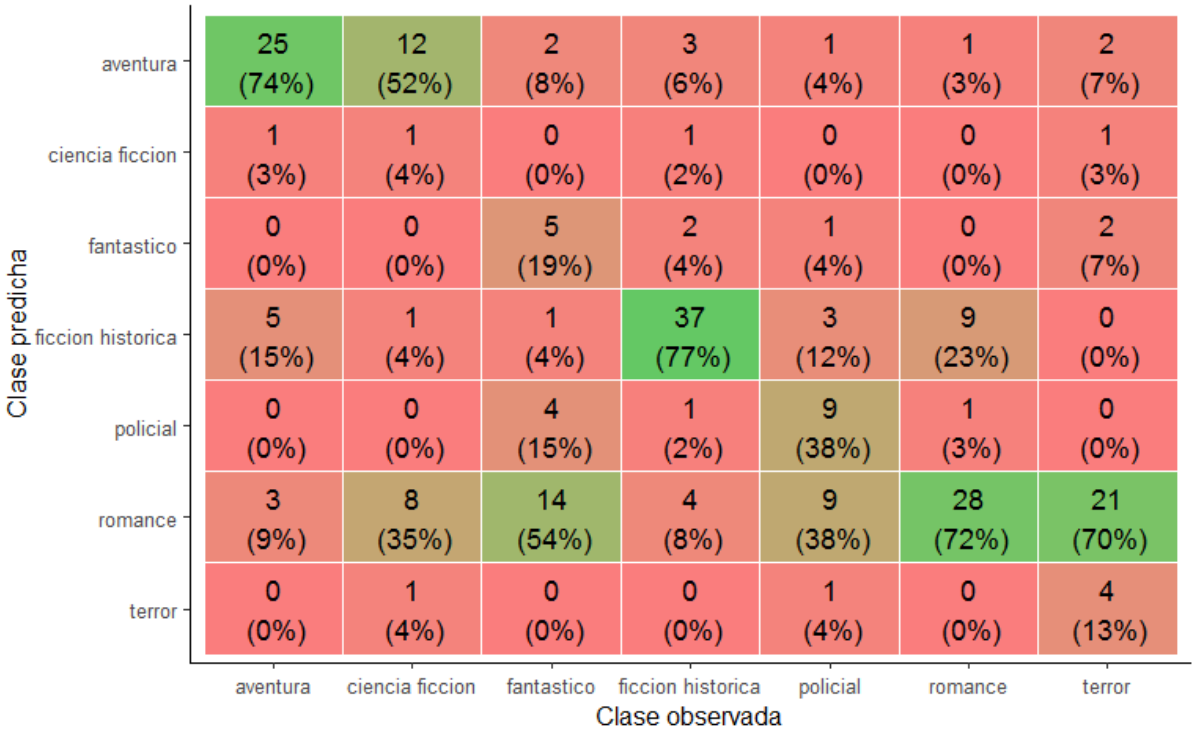


Figura 29: Matriz de confusión para el escenario G

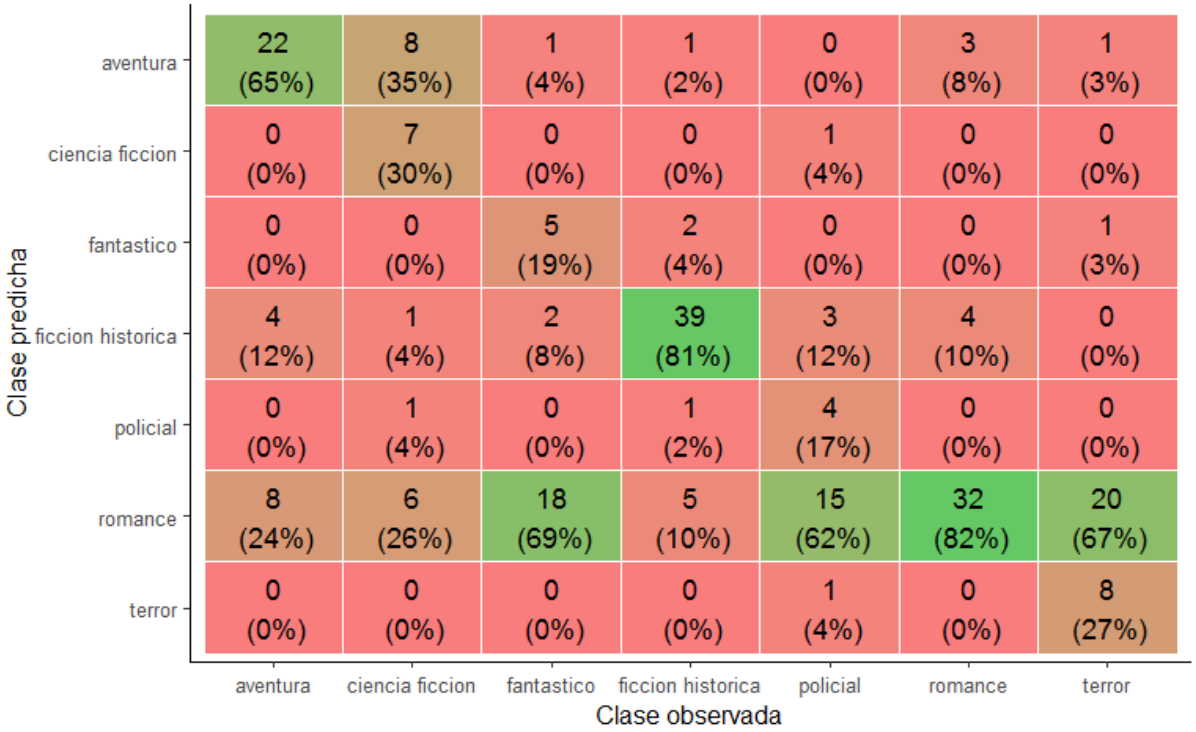


Figura 30: Matriz de confusión para el escenario H