



Universidad Politécnica
de Madrid

**Escuela Técnica Superior de
Ingenieros Informáticos**



Grado en Matemáticas e Informática

Trabajo Fin de Grado

**Clasificación inferida mediante datos:
Titulación de Grado del autor del TFG**

Autor: Guillermo Fernández Sánchez

Tutor: Juan Antonio Fernández del Pozo de Salamanca

Madrid, Enero 2025

Este Trabajo Fin de Grado se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Trabajo Fin de Grado
Grado en Matemáticas e Informática

Título: Clasificación inferida mediante datos: Titulación de Grado del
autor del TFG

Enero 2025

Autor: Guillermo Fernández Sánchez

Tutor: Juan Antonio Fernández del Pozo de Salamanca
Inteligencia Artificial
Escuela Técnica Superior de Ingenieros Informáticos
Universidad Politécnica de Madrid

Resumen

Este Trabajo de Fin de Grado consiste en realizar un análisis estadístico sobre una muestra de datos obtenidos del Archivo Digital de la Universidad Politécnica de Madrid (AD-UPM) para posteriormente, como objetivo principal del TFG, desarrollar un modelo de clasificación centrado en la variable Titulación de Grado. Para obtener los datos sobre los que se va a trabajar, previamente se realizará un estudio de la fuente de información (AD-UPM), una definición de la muestra y una selección del método de muestreo, para a continuación extraer los datos y prepararlos. Dicha muestra se obtendrá de los Grados que tengan mayor número de TFG'S subidos al Archivo Digital para tener un mayor conjunto de datos disponibles, minimizando así el error y obteniendo un modelo más robusto.

Se creará una base de datos donde se guardará la Titulación del Grado, el título y el resumen de cada TFG, siendo las dos últimas las variables a modelizar, a opción de añadir más en función del rendimiento del clasificador. Además, se llevará acabo un análisis de los principales métodos de clasificación, buscando así el que mejor se ajuste con los datos disponibles y comparando los distintos resultados que se vayan obteniendo.

Abstract

This Final Degree Project consists of carrying out a statistical analysis on a sample of data obtained from Archivo Digital de la Universidad Politécnica de Madrid (AD-UPM) to subsequently, as the main objective of the TFG, develop a classification model focused on the variable Degree Qualification. To obtain the data on which to work, a study of the information source (AD-UPM), a definition of the sample and a selection of the sampling method will be previously carried out, in order to then extract the data and prepare it. This sample will be obtained from the Degrees that have the largest number of TFG's uploaded to the Digital Archive in order to have a larger set of available data, thus minimizing the error and obtaining a more robust model.

A database will be created where the Degree Title, the title and the summary of each TFG will be stored, the last two being the variables to be modelled, with the option of adding more depending on the performance of the classifier. In addition, an analysis of the main classification methods will be carried out, thus looking for the one that best fits the available data and comparing the different results that are obtained.

Tabla de contenidos

1. Introducción	1
1.1. Motivación del trabajo	1
1.2. Objetivos	2
1.3. Contenido de la memoria	2
2. Estado del arte	5
2.1. <i>Text Mining</i>	5
2.1.1. Principales pasos en el <i>text mining</i>	5
2.2. Aprendizaje automático	6
2.2.1. Aprendizaje supervisado	6
2.2.2. Aprendizaje no supervisado	6
2.3. Clasificación de textos	7
2.3.1. Selección de características	8
2.4. Modelos de clasificación	9
2.4.1. Naive Bayes	10
2.4.2. K-Nearest Neighbors (KNN)	11
2.4.3. Regresión Logística	11
2.5. Herramientas de análisis	13
2.5.1. R	13
2.5.2. RStudio	13
2.5.3. Librerías de R	13
3. Base de datos	15
3.1. Selección de la muestra	15
3.2. Diseño de la Base de Datos	17
4. Análisis y modelización de los datos	23
4.1. Preprocesamiento	23
4.2. Frecuencia de palabras y TF-IDF	24
4.3. Tabla documento-término	27
5. Construcción del clasificador y análisis de rendimiento	29
5.1. K-Medoids	29
5.2. Modelos de clasificación	31
5.2.1. KNN	32
5.2.2. Naive Bayes	33

TABLA DE CONTENIDOS

5.2.3. Regresión Logística	35
5.3. Clustering sobre documentos	37
5.3.1. Clustering jerárquico	37
5.3.2. Topic Modeling	37
5.4. Modelos de clasificación en cada clase	39
6. Conclusiones y líneas futuras	47
6.1. Evaluación de los objetivos	47
6.2. Líneas futuras	48
6.3. Evaluación personal del proceso de realización del TFG	48
6.4. Análisis de impacto	49
6.4.1. Alineamiento con las ODS	49
Bibliografía	51
Anexos	55
A. Anexo	55
A.1. Anexo de script en R	55

Capítulo 1

Introducción

1.1. Motivación del trabajo

En mundo con cada vez más información y en su gran mayoría digitalizada, se ha vuelto una prioridad el poder organizar y acceder a dicha información de la manera más eficiente posible, para así conseguir analizar y extraer el conocimiento necesario para poder tomar las decisiones adecuadas.

Este proceso de extracción de información de los textos se conoce como *text mining*, una técnica que se utiliza para extraer información útil y significativa a partir de grandes volúmenes de datos textuales [1]. Los primeros pasos dentro de este proceso consisten en recopilar los datos que se van a analizar y llevar a cabo un preprocesamiento de estos datos, en el que el texto se limpia y se transforma en un formato adecuado para el análisis.

Una vez explorado los datos y encontrado los distintos patrones dentro del texto, entra en juego el campo del aprendizaje automático, o más conocido como machine learning. Este subárea de la inteligencia artificial se enfoca en desarrollar algoritmos que permiten a las máquinas aprender de los datos y hacer predicciones o tomar decisiones sin estar programadas explícitamente. Dentro de este campo, una de las tareas más comunes es la clasificación de textos [2], donde el objetivo es asignar una categoría o clase a un texto en particular para explorar y analizar grandes volúmenes de datos. En este contexto, el Archivo Digital de la Universidad Politécnica de Madrid (AD-UPM) ofrece un acceso amplio y diversificado a los documentos académicos y científicos generados por la Universidad, siendo toda esta información de datos no estructurados. Por ello, este repositorio no solo es una fuente valiosa de conocimiento en diversas áreas del saber, sino que además es perfecto para aplicar tanto técnicas de data mining que permitan analizar patrones y tendencias en la producción académica, como los algoritmos de machine learning para poder clasificar los distintos trabajos.

Este estudio, además de aportar una visión general sobre la producción académica en la UPM, ofrece una metodología que podría aplicarse en el futuro para organizar y clasificar automáticamente los TFGs según sus características principales, facilitando así el acceso a la información y su consulta por parte de los

usuarios de este archivo digital.

1.2. Objetivos

El presente Trabajo de Fin de Grado se centra en la creación de un clasificador de los TFG almacenados en el AD-UPM, para lo que se realizará un análisis de sus características y tendencias en las distintas Escuelas y Departamentos a lo largo de los últimos años. Este estudio busca la clasificar los trabajos en función de la variable “Titulación del Grado”, lo que nos permitirá entender mejor como se distribuyen los temas de los TFGs en las distintas titulaciones y tener así una visión más estructurada y generalizada del contenido que nos ofrece el repositorio. Esto implica un proceso que va desde la recopilación y estructuración de datos, la creación de una base de datos, hasta el desarrollo y evaluación de modelos de clasificación de textos.

Como objetivos más específicos dentro de este proyecto, se encuentran:

- Realizar un análisis exploratorio de los datos disponibles en el Archivo Digital de la UPM, identificando las variables relevantes y los patrones textuales específicos de cada titulación.
- Diseñar y construir una base de datos que permita almacenar y organizar de manera estructurada la información recopilada de los TFGs, asegurando su accesibilidad para el análisis y modelización posterior.
- Desarrollar y evaluar distintos modelos de clasificación de textos, como Naive Bayes o Regresión Logística, determinando cuál de ellos ofrece el mejor rendimiento en la clasificación de las titulaciones.
- Implementar un proceso de preprocesamiento de los textos, aplicando técnicas de minería de texto y procesamiento de lenguaje natural (NLP) para transformar el texto en datos numéricos que puedan ser utilizados por los modelos de clasificación.
- Proponer posibles mejoras y líneas futuras para optimizar el modelo o ampliar su alcance, explorando opciones como el uso de datos adicionales o técnicas más avanzadas de minería de texto.

1.3. Contenido de la memoria

A continuación, se resume brevemente los contenidos de los siguientes capítulos que conforman este trabajo:

- *Estado del arte*: En este capítulo se da contexto a la clasificación de texto, se revisan investigaciones y estudios previos relacionados con la clasificación de textos y la minería de datos, además de las distintas técnicas para extraer información de los textos. También se explicarán las herramientas utilizadas para realizar el proyecto.

- *Base de Datos:* En esta sección se describe el diseño y construcción de la base de datos utilizada para almacenar y organizar la información extraída del Archivo Digital de la UPM. Se detallan las especificaciones de la base de datos, su estructura (tablas y relaciones), los procesos de extracción y limpieza de los datos, así como los aspectos de seguridad y consistencia.
- *Análisis y Modelización de los Datos:* Aquí se detalla el análisis exploratorio de los datos para comprender mejor sus características y patrones. Incluye la preparación y transformación de datos textuales utilizando las técnicas explicadas en capítulos anteriores para su posterior modelización.
- *Modelos de clasificación:* En este capítulo se explican los modelos de clasificación de textos seleccionados para resolver el problema. Se describen las características, ventajas y desventajas de los modelos candidatos, como Naive Bayes, SVM, o redes neuronales, y se justifica la elección de los modelos utilizados en el estudio.
- *Construcción del Clasificador y Análisis del Rendimiento:* Este capítulo cubre el proceso de entrenamiento y optimización del clasificador final, así como la evaluación de su rendimiento utilizando métricas. Además, se incluyen los resultados obtenidos, análisis de errores y comparaciones de desempeño entre distintos modelos.
- *Conclusiones y Líneas Futuras:* Aquí se presentan las conclusiones del trabajo, resaltando los logros y limitaciones del clasificador desarrollado. También se proponen posibles mejoras y líneas de investigación futura que podrían optimizar o ampliar los resultados del estudio, como el uso de datos adicionales o técnicas avanzadas de procesamiento de texto.
- *Anexo:* En este capítulo final se muestra el código más importante utilizado en el proceso de extracción, limpieza, análisis y modelización de los datos, junto a comentarios explicativos para facilitar la comprensión de cada sección del código.

Capítulo 2

Estado del arte

2.1. *Text Mining*

El *data mining*, o minería de datos, [1] es el proceso de analizar grandes cantidades de datos para identificar patrones, relaciones y tendencias útiles. Su objetivo consiste en extraer información de un conjunto de datos y transformarla en una estructura comprensible para su uso posterior. El *text mining*, o minería de texto, es básicamente un subcampo de la minería de datos que se enfoca en estructurar datos no estructurados y analizarlos para generar nueva información de valor. Este tipo de datos no tienen un formato o esquema predefinido, por lo que son más difíciles de organizar, procesar y analizar directamente. Dado que aproximadamente el 80% de los datos del mundo pertenece a este tipo [3], la minería de texto se ha vuelto una herramienta de gran valor.

Dentro del contexto del *text mining*, surgen varios problemas, entre los que se encuentran el *clustering*, la clasificación de textos, realizar resumen de textos, reducción de la dimensionalidad o la extracción de información [1].

2.1.1. Principales pasos en el *text mining*

Recopilación de los datos

El primero de ellos sería recopilar los datos de un conjunto de documentos de texto, con sus correspondientes categorías, sobre los cuales se van a aplicar los modelos y almacenarlos, en este caso, en la base de datos creada. Este paso se realizará haciendo Web Scraping, que es el proceso de recolectar datos desde la web, ya sea de forma manual o automática.

Preprocesamiento

Lo siguiente que se debe hacer es el preprocesamiento de los datos, una fase fundamental que influirá de manera notoria en la precisión y el rendimiento que se obtenga tanto en el supervisado como en el no supervisado [4]. Esta fase consiste en limpiar y preparar los datos del texto eliminando símbolos innecesarios,

convirtiéndolos a minúsculas y manejando caracteres especiales como la puntuación. Una vez hecho esto, se pasará a la *tokenización* del texto, que consiste en dividir el texto en *tokens*, que son unidades pequeñas, como palabras. Los *tokens* ayudan a encontrar coincidencias y conexiones mediante la creación de partes que se pueden buscar individualmente. Además, se deberán eliminar las llamadas *stopwords*, que son términos comunes y frecuentes en un idioma que, en general, no aportan un significado relevante o diferenciador en el contexto del análisis de texto. Estas no solo no aportan información, sino que pueden llegar a mostrar un resultado incorrecto dependiendo que información y características del texto se quieran obtener.

Análisis

En este paso se aplican técnicas de procesamiento de lenguaje natural (NLP) y aprendizaje automático para la extracción de patrones, clasificación, agrupamiento y generación de nuevo conocimiento.

2.2. Aprendizaje automático

El aprendizaje automático (machine learning) [5] es una subcategoría de la inteligencia artificial, en el que se aplica un proceso por el cual los ordenadores aprovechan algoritmos y/o modelos para reconocer patrones y mejorar su capacidad para identificarlos. Con los suficientes ajustes y datos, un algoritmo de aprendizaje automático puede predecir nuevos patrones e información. Dentro de este campo existen varios tipos de aprendizaje automático [2]. En este proyecto se trabajará con algoritmos de dos tipos distintos de aprendizaje, el supervisado y el no supervisado.

2.2.1. Aprendizaje supervisado

El aprendizaje supervisado [6] es un campo del aprendizaje automático donde un modelo se entrena utilizando un conjunto de datos etiquetados. Esto significa que cada ejemplo del conjunto de entrenamiento incluye tanto las entradas como las salidas esperadas (etiquetas). El objetivo del modelo es aprender la relación entre estas entradas y sus correspondientes etiquetas para predecir con precisión las etiquetas de nuevas entradas no vistas.

Dentro de esta rama del machine learning es donde encontramos el caso principal de este trabajo, la clasificación de textos etiquetados previamente con la titulación de los distintos Grados. Sin embargo, la clasificación de textos no tiene porque ser necesariamente supervisada, pudiendo aplicarse también sobre datos no etiquetados.

2.2.2. Aprendizaje no supervisado

El aprendizaje no supervisado [7] es una rama del aprendizaje automático en la que, al contrario que sucede con el supervisado [8], el modelo se entrena con datos no etiquetados. En lugar de aprender a predecir una salida específica, el

modelo busca patrones, estructuras o relaciones ocultas en los datos. Pese a que los datos que disponemos estén etiquetados en las categorías que queremos obtener, en este caso las titulaciones de los distintos grados, en este trabajo se aplicará un método que se explicará a continuación para ver cuál es su comportamiento y poder compararlo con los algoritmos del aprendizaje supervisado.

Clustering

El clustering [9] es una técnica fundamental dentro del aprendizaje no supervisado, cuyo objetivo principal es agrupar un conjunto de datos en clusters o grupos, de manera que los elementos dentro de un mismo grupo sean lo más similares posible entre sí, y al mismo tiempo, lo más diferentes posible de los elementos en otros grupos. Esta similitud o diferencia se mide a través de métricas como la distancia euclidiana, la distancia Manhattan o la similitud coseno, dependiendo del problema y el tipo de datos.

No requiere etiquetas o categorías predefinidas; en su lugar, trabaja con datos no etiquetados y descubre patrones o estructuras inherentes en ellos. Por ejemplo, puede utilizarse para identificar segmentos de clientes en marketing, detectar anomalías en sistemas de seguridad o analizar datos genómicos en biología.

Los distintos tipos de algoritmos de clustering se pueden dividir en varias clases basado en el enfoque que se quiera dar, aunque las más conocidas serían las 3 siguientes: jerárquicos, particionados y basados en la densidad. Algunos de los algoritmos más comunes son el DBSCAN (Density-Based Spatial Clustering of Applications with Noise), K-medoids o el HCLUST.

El clustering se puede aplicar a una gran variedad de problemas, entre ellos, organización de documentos, resúmenes de textos o la clasificación de texto [10]. Respecto a esta última, si bien la agrupación es inherentemente un método de aprendizaje no supervisado, se puede aprovechar para mejorar la calidad de los resultados en su variante supervisada. En particular, utilizar agrupaciones de palabras [11] para mejorar la precisión de la clasificación de las aplicaciones supervisadas con el uso de técnicas de agrupación. Por esto mismo, para reducir la dimensionalidad de los datos con los que va a trabajar el clasificador, se hará uso de estos recursos.

2.3. Clasificación de textos

La clasificación de texto [12], como se ha comentado antes, es un tipo de aprendizaje automático en el que se aplica un proceso de asignación de categorías predeterminadas a documentos de texto u oraciones. Analiza el contenido y el significado del texto y, luego, utiliza etiquetas de texto para asignarle la más adecuada. Es una tarea fundamental en el campo del Procesamiento del Lenguaje Natural (NLP), permitiendo que las computadoras comprendan y organicen grandes cantidades de texto no estructurado. Sin embargo, para que la mayoría de los algoritmos de clasificación funcionen, se necesita realizar una preparación

tidy de los datos previa, lo que significa que los datos estén organizados de manera lógica y consistente.

La clasificación de textos tiene múltiples aplicaciones prácticas [13], como la clasificación de correos electrónicos (spam vs. no spam), el análisis de sentimientos (determinar si un texto expresa una opinión positiva, negativa o neutral) y la detección de temas (asignar textos a categorías como política, deportes o tecnología). También se utiliza en la clasificación de idiomas (identificar el idioma en que está escrito un texto), la moderación de contenido (detectar y clasificar contenido inapropiado) y otras áreas como la atención al cliente.

Estos modelos mejoran la precisión al categorizar textos de manera consistente, superando los errores que los humanos podrían cometer al clasificar datos manualmente. Además, permiten obtener análisis en tiempo real, proporcionando información accionable rápidamente, como en el análisis de comentarios de clientes para responder de inmediato a solicitudes urgentes. También facilitan la escalabilidad, permitiendo clasificar documentos de manera más eficiente en diferentes departamentos, lo que optimiza recursos y apoya el crecimiento organizacional.

Por ello, muchas organizaciones tienen grandes archivos de documentos y flujos de trabajo empresariales que generan documentos continuamente a gran escala, como documentos legales, contratos, documentos de investigación, datos generados por los usuarios y correo electrónico, utilizan la clasificación de texto como primer paso para organizar, estructurar y categorizar estos datos para posteriormente realizar análisis. Esto permite a las organizaciones ahorrar miles de horas que de otro modo serían necesarias para leer, comprender y clasificar documentos manualmente. Sin embargo, será necesaria, en general, una revisión de los resultados debido a la naturaleza del dominio, pues el lenguaje natural es ambiguo y depende del contexto [14].

2.3.1. Selección de características

Antes de cualquier tarea de clasificación, uno de los pasos más fundamentales que se deben realizar es la selección de características. Con los datos ya limpios tras el preprocesamiento, en este paso se pasa a recopilar y analizar las características del texto, convirtiendo así el texto en representaciones numéricas que los modelos de machine learning puedan comprender. Existen numerosas técnicas de *data mining* para extraer características de texto [10]. Para este trabajo se utilizará la técnica TF-IDF.

La Frecuencia de Palabras (o TF) [10] es una medida que indica cuántas veces aparece una palabra en un documento respecto al número total de palabras del documento. Su objetivo es medir la importancia relativa de una palabra dentro de un documento en particular. Es una métrica sencilla y fácil de calcular además de ser útil cuando se desea conocer la distribución de palabras dentro de un único documento. Sin embargo, no tiene en cuenta la importancia global de una palabra en el conjunto completo de documentos (el corpus), es decir, palabras comunes como “el”, “de”, “y” tendrán un valor elevado en cada documento aun-

que no aporten mucha información. El TF-IDF es una extensión de la técnica de TF, y tiene en cuenta no solo la frecuencia de aparición de una palabra dentro de un documento, sino también cuán común o rara es esa palabra en todo el corpus de documentos. El TF-IDF se calcula combinando dos componentes:

- TF: La frecuencia de la palabra en el documento.
- IDF (Inverse Document Frequency): La importancia de la palabra en el corpus completo.

Si una palabra aparece en muchos documentos, su IDF será bajo, lo que indica que no es muy informativa. Al contrario, si una palabra aparece en pocos documentos, su IDF será alto, indicando que es más rara y, por lo tanto, más significativa. Esta técnica mejora la importancia de las palabras raras y penaliza las palabras demasiado comunes que no aportan mucho valor en tareas de clasificación o análisis de texto.

Una vez calculado el valor TF-IDF de cada palabra, se creará una tabla que relacione dichos valores, las palabras y los distintos documentos, convirtiendo así los datos en estructurados, que posteriormente utilizarán los modelos de clasificación que se explican a continuación.

2.4. Modelos de clasificación

Los modelos de clasificación son una de las herramientas fundamentales en el ámbito del aprendizaje automático y la estadística. Su objetivo principal es asignar una etiqueta o categoría a cada elemento de un conjunto de datos en función de sus características. Estos modelos se utilizan ampliamente en diversas áreas, donde las más comunes serían las siguientes [10]:

- Filtrado y organización de noticias. Cada día se genera un gran cantidad de noticias, en su mayoría manera electrónica, lo que hace que sea complicado para las organizaciones poder organizarlas adecuadamente. Los métodos automáticos facilitan esta labor a todos los portales webs, evitando así tener que hacerlo de forma manual, lo que supondría un gasto en tiempo y recursos inmenso.
- Organización de documentos. Al igual que sucede con las noticias, la necesidad de organizar la cantidad de documentos hace que estos algoritmos sean de gran valor dentro de grandes repositorios como librerías digitales, literatura científica o “web collections”. Un claro ejemplo sería el Archivo Digital de la UPM.
- “Opinion Mining”. Las opiniones o reseñas de clientes suelen ser documentos de texto breves que se pueden analizar para determinar información útil a partir de la reseña. Con los modelos de clasificación, las empresas pueden dar soluciones a problemas que surjan de manera más eficiente.
- Clasificación de emails y filtrado de spam. Una tarea importante en los últimos años ha sido también clasificar los correos electrónicos [15] para

determinar el asunto o para determinar el correo basura, conocido como spam, de forma automática.

En este capítulo, se explorarán algunos de los modelos de clasificación más conocidos dentro de este ámbito, abordando tanto los aspectos teóricos como prácticos. Se presentarán las principales ventajas, limitaciones y los escenarios en los que son más eficaces.

2.4.1. Naive Bayes

El modelo de Naive Bayes [16] es un clasificador probabilístico basado en el Teorema de Bayes. Es especialmente útil en problemas de clasificación donde las características son independientes entre sí (de ahí el término “naive” o ingenuo). Aunque la independencia rara vez se cumple en la práctica, el modelo funciona sorprendentemente bien en muchas aplicaciones, como clasificación de texto y detección de spam.

Este algoritmo funciona de la siguiente manera:

1. Cálculo de Probabilidades: para cada clase posible, calcula la probabilidad de que un nuevo dato pertenezca a esa clase.
2. Aplicación del Teorema de Bayes: utiliza el teorema de Bayes para calcular la probabilidad posterior de cada clase, dado el nuevo dato.
3. Clasificación: asigna el nuevo dato a la clase con la probabilidad posterior más alta.

La fórmula del Teorema de Bayes que se utiliza para calcular las probabilidades sería:

$$P(A/B) = \frac{P(B/A) * P(A)}{P(B)}$$

Donde $P(A|B)$ representa la probabilidad de que el evento A ocurra, dado que ya sabemos que el evento B ha ocurrido. Es decir, es la probabilidad actualizada de A después de observar B.

$P(B|A)$ es la probabilidad de que el evento B ocurra, dado que sabemos que el evento A ha ocurrido, también conocido como verosimilitud.

$P(A)$ a probabilidad inicial o previa de que el evento A ocurra, antes de considerar cualquier otra evidencia y por último, $P(B)$ representa la probabilidad total de que el evento B ocurra, sin considerar ninguna relación con A.

Este modelo probabilístico es uno de los más sencillos de implementar y comprender, además de ser muy eficiente computacionalmente, especialmente en conjuntos de datos grandes. A pesar de su simplicidad, suele obtener resultados sorprendentemente buenos en muchas aplicaciones y funciona bien tanto para problemas de clasificación binaria como multiclase, por lo que encaja perfectamente en este proyecto.

2.4.2. K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) [17] es un modelo de aprendizaje supervisado que utiliza la proximidad para clasificar nuevos datos. En términos más simples, clasifica un nuevo punto de datos basándose en los puntos de datos más cercanos a él en un conjunto de datos de entrenamiento.

Para aplicar este modelo se siguen varios pasos:

1. Selección de K: el primer paso es elegir un valor para K, que representa el número de vecinos más cercanos que se considerarán.
2. Cálculo de la distancia: para cada nuevo punto de datos, se calcula la distancia a todos los puntos de datos en el conjunto de entrenamiento.
3. Selección de los K vecinos más cercanos: se identifican los K puntos de datos del conjunto de entrenamiento que están más cerca del nuevo punto de datos.
4. Votación: la clase del nuevo punto de datos se determina por votación entre los K vecinos más cercanos. La clase más frecuente entre los K vecinos se asigna al nuevo punto de datos.

La distancia que se suele calcular suele ser la euclídea, una de las distancias más básicas, pero a la vez, de las más utilizadas en el mundo del aprendizaje automático [18]. La fórmula de dicha distancia es la siguiente:

$$d(P_1/P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

El modelo KNN es fácil de implementar debido a su simplicidad, lo que lo convierte en uno de los primeros clasificadores que suele aprender un científico de datos. Además, se adapta fácilmente a nuevas muestras, ya que al almacenar todos los datos de entrenamiento en memoria, el algoritmo puede ajustarse dinámicamente a medida que se incorporan nuevos datos. Por último, su configuración es sencilla, ya que requiere pocos hiperparámetros, limitándose principalmente al valor de k y a la elección de una métrica de distancia, lo que lo hace más manejable en comparación con otros algoritmos de aprendizaje automático.

Sin embargo, este modelo presenta dos problemas. El primero, que las distancias son poco eficaces para discriminar o clasificar si hay muchas dimensiones, además de ser computacionalmente muy complejo. El segundo es que este modelo es muy sensible a la métrica y al tamaño del vecindario que se elija.

Aún así, veremos que tal se comporta y que rendimiento obtiene una vez entrenado y aplicado al conjunto de prueba.

2.4.3. Regresión Logística

La Regresión Logística [19] es un modelo lineal para problemas de clasificación binaria o multiclase. Se basa en predecir la probabilidad de que una muestra

Capítulo 2. Estado del arte

pertenezca a una clase utilizando una función logística. A diferencia de la regresión lineal, que predice un valor numérico continuo, la regresión logística predice la probabilidad de que una observación pertenezca a una de dos o más categorías.

Este modelo utiliza una función matemática llamada función logística (o sigmoide) que mapea cualquier valor real a un valor entre 0 y 1. Este valor representa la probabilidad estimada de que la observación pertenezca a una categoría específica. Después, el modelo calcula coeficientes para cada variable predictora. Estos coeficientes indican la importancia de cada variable en la predicción. Por último, al introducir nuevos datos en el modelo, se calcula la probabilidad utilizando la función logística y los coeficientes estimados. Si la probabilidad es mayor que un umbral determinado (por ejemplo, 0.5), se predice que la observación pertenece a una categoría, de lo contrario, se asigna a la otra categoría.

La función logística se define mediante la siguiente expresión matemática y que gráficamente se ve en la figura 2.1:

$$f(x) = \frac{1}{1 + e^{-x}}$$

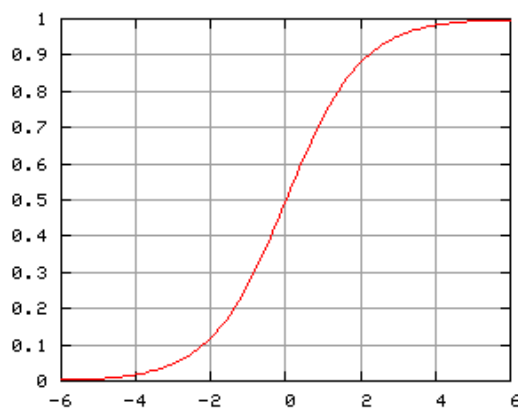


Figura 2.1: Función Logística

Existen tres tipos de modelos de regresión logística, que se definen en función de la respuesta categórica.

- Regresión logística binaria: la respuesta o variable dependiente solo tiene dos resultados posibles (p. ej., 0 o 1).
- Regresión logística multinomial: la variable dependiente tiene tres o más resultados posibles; sin embargo, estos valores no tienen un orden especificado. Este será el caso que se utilizará en este trabajo.
- Regresión logística ordinal: este tipo de modelo de regresión logística se utiliza cuando la variable de respuesta tiene tres o más resultados posibles, pero en este caso, estos valores tienen un orden definido.

Este modelo, al igual que el anterior, es fácil de interpretar, además de ser eficiente para datos linealmente separables. Al permitir ajustar pesos para cada característica y no requerir que los datos cumplan con supuestos estrictos hace que sea flexible y a la vez robusto, lo que lo convierte en una de las técnicas de aprendizaje automático más utilizadas.

2.5. Herramientas de análisis

2.5.1. R

R [20] es un sistema de análisis estadístico y gráficos. Consta de un lenguaje funcional orientado a objetos más un entorno de ejecución con gráficos, un depurador, acceso a determinadas funciones del sistema y la capacidad de ejecutar programas almacenados en archivos de script. Además, contiene una colección de más de 20.000 paquetes de muy fácil acceso que, para este trabajo, nos sirven para implementar métodos para manipular y preparar los datos, construir modelos y evaluarlos.

2.5.2. RStudio

RStudio [21] es un entorno de desarrollo integrado (IDE) para R y Python. Incluye una consola, un editor de resaltado de sintaxis que admite la ejecución directa de código y herramientas para trazar gráficos, historial, depuración y gestión del espacio de trabajo.

2.5.3. Librerías de R

Aquí se muestran los paquetes más importantes que se van a utilizar a lo largo del trabajo.

- **DBI:** La librería DBI en R es una interfaz estándar que permite conectarse y trabajar con bases de datos relacionales de manera independiente del tipo de base de datos (como MySQL, PostgreSQL, SQLite, etc.). Proporciona una serie de funciones que definen una forma común de interactuar con distintas bases de datos mediante adaptadores específicos para cada tipo. [22]
- **RSQLite:** La librería RSQLite en R permite interactuar con bases de datos SQLite, una base de datos ligera, sin servidor y almacenada en un solo archivo. RSQLite se utiliza junto con DBI para facilitar la creación, consulta y manipulación de bases de datos SQLite directamente desde R. [23]
- **dplyr:** La librería dplyr en R es una de las herramientas más populares para la manipulación y transformación de datos. Forma parte del ecosistema tidyverse y proporciona un conjunto de funciones intuitivas y eficientes para realizar operaciones comunes sobre data frames y tibbles de forma rápida, legible y escalable. [24]

- **tidytext:** La librería tidytext en R es una herramienta diseñada para realizar análisis de texto de manera fácil y estructurada utilizando principios de manipulación de datos "tidy"(ordenados). Esto significa que convierte datos de texto en estructuras tabulares (data frames o tibbles), lo que facilita su manejo y análisis usando otras librerías del ecosistema tidyverse como dplyr y ggplot2. [25]
- **ggplot2:** La librería ggplot2 en R es una de las herramientas más populares para la creación de gráficos y visualizaciones de datos. Forma parte del ecosistema tidyverse y se basa en la gramática de gráficos (Grammar of Graphics), un enfoque que descompone las visualizaciones en componentes fundamentales, como datos, capas, escalas y facetas, permitiendo construir gráficos de manera flexible y coherente. [26]

Capítulo 3

Base de datos

El objetivo principal de este apartado es presentar cómo se seleccionó la muestra utilizada para el análisis, asegurando que los datos fueran representativos y adecuados para cumplir con los objetivos del estudio para garantizar la calidad de la información antes de realizar el análisis exploratorio y las tareas de clasificación. Además, se muestra el diseño de la base de datos creada específicamente para este trabajo, la cual permite organizar y manejar la información de manera eficiente. La implementación de la base de datos se llevó a cabo utilizando las librerías RSQLite y DBI en la plataforma RStudio, herramientas que facilitaron tanto la creación de la estructura como la gestión y consulta de los datos.

También se incluye una breve descripción de las variables seleccionadas y su relevancia en el contexto del modelo de clasificación. En particular, se enfatiza la importancia de la variable objetivo, 'Titulación de Grado', como elemento central para desarrollar hipótesis y entrenar los modelos propuestos en el trabajo.

3.1. Selección de la muestra

Para escoger de manera adecuada la muestra que se va a utilizar en este trabajo, se decide hacer muestreo estadístico [27]. Este es una técnica fundamental en la estadística y consiste en seleccionar un subconjunto representativo (llamado muestra) de una población total para analizarlo y extraer conclusiones sobre la población completa sin necesidad de estudiar a todos sus integrantes. Este proceso es clave cuando estudiar toda la población es costoso, imposible o innecesario. Para ello, se ha llevado a cabo el siguiente proceso:

1. **Definir la población:** La población se ha determinado que será el conjunto de todos los Trabajos de Fin de Grado (TFG) disponibles en el Archivo Digital de la UPM durante los últimos 10 años de los 7 grados en los que más trabajos han sido publicados. Estos grados serían el Grado en Biotecnología (353), Grado en Ciencias de la Actividad Física y del Deporte (1471), Grado en Fundamentos de la Arquitectura (2463), Grado en Ingeniería del Software (734), Grado en Ingeniería en Tecnologías Industriales (1219), Grado en Ingeniería Informática (1461) y Grado en Matemáticas e

Informática (293).

2. **Elegir el tipo de muestreo:** Para seleccionar la muestra, se utilizará una técnica de muestreo probabilístico llamada muestreo estratificado, no solo por ser el más adecuado para este caso, si no porque además da mejores resultados respecto a otros métodos [28]. Esta se utiliza cuando la población tiene subgrupos (estratos) claramente definidos. Para este trabajo, queda claro que los estratos serán los distintos Grados a los que pertenece cada TFG. Una vez que se estratifica la población, el siguiente paso es seleccionar una submuestra de cada estrato utilizando el muestreo sistemático. El total de la muestra estará conformada por la suma de cada una de las submuestras de los estratos.
3. **Determinar el tamaño de la muestra:** Se quiere obtener una muestra del 3 % de la población total y que todos los grados estén representados con el mismo número de elementos en la muestra.

- Tamaño de la población total: $N = 353 + 1471 + 2463 + 734 + 1219 + 1461 + 293 = 7994$
- Tamaño total de la muestra: $n = 0,03 \cdot N = 0,03 \cdot 7994 = 239,82 \approx 240$
- Tamaño igual por grado: $n = \frac{400}{7} \approx 34$ elementos por grado

Una vez hechos estos cálculos, para la selección de los 34 trabajos de cada estrato, como se ha comentado antes, se utilizará el muestreo sistemático, donde todos los miembros de la población están listados con un número, y estos se eligen en intervalos regulares. El intervalo de muestreo k se calcula dividiendo el número total de trabajos del grado N entre el tamaño de la muestra deseada, en este caso 34: $k = \frac{N}{34}$. Se redondea k al número entero más cercano si no es exacto. Después, se elige un número aleatorio r entre 1 y k . Este será tu primer elemento seleccionado. Por último, a partir de r , se selecciona cada k -ésimo trabajo hasta completar los 34 trabajos necesarios. Los índices seleccionados serán: $r, r + k, r + 2k, r + 3k, \dots$

Una vez determinado el tamaño de la muestra, se fue extrayendo los datos de la web del Archivo Digital haciendo Web Scrapping manual. Este proceso consistiría en entrar en la web del AD-UPM y seleccionar un Grado de los que se ha elegido para el proyecto. Por ejemplo, el Grado de Biología. A continuación, calcular k para ese Grado. En el caso de Biología, sería $k = \frac{360}{34} \approx 11$. Después, elegir el primer documento que se va guardar en la base de datos y copiar el Grado, Título, Resumen, Año y Autor. Con ello, crear un *data frame* con esa información y copiarlo en la tabla "data", para guardarlo así en la base de datos.

A continuación, se muestra el código de ejemplo en R:

```
1 # Conectar una base de datos SQLite
2 con <- dbConnect(RSQLite::SQLite(), "tfg_database.sqlite")
3
4 # Crear un data frame con información de los TFGs
5 tfg_data <- data.frame(
6   Grado = "Grado en Biología",
```

3.2. Diseño de la Base de Datos

```
7   Resumen = "El polen de abedul es una de las principales causas
      de enfermedades mediadas por la inmunoglobulina E en paí
      ses del norte y centro de Europa y en Norteamérica. Los
      pacientes con alergia al polen de abedul suelen desarrollar
      alergia alimentaria, especialmente...",
8   Título = "Aislamiento y caracterización de anticuerpos con
      potencial terapéutico frente a la alergia al polen de
      abedul y a la manzana",
9   Autor = Castillo Marcos, Natalia
10 )
11
12 # Escribir el data frame en la tabla "data" en la base de datos
13 dbWriteTable(con, "data", tfg_data, overwrite = TRUE)
```

Después de esto, se iría seleccionando el resto de TFGs con un intervalo de 11 documentos entre cada uno de ellos y se volvería a repetir el mismo procedimiento.

3.2. Diseño de la Base de Datos

La base de datos se creó utilizando las librerías de RSQLite y DBI, lo que permitió crear la base de datos directamente en el entorno de R y facilitar su manipulación mediante consultas SQL, ofreciendo eficiencia a la hora de consultar grandes volúmenes de datos, además de brindar escalabilidad en caso de que se quieran realizar futuras ampliaciones del proyecto. La base de datos consta de 3 tablas.

La primera se llama “data”, véase la figura 3.1, que es el dataset que contiene la información de los TFGs recopilada del Archivo Digital. Esta tabla contiene 5 columnas, que son las variables que se han recogido de cada TFG y 240 filas, que son los TFGs de la muestra:

Grado	Resumen	Título	Año	Autor	tfg_id
1 Grado en Ingeniería Informática	Este Trabajo de Fin de Grado consiste en la ampliación y ma...	Actividades y mantenimiento para un sistema de gamificaci...	2024	Giménez Ibáñez, María	1
2 Grado en Ingeniería Informática	Este trabajo de Fin de Grado (TFG) ha consistido en la adapt...	Adaptación del sistema de ayuda a la decisión GMAA (Gene...	2023	López Palacios, Javier	2
3 Grado en Ingeniería Informática	En la actualidad es cada vez más común que ayuntamientos...	Actualización completa del vocabulario sobre alumbrado pú...	2022	Martín Blázquez, Marian	3
4 Grado en Fundamentos de la Arquitectura	Este Trabajo Fin de Grado es un estudio comparativo econó...	Análisis del coste económico estructural respecto a la NCSR...	2024	Cuesta González, Javier	4
5 Grado en Fundamentos de la Arquitectura	Somos memoria. La memoria almacena solo las experiencia...	Alzheimer y contaminación, riesgos y efectos. Casos de estu...	2023	Martínez Olivares, Carolina	5
6 Grado en Fundamentos de la Arquitectura	A lo largo de la carrera de arquitectura nos enseñan a deten...	Análisis del despilfarro: coste en construcción de edificios p...	2022	Sáenz Pérez, Beltrán	6
7 Grado en Ingeniería en Tecnologías Industriales	La gestión de los residuos radiactivos es un tema de trascen...	Análisis del ciclo nuclear: estrategias para la eliminación de r...	2024	Reina Domínguez, Helena	7
8 Grado en Ingeniería en Tecnologías Industriales	El avance tecnológico de la sociedad ha hecho que cada vez...	Análisis de las inyecciones armónicas de la carga de vehicul...	2023	Contreras Mora, Alberto	8
9 Grado en Ingeniería en Tecnologías Industriales	El objetivo principal de este trabajo se basa en realizar una s...	Análisis de medidas de corrientes armónicas en redes de dis...	2022	Ramírez Martínez, Marta	9
10 Grado en Ciencias de la Actividad Física y del Deporte	La coeducación se basa en la necesidad de educar para la ig...	Actividades de expresión corporal coeducativas para educac...	2024	Vázquez Pazos, Eduardo	10
11 Grado en Ciencias de la Actividad Física y del Deporte	El objetivo del estudio ha sido el de conocer la percepción d...	Análisis de las percepciones personales durante la práctica ...	2023	Fernández Martínez, Adrián	11
12 Grado en Ciencias de la Actividad Física y del Deporte	La educación física de base ha evolucionado durante decád...	Adherencia y motivación de la mujer en actividad física dura...	2022	Martínez López-Brea, Carlos	12
13 Grado en Ingeniería del Software	El presente trabajo se centra en un análisis cuantitativo de d...	Análisis cuantitativo de datos con temática política en Reddit	2024	Fernández Carmona, Santiago	13
14 Grado en Ingeniería del Software	El proyecto consiste en la creación de un marketplace utiliza...	Análisis y desarrollo de un Marketplace basado en tecnologi...	2023	García Frino, Manuel Antonio	14
15 Grado en Ingeniería del Software	El presente trabajo se centra en el análisis de la efectivi...	Análisis de la efectividad de las estrategias de marketing d...	2022	Frutos, Carlos	15

Showing 1 to 15 of 240 entries. 6 total columns

Figura 3.1: Tabla TFGs

Capítulo 3. Base de datos

Esta tabla contiene las siguientes variables:

- **Grado:** el Grado o la Titulación a la que pertenece el TFG. Es la variable objetivo o dependiente que se va a modelizar para la clasificación de los Trabajos de Fin de Grado.
- **Resumen:** el resumen del TFG. Se utiliza como variable clasificadora o independiente que servirá al modelo, junto a la variable Título, para clasificar los TFG's.
- **Título:** el título del TFG. Se utiliza como variable clasificadora o independiente, junto que servirá al modelo, junto a la variable Resumen, para clasificar los TFG's.
- **Año:** el año en el que se publicó el TFG.
- **Autor:** el autor que desarrolló el TFG.
- **tfg_id:** el identificador de cada documento.

La segunda tabla se llama “tf_idf”, veáse en la figura 3.2. En esta tabla se almacena la información calculada de las frecuencias de cada palabra de los resúmenes y títulos de los TFGs y está formada por 7 columnas y 28681 filas:

3.2. Diseño de la Base de Datos

	tfg_id	word	n	tf	doc_count	idf	tf_idf
1	103	climáticas	2	0.2000	2	4.3820	0.8764
2	103	modernidad	2	0.2000	2	4.3820	0.8764
3	103	utopías	2	0.2000	2	4.3820	0.8764
4	103	naturaleza	2	0.2000	18	2.5362	0.5072
5	147	precio	3	0.0938	6	3.5347	0.3316
6	147	electricidad	3	0.0938	7	3.4012	0.3190
7	103	arquitectura	2	0.2000	56	1.4376	0.2875
8	50	bot	7	0.0833	7	3.4012	0.2833
9	98	sonido	6	0.0811	7	3.4012	0.2758
10	222	spring	4	0.0727	6	3.5347	0.2570
11	50	noticias	6	0.0714	6	3.5347	0.2524
12	147	predicción	3	0.0938	16	2.6474	0.2483
13	232	hash	3	0.0612	4	3.8712	0.2369
14	22	datasets	5	0.0625	6	3.5347	0.2209

Showing 1 to 14 of 28,681 entries, 7 total columns

Figura 3.2: Tabla TF-IDF

Las variables que componen esta tabla son las siguientes:

- **tfg_id**: el identificador de cada documento.
- **word**: los términos que aparecen tanto en el título como en el resumen de cada Trabajo de Fin de Grado.
- **n**: la cantidad de veces que aparece un término en un documento específico.
- **tf**: la medida que indica cuántas veces aparece una palabra en un documento respecto al número total de palabras del documento.
- **doc_count**: la cantidad de documentos en los que aparece una palabra.
- **idf**: la importancia de una palabra dentro de un documento en relación con un conjunto de documentos.
- **tf_idf**: la combinación de TF y el IDF.

La tercera tabla tiene por nombre “tabla_documento_termino”, véase en la figura 3.3. En esta tabla contiene la información que se va pasar a los algoritmos de clustering y de clasificación. Está formada por 9384 columnas, la primera de ellas es el identificador de cada TFG, y las demás son los elementos del diccionario creado apartir de todos los términos que aparecen en el título y el resumen de los TFGs. Las filas son los 240 TFGs:

Capítulo 3. Base de datos

	alucinaciones	alude	alumbrado	alumbradopublico	alumna	alumnado	alumnas	alumno
1	0.0000	0.0000	0.0000	0.0000	0	0.0188	0.0000	0.0248
2	0.0000	0.0000	0.0000	0.0000	0	0.0000	0.0000	0.0000
3	0.0000	0.0000	0.0948	0.0292	0	0.0000	0.0000	0.0000
4	0.0000	0.0000	0.0000	0.0000	0	0.0000	0.0000	0.0195
5	0.0000	0.0000	0.0000	0.0000	0	0.0000	0.0000	0.0000
6	0.0000	0.0000	0.0000	0.0000	0	0.0000	0.0000	0.0000
7	0.0000	0.0000	0.0000	0.0000	0	0.0000	0.0000	0.0000
8	0.0000	0.0000	0.0000	0.0000	0	0.0000	0.0000	0.0000
9	0.0000	0.0000	0.0000	0.0000	0	0.0000	0.0000	0.0000
10	0.0000	0.0000	0.0000	0.0000	0	0.0276	0.0000	0.0000
11	0.0000	0.0000	0.0000	0.0000	0	0.0000	0.1726	0.0000
12	0.0000	0.0000	0.0000	0.0000	0	0.0000	0.0000	0.0000
13	0.0000	0.0000	0.0000	0.0000	0	0.0000	0.0000	0.0000
14	0.0000	0.0000	0.0000	0.0000	0	0.0000	0.0000	0.0000

Showing 1 to 14 of 240 entries. 9384 total columns

Figura 3.3: Tabla documento-término

Esta última tabla contiene las siguientes variables:

- **tfg_id**: el identificador de cada documento.
- **Términos**: Términos del diccionario.

A continuación, se muestran los primeros términos del diccionario. Los primeros que aparecen son cifras, ya que también se han tenido en cuenta como si fueran palabras.

```
1 print(diccionario, n = 20)
2 # A tibble: 9,383 1
3   value
4   <chr>
5 1 0
6 2 0,071
7 3 0,142
8 4 0.025mm
9 5 0.125
10 6 0.2
11 7 0.76
12 8 02
13 9 1
14 10 1,02
15 11 1,8
16 12 1.8
17 13 10
18 14 100
19 15 101
20 16 11
```

3.2. Diseño de la Base de Datos

```
21 17 11,4
22 18 11,8
23 19 115
24 20 12
25 #      9,363 more rows
26 #      Use 'print(n = ...)' to see more rows
```


Capítulo 4

Análisis y modelización de los datos

En este capítulo se detalla el proceso de análisis y modelización de los datos extraídos del Archivo Digital de la UPM, con el objetivo de entender mejor sus características, patrones y relaciones internas. Este análisis exploratorio es una fase fundamental para la preparación de los datos, que proporciona una base sólida para la construcción de modelos de clasificación centrados en la variable "Titulación de Grado".

A la hora de realizar tanto el preprocesamiento de los datos como la selección de características, se tendrá muy en cuenta el libro *Text Mining with R: A Tidy Approach* [25], debido a su claridad y simplicidad, mostrando ejemplos explicados detenidamente de como llevar acabo un preprocesamiento de un conjunto de documentos y extraer de estos las características más relevantes. Este estudio introduce el análisis de texto usando el enfoque *tidy* en R, que facilita la manipulación y análisis de datos textuales mediante estructuras de datos ordenadas. El texto se centra en el uso del paquete `tidytext` y herramientas complementarias como `dplyr`, `tidyr` y `ggplot2`.

4.1. Preprocesamiento

Una vez almacenados los datos en la base de datos, se debe realizar el preprocesamiento. El análisis se aplica tanto sobre el Resumen como el Título de los TFGs, por lo que primero se decide unir ambos en una misma columna, llamandose esta contenido, que contiene el Título seguido por el Resumen, separados únicamente por un espacio.

Para llevar acabo el preprocesamiento de los datos, se hizo uso de la función `unnest_tokens()` [25]. Esta pertenece al paquete `tidytext` en R y se utiliza para dividir texto en *tokens*. Es una herramienta clave en el análisis de texto porque facilita el preprocesamiento al convertir textos largos en formatos más manejables.

Capítulo 4. Análisis y modelización de los datos

`unnest_tokens()` toma un marco de datos (*data frame*) que contiene texto y lo transforma en un formato *tidy* (ordenado), donde cada *token* ocupa una fila separada, manteniendo los metadatos asociados. Además, elimina signos de puntuación automáticamente, convierte a minúsculas por defecto y mantiene las columnas originales (id en este caso) para relacionar los *tokens* con los datos originales. Esta funcionalidad hace que `unnest_tokens()` sea ideal para transformar textos en datos estructurados.

El siguiente paso es eliminar las *stopwords* de los *tokens*. Para ello, del paquete `stopwords` [29] de R, se obtiene una lista con aquellas palabras que no suelen ser relevantes en un texto y que aportan poca información. Eliminando de la lista de *tokens* aquellos que estén en la lista de *stopwords*, pasamos de tener 72871 *tokens* a 38588, donde claramente se ve que casi la mitad de los *tokens* iniciales no nos habrían sido de gran ayuda a la hora de realizar el análisis. Así, obtenemos el *data frame* de la figura 4.1:

	tfg_id	word
1	1	actividades
2	1	mantenimiento
3	1	sistema
4	1	gamificación
5	1	educación
6	1	trabajo
7	1	fin
8	1	grado
9	1	consiste
10	1	ampliación
11	1	mantenimiento
12	1	plataforma
13	1	web
14	1	draco
15	1	sistema
16	1	draco
17	1	dinámica

Showing 1 to 17 of 38,588 entries, 2 total columns

Figura 4.1: *Tokens* una vez eliminadas las *stopwords*

4.2. Frecuencia de palabras y TF-IDF

Tras haber transformado el texto a minúsculas, quitado los signos de puntuación, *tokenizado* y eliminado las *stopwords*, pasamos a obtener el índice TF-IDF para así identificar la relevancia de términos específicos dentro de los textos, permitiendo resaltar palabras clave relacionadas con cada titulación. Esta técnica la aplicaremos haciendo uso principalmente de las librerías `dplyr`, `tidytext`, `ggplot2`. Con la primera se manipularán y transformarán datos de manera eficiente y legible, realizando operaciones comunes para la manipulación de datos

4.2. Frecuencia de palabras y TF-IDF

estructurados, en este caso *data frames*. *Tidyttext* se utilizará para el análisis de texto, pudiendo así realizar tareas como la *tokenización* del texto o eliminar las *stopwords*. La última de las 3 librerías mencionadas servirá para visualizar los resultados obtenidos, creando así gráficos donde poder ver de manera clara las palabras con mayor ocurrencia dentro del texto y las aquellas que tengan un mayor valor de TF-IDF.

Para calcular el TF-IDF, seguimos los siguientes pasos:

1. Frecuencia de Término (TF): La frecuencia de término (*TF*) mide la frecuencia relativa de cada palabra en un documento. Para ello, calculamos la frecuencia de cada palabra en un documento (*n*) y la dividimos por el número total de palabras en ese documento ($\sum n_{\text{doc}}$):

$$TF = \frac{n_{\text{palabra}}}{\sum n_{\text{doc}}}$$

Donde n_{palabra} es el número de veces que una palabra aparece en el documento.

2. Frecuencia Inversa de Documentos (IDF): Esta se calcula como el logaritmo del número total de documentos dividido entre el número de documentos que contienen la palabra ($\text{doc_count}_{\text{palabra}}$):

$$IDF = \log \left(\frac{N}{1 + \text{doc_count}_{\text{palabra}}} \right)$$

Donde *N* es el número total de documentos en el corpus, en este caso 240.

3. Cálculo de TF-IDF: Finalmente, el índice TF-IDF se calcula multiplicando el *TF* de cada palabra en un documento por su *IDF* correspondiente:

$$TF - IDF = TF \times IDF$$

Este proceso nos lleva a obtener la tabla TF-IDF descrita en el diseño de la base de datos del Capítulo 3, de la cuál obtenemos los siguientes gráficos representados en las figuras 4.2 y 4.3, que nos muestran cuáles serían las palabras más y menos relevantes en el conjunto de TFGs según el índice TF-IDF:

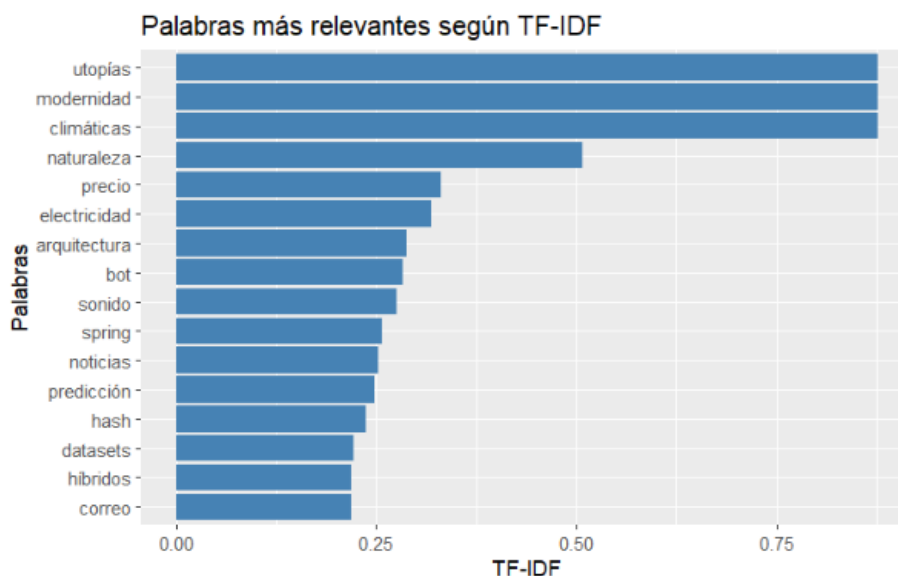


Figura 4.2: Palabras relevantes

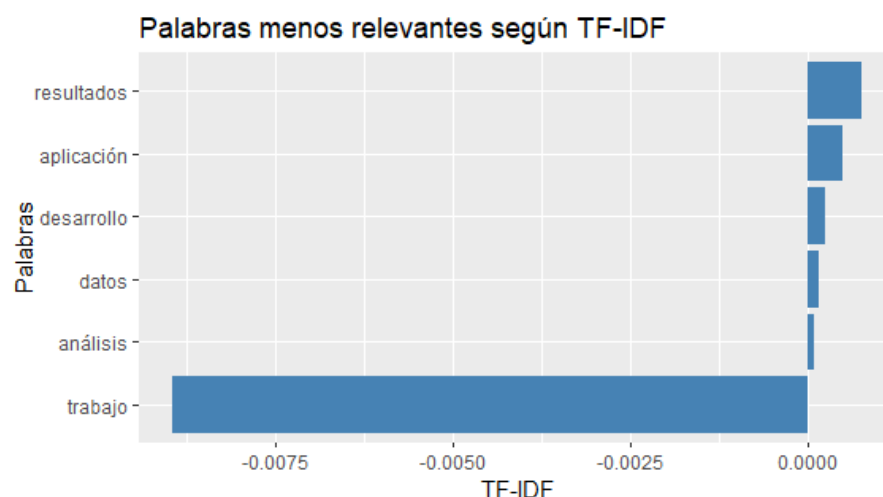


Figura 4.3: Palabras menos relevantes

En cuanto al gráfico 4.2, las palabras con los valores más altos de TF-IDF, como utopías, modernidad, climáticas y naturaleza, indican términos específicos y diferenciadores en el contexto del análisis. Estas palabras son menos frecuentes en el corpus global pero aparecen significativamente en ciertos documentos, lo que sugiere que estos términos son clave para caracterizar el contenido único de esos textos. Esto podría reflejar la relevancia de temas específicos como arquitectura sostenible, tecnología avanzada, o conceptos innovadores abordados en los trabajos analizados.

En la figura 4.3, podemos observar que las palabras con los valores más bajos de TF-IDF, como trabajo, análisis, resultados, aplicación y desarrollo, son términos

comunes en los resúmenes analizados. Estas palabras tienen una presencia uniforme en los textos, lo que indica que no ayudan a distinguir unos documentos de otros. Esto es esperable, ya que son términos genéricos que suelen aparecer de manera recurrente en textos académicos para describir metodologías y conclusiones generales.

Visto esto, se pueden comprobar las ventajas que tiene la técnica de TF-IDF como método para representar el contenido textual, ya que nos ofrece una diferenciación entre los términos más relevantes, lo que facilitará que los modelos de clasificación se concentren en términos distintivos que caracterizan cada trabajo, ignorando palabras comunes que no aportan valor discriminativo, además de ser una técnica sencilla de implementar que no requiere recursos computacionales excesivos.

4.3. Tabla documento-término

Una vez obtenido los valores del TF-IDF de cada token, se debe construir la tabla 3.3 mostrada en el Capítulo 3, ya que esta matriz convierte un problema de procesamiento de lenguaje natural en un problema de álgebra lineal, haciéndolo manejable para los algoritmos de clasificación. Esto nos permitirá predecir la titulación del grado de un TFG basándose en las palabras clave y patrones lingüísticos presentes en sus títulos y resúmenes.

Para ello, primero debemos crear el diccionario con todos los términos que aparecen tanto en el título como en el resumen de los TFGs de la base de datos. Estos términos se obtienen de la lista de *tokens* para posteriormente eliminar los duplicados que existan y ordenarlos alfabéticamente.

```
1 diccionario <- clean_tokens$word
2 diccionario <- unique(diccionario)
3 diccionario <- sort(diccionario)
```

Con el diccionario una vez creado, se crea un bucle para ir recorriendo todos los términos e ir asignando el valor del TF-IDF correspondiente o 0 si ese término no aparece en el TFG.

```
1 for (word in termino) {
2   for (i in 1:240) {
3     valor <- tf_idf[tf_idf$tf_id == i & tf_idf$word == word, "
4       tf_idf"]
5     if (length(valor$tf_idf) > 0) {
6       tabla[i, word] <- valor
7     } else {
8       tabla[i, word] <- 0
9     }
10  }
```

Capítulo 4. Análisis y modelización de los datos

Tenemos ya lista la tabla con la que van a trabajar los distintos algoritmos de clasificación.

Capítulo 5

Construcción del clasificador y análisis de rendimiento

En este capítulo se verán los pasos que se han realizado para llegar a aplicar los modelos de clasificación expuestos en el capítulo pasado además de el cálculo de distintas métricas que nos indicarán el rendimiento de cada uno.

5.1. K-Medoids

Dado que el dataset con el que vamos a trabajar tiene una dimension demasiado grande para pasarselo directamente a los modelos de clasificación (240 x 9384), se decide hacer un clustering sobre los términos del diccionario para así reducir en este caso el elementos de columnas.

Para ello se decide llevar a cabo el clustering del método K-Medoids [30]. Este es un algoritmo de agrupamiento (clustering) muy similar al conocido K-means, cuya principal diferencia es que mientras que en el algoritmo k-means un clúster se representa con su centro, en el k-medoids se representa con el objeto más cercano al centro del clúster. Su principal ventaja radica en su capacidad para manejar outliers y en el hecho de que los centros de los clusters siempre son puntos reales del conjunto de datos.

El método PAM (Partitioning Around Medoids) es uno de los enfoques más comunes para realizar el agrupamiento k-medoids, pero en este caso se utilizará una variante de PAM llamada CLARA, que sirve para analizar grandes volúmenes de datos. Para utilizar este método, se debe previamente asignar el número de clusters que se quieren obtener.

Para estimar el número óptimo de clústeres, se utiliza el método del promedio de la silueta. El objetivo es calcular el algoritmo PAM con diferentes valores de clústeres k y trazar el promedio de la silueta de los clústeres en función del número de clústeres. La silueta promedio mide la calidad del agrupamiento, y se busca el número óptimo de clústeres k que maximice la silueta promedio. La

Capítulo 5. Construcción del clasificador y análisis de rendimiento

función `fuiz_nbclust()` del paquete `factoextra` en R proporciona una solución conveniente para estimar el número óptimo de clústeres.

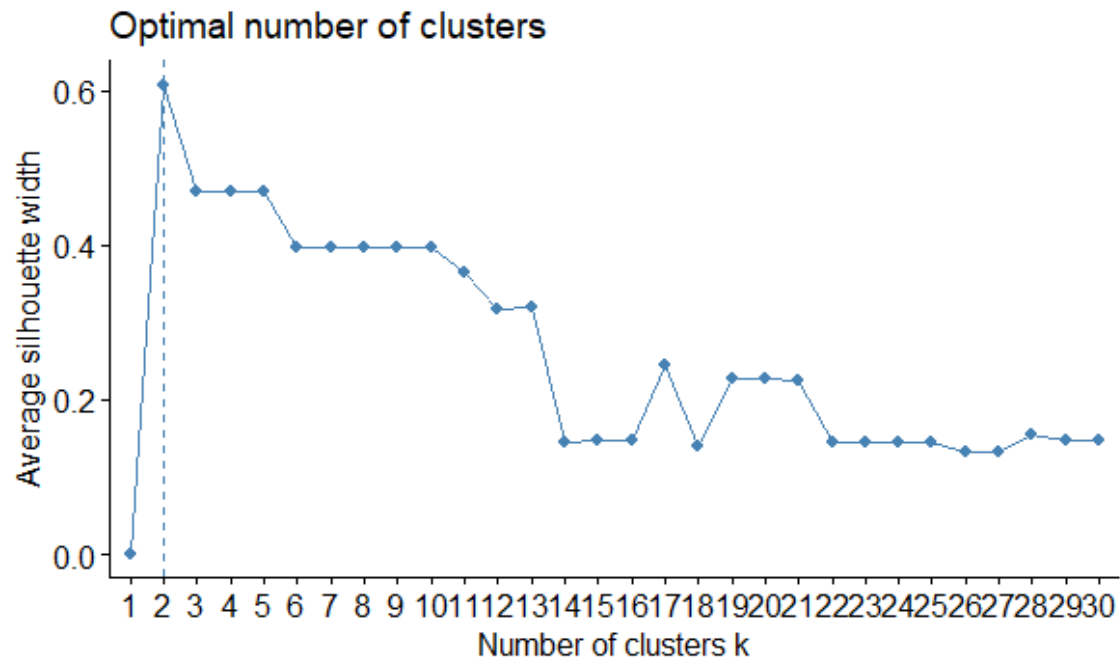


Figura 5.1: Número óptimo de clusters

Pese a que como se ve en la figura 5.1, el número de clusters óptimo para este caso sería 2, estamos buscando reducir el número de palabras a un rango de entre 20 y 30 términos, reducción suficiente para que puedan trabajar los modelos de clasificación sin perder demasiada información. Por ello, calculamos cual tiene mayor coeficiente de Silhouette entre 20 y 30:

```
1 num_clusters <- as.numeric(g$data[g$data$y==max(g$data$y[20:30])  
2 , "clusters"])  
3 num_clusters  
[1] 20
```

El número óptimo de clusters es 20, por lo que calculamos el cluster con el algoritmo CLARA:

```
1 cl.pam <- clara(tabla_transpuesta, k=num_clusters)
```

Así, obtenemos los distintos clusters, que se muestran en un dendrograma usando la función `hclust()` con los medoides calculados en el paso anterior con el algoritmo CLARA para ver cuales son las palabras que representan a cada cluster y la relación entre ellas:

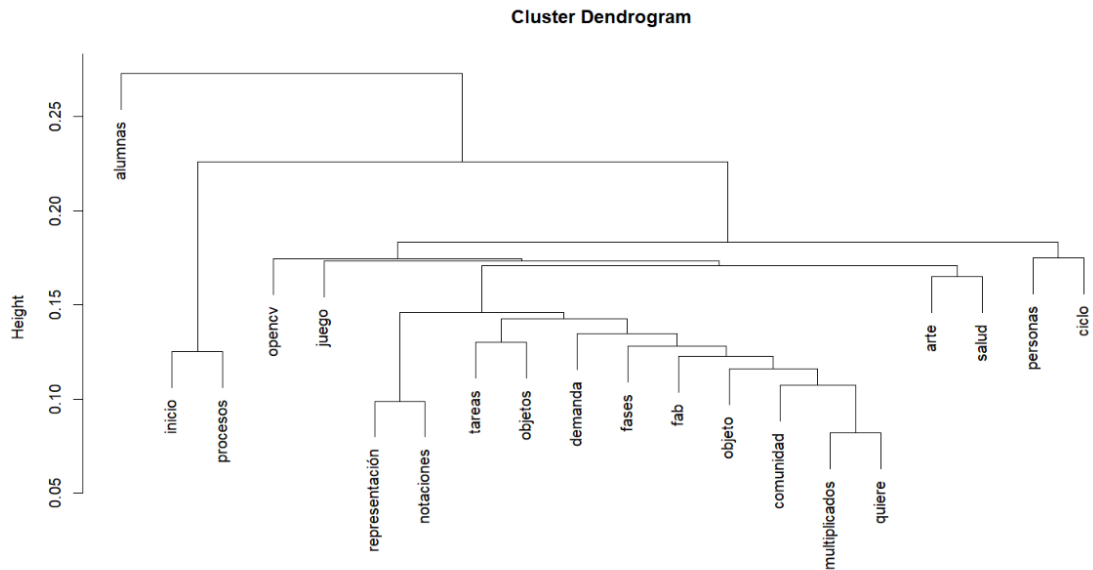


Figura 5.2: Dendrograma

5.2. Modelos de clasificación

Una vez reducida la dimensión de los datos, se procede a aplicar los modelos de clasificación seleccionados para este trabajo para posteriormente medir su rendimiento.

Primero, se dividen los datos en dos conjuntos, train y test, que servirán para entrenar al modelo y luego para comprobar que tal funciona correspondientemente. Después se aplicará el modelo, que producirá unas predicciones sobre el conjunto de prueba. Comparando estos dos, veremos el rendimiento del clasificador.

Para ello, crearemos una matriz de confusión, que es una tabla que muestra los resultados de un modelo de clasificación comparando las predicciones del modelo con las etiquetas reales. De ella obtendremos varias medidas de rendimiento [31] que nos permitirán ver que tal funciona el clasificador.

Las medidas que utilizaremos serán:

- **Accuracy:** Es la proporción de predicciones correctas sobre el total de casos.
- **Sensibilidad:** Mide qué proporción de los positivos reales se identificaron correctamente.
- **Especificidad:** Mide qué proporción de los negativos reales se identificaron correctamente.
- **Precisión:** Es la proporción de predicciones positivas que son correctas.

- **Curva ROC:** Es un gráfico que evalúa el rendimiento de un modelo en diferentes umbrales de decisión, trazando en el eje Y la sensibilidad (Tasa de Verdaderos Positivos) y en el eje X especificidad (Tasa de Falsos Positivos). Un modelo perfecto estaría representado por un punto en la esquina superior izquierda, con sensibilidad (TPR) de 1 y especificidad (1-FPR) de 1.

5.2.1. KNN

Empezaremos estudiando el modelo KNN. Obtenemos la matriz de confusión del cuadro 5.1, donde las filas serían las predicciones del modelo y las columnas serían los resultados reales. Podemos observar como la mayoría de predicciones se realizan sobre los Grados en Biotecnología, Ciencias de la Actividad Física y del Deporte y en el de Matemáticas e Informática. En las otras 4 titulaciones obtenemos muy pocas predicciones, llegando a no tener ninguna en el caso del Grado en Ingeniería en Tecnologías Industriales. Esto nos puede dar una idea ya de que el clasificador no ha funcionado todo lo bien que debería.

	Bio	Deportes	F. Arq	I. Soft	I. Ind	I. Inform	Mates
Bio	7	0	1	2	2	1	3
Deportes	3	2	4	1	4	7	3
F. Arq	0	1	1	1	0	0	0
I. Soft	0	0	0	0	1	0	0
I. Ind	0	0	0	0	0	0	0
I. Inform	0	0	0	0	0	0	1
Mates	3	6	5	3	3	3	4

Cuadro 5.1: Matriz de confusión

Se obtienen las métricas de la figura 5.3 de la matriz de confusión. Considerando una titulación frente al resto, 7 clasificadores binarios, y obteniendo la media tendríamos el general del método KNN. La media del accuracy y la especificidad son relativamente altos (0.77 y 0.87), si nos fijásemos solo en eso, podríamos pensar que el modelo está realizando una buena clasificación en promedio. Sin embargo, cuando miramos la sensibilidad, podemos ver que es muy baja (0.17), que, unido a la baja precisión (0.25), nos indica que el modelo no es bueno porque se equivoca mucho al clasificar un documento como perteneciente a las clases y que no suele identificar correctamente los casos positivos.

En cuanto a los clasificadores de cada titulación, podemos destacar que los valores más bajos de accuracy y especificidad son aquellas titulaciones que tienen los valores de sensibilidad y precisión más altos. Con ello, podemos ver que el Grado mejor clasificado sería el de Biotecnología y los peores los de Ingeniería del Software, Ingeniería en Tecnologías Industriales y el de Ingeniería Informática. El NA de la precisión de la clase 5 se debe a que los verdaderos positivos y los falsos positivos son 0.

5.2. Modelos de clasificación

Class	Accuracy	Sensitivity	Specificity	Precision
1 Grado en Biotecnología	0.7916	0.5384	0.8474	0.8474
2 Grado en Ciencias de la Actividad Física y del Deporte	0.5972	0.2222	0.6507	0.6507
3 Grado en Fundamentos de la Arquitectura	0.8333	0.0909	0.9672	0.9672
4 Grado en Ingeniería del Software	0.8889	0.0000	0.9846	0.9846
5 Grado en Ingeniería en Tecnologías Industriales	0.8611	0.0000	1.0000	1.0000
6 Grado en Ingeniería Informática	0.8333	0.0000	0.9836	0.9836
7 Grado en Matemáticas e Informática	0.5833	0.3636	0.6229	0.6229
8 GENERAL	0.7698	0.1736	0.8652	0.8652

Figura 5.3: Métricas KNN

5.2.2. Naive Bayes

En la tabla 5.2 se muestra la matriz de confusión obtenida tras aplicar el modelo Naive Bayes. Se puede destacar como el modelo tiende a predecir los documentos como pertenecientes al Grado en Biotecnología, ya que ha predicho que 45 documentos son de esa clase cuando en realidad solo hay 13. Otro dato a destacar es que el Grado en Ingeniería Informática es el único en el que no se ha clasificado como positivo ningún documento correctamente.

	Bio	Deportes	F. Arq	I. Soft	I. Ind	I. Inform	Mates
Bio	11	4	7	2	6	7	8
Deportes	0	2	2	1	0	2	1
F. Arq	0	0	1	1	0	0	0
I. Software	0	1	0	1	1	1	0
I. Indust	1	0	1	1	1	0	0
I. Inform	1	1	0	1	1	0	1
Mates	0	1	0	0	1	1	1

Cuadro 5.2: Matriz de confusión

En la figura 5.4 se muestran las métricas obtenidas de la matriz de confusión. Podemos ver que todas las métricas generales han mejorado con este modelo, pero, en los casos de la sensibilidad y la precision, siguen sin ser muy altas (0.21 y 0.24), lo que nos indica que seguimos sin obtener un buen rendimiento. En este caso, la clase peor clasificada sería la de Ingeniería Informática, pues tiene tanto la sensibilidad como la precision como valores iguales a 0. Cabe destacar la alta sensibilidad del Grado en Biotecnología (0.84), muy superior al resto. Esto se debe al haber pocos falsos negativos (2). Sin embargo, esto no quiere decir que sea buen clasificador, pues la precision es baja (0.24), dado que los falsos positivos son muy elevados (34).

Capítulo 5. Construcción del clasificador y análisis de rendimiento

Class	Accuracy	Sensitivity	Specificity	Precision
1 Grado en Biotecnología	0.5000	0.8462	0.4237	0.2444
2 Grado en Ciencias de la Actividad Física y del Deporte	0.8194	0.2222	0.9048	0.2500
3 Grado en Fundamentos de la Arquitectura	0.8472	0.0909	0.9836	0.5000
4 Grado en Ingeniería del Software	0.8750	0.1429	0.9538	0.2500
5 Grado en Ingeniería en Tecnologías Industriales	0.8333	0.1000	0.9516	0.2500
6 Grado en Ingeniería Informática	0.7778	0.0000	0.9180	0.0000
7 Grado en Matemáticas e Informática	0.8194	0.0909	0.9508	0.2500
8 GENERAL	0.7817	0.2133	0.8695	0.2492

Figura 5.4: Métricas Naive Bayes

A continuación se representan las curvas ROC en la figura 5.5. Existe una curva por clase, pues la curva ROC es una técnica de análisis de clasificación binaria. Observándolas, podemos comprobar como los rendimientos no son muy buenos, ya que se están muy cerca de la línea discontinua, que indica el rendimiento un clasificador aleatorio, es decir, uno que adivina la clase al azar, llegando a ser alguno de ellos peor, como podría ser el caso de el Grado en Ingeniería Informática, que como hemos visto en la matriz de confusión, no se había clasificado como positivo ningún documento correctamente. La titulación que mejor se clasifica sería la del Grado de Biotecnología, que pese a tener un accuracy del 50% tiene una sensibilidad muy superior a la del resto de titulaciones, haciendo así que su gráfica sea un poco mejor que las demás, sin tampoco ser muy superior.

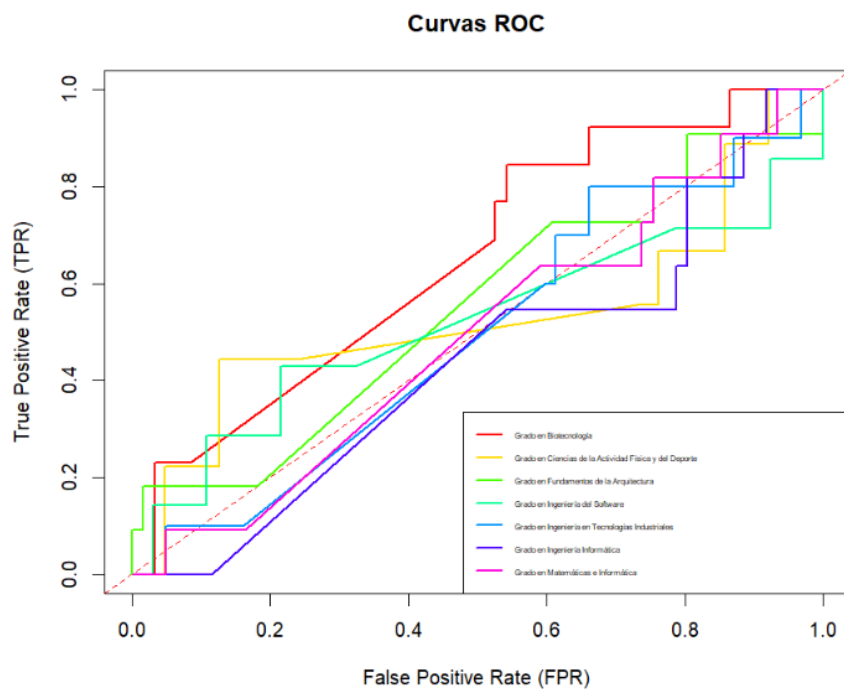


Figura 5.5: Curva ROC Naive Bayes

5.2.3. Regresión Logística

Con el modelo de Regresión Logística ocurre algo parecido que con el anterior, solo que esta vez, en vez de tender a clasificar los documentos como pertenecientes al Grado en Biotecnología, tiende a catalogarlos al Grado en Matemáticas e Informática, según podemos ver en la matriz de confusión del cuadro 5.3. Además, ocurre otra vez que el Grado en Ingeniería Informática es el único que no contiene ningún clasificado positivo de manera correcta.

	Bio	Deportes	F. Arq	I. Soft	I. Ind	I. Inform	Mates
Bio	3	0	0	0	0	1	1
Deportes	2	3	2	0	0	3	1
F. Arq	0	1	1	2	0	0	0
I. Soft	2	1	0	1	1	0	1
I. Ind	0	1	2	1	1	0	1
I. Inform	0	1	0	1	1	0	1
Mates	6	2	6	2	7	7	6

Cuadro 5.3: Matriz de confusión

En cuanto a las métricas de la figura 5.6 obtenidas de la matriz de confusión, se puede observar como los valores generales son prácticamente iguales que el modelo Naive Bayes. Si nos fijamos en cada clase, los Grados en Biotecnología y Matemáticas e Informática son los que han sufrido un cambio más significativo por lo comentado anteriormente. Con este modelo, el accuracy, la precisión y la especificidad del Grado en Biotecnología han subido, ya que los falsos positivos son mucho menores ($34 > 2$), sin embargo, la sensibilidad ha bajado, pues hay muchos falsos negativos ($2 < 10$). Con el Grado en Matemáticas e Informática pasa lo contrario, pues los falsos positivos son mucho mayores ($3 < 30$) y los falsos negativos han disminuido a la mitad ($10 > 5$).

Class	Accuracy	Sensitivity	Specificity	Precision
1 Grado en Biotecnología	0.8333	0.2308	0.9661	0.6000
2 Grado en Ciencias de la Actividad Física y del Deporte	0.8056	0.3333	0.8730	0.2727
3 Grado en Fundamentos de la Arquitectura	0.8194	0.0909	0.9508	0.2500
4 Grado en Ingeniería del Software	0.8472	0.1429	0.9231	0.1667
5 Grado en Ingeniería en Tecnologías Industriales	0.8056	0.1000	0.9194	0.1667
6 Grado en Ingeniería Informática	0.7917	0.0000	0.9344	0.0000
7 Grado en Matemáticas e Informática	0.5139	0.5455	0.5082	0.1667
8 GENERAL	0.7738	0.2062	0.8679	0.2318

Figura 5.6: Métricas Regresión Logística

Capítulo 5. Construcción del clasificador y análisis de rendimiento

Vemos que las curvas ROC de la figura 5.7 son muy parecidas a las del modelo Naive Bayes. Se podría destacar una pequeña mejora en los tramos finales de las titulaciones de Ciencias de la Actividad Física y del Deporte, Ingeniería Informática y Matemáticas e Informática. Aun así, el rendimiento sigue siendo muy pobre, pues la sensibilidad en general sigue siendo muy baja.

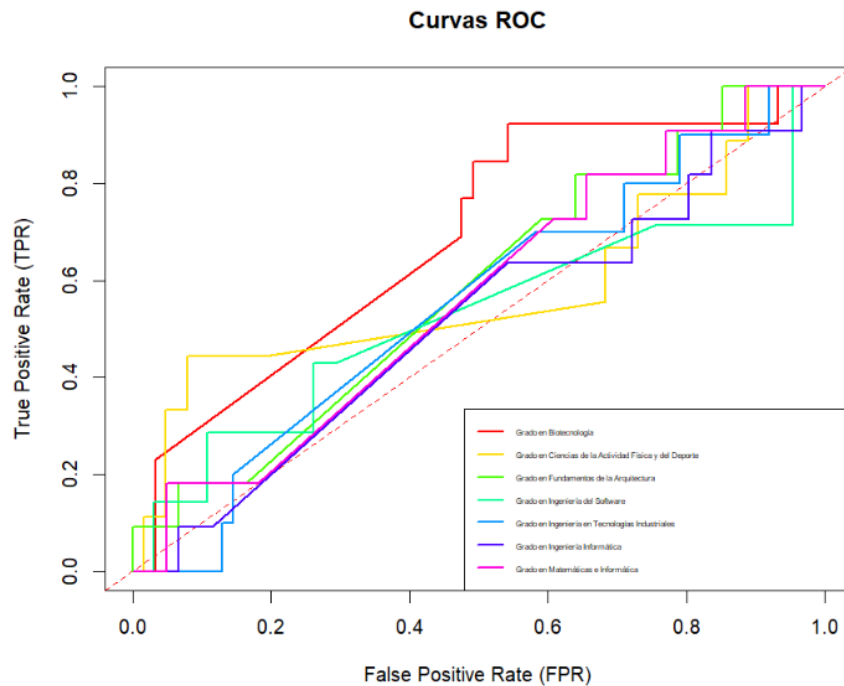


Figura 5.7: Curva ROC Regresión Logística

5.3. Clustering sobre documentos

Para intentar obtener unos resultados mejores, se decide hacer un segundo clustering, esta vez sobre los documentos, y aplicar los modelos a cada cluster. Se decide dividir los TFGs en 7 clases, ya que son el número de Titulaciones con las que estamos trabajando.

5.3.1. Clustering jerárquico

Primero se intenta con el clustering jerárquico. Sin embargo, las clases obtenidas están muy desbalanceadas, ya que la gran mayoría de los TFGs los ha agrupado en el grupo 1 mientras que en los demás no hay casi documentos, tal y como podemos ver en el cuadro 5.4:

Clase 1	Clase 2	Clase 3	Clase 4	Clase 5	Clase 6	Clase 7
231	1	1	1	3	2	1

Cuadro 5.4: Tabla de documentos por clase

5.3.2. Topic Modeling

Al ver que el clustering jerárquico no funciona como nos gustaría, se decide aplicar una técnica de clustering llamada Topic Modeling. Este es un método [25] que permite identificar temas latentes dentro de un gran conjunto de documentos. Consiste en descubrir automáticamente los “temas” o “conceptos” que se repiten a lo largo de los documentos, agrupando palabras que tienden a aparecer juntas y que representan un tema en común. Estos temas permiten organizar y resumir el contenido de los textos, facilitando su análisis, búsqueda y clasificación.

Para llevarla a cabo, se utilizará el Latent Dirichlet Allocation (LDA), un modelo generativo estadístico que permite identificar temas en un conjunto de documentos, en los que cada documento está representado como una mezcla de temas, y cada tema es una mezcla de palabras. El funcionamiento de esta técnica se basa en la suposición de que cada documento está formado por una combinación de varios temas en distintas proporciones, y que cada tema está compuesto por palabras que tienen diferentes probabilidades de ocurrencia.

LDA trata a los documentos como una “bolsa de palabras”, ignorando el orden y contexto de las palabras. En cambio, se centra en la frecuencia de aparición y co-ocurrencia de las palabras dentro de cada documento. Estos datos se representan en una matriz documento-término, donde las filas corresponden a documentos y las columnas a palabras, o viceversa. El valor en cada celda indica la frecuencia con que una palabra específica aparece en un documento dado.

A partir de esta matriz de documentos y términos, el algoritmo LDA genera distribuciones de temas (listas de palabras clave asociadas con probabilidades) en función de la frecuencia y co-ocurrencia de las palabras. Al analizar la frecuencia de coincidencias, LDA asume que las palabras que tienden a aparecer juntas

Capítulo 5. Construcción del clasificador y análisis de rendimiento

pertenecen probablemente a un mismo tema. Luego, el algoritmo asigna una distribución de temas a cada documento, basándose en los grupos de palabras que aparecen en él.

En este algoritmo, es necesario indicar previamente en cuantos temas se quiere dividir el conjunto de documentos. Al igual que con el clustering jerárquico, se deciden 7 clases, el número de titulaciones. Con esta técnica, conseguimos tener unas clases mucho más balanceadas en cuanto a número de documentos que hay en cada una, tal y como se puede observar en el cuadro 5.5:

Tema 1	Tema 2	Tema 3	Tema 4	Tema 5	Tema 6	Tema 7
30	33	24	43	33	32	45

Cuadro 5.5: Tabla de documentos por tema

Además, con este método, podemos ver los términos más comunes dentro de cada una de las 7 clases en la que ha dividido los documentos, como se puede observar en la figura 5.8. En esta, palabras como “desarrollo”, “trabajo” o “análisis” son prácticamente las que más se repiten, siendo muy complicado deducir a que Titulación se está refiriendo cada clase. Aun así, en alguna de ellas, se pueden distinguir términos que más específicos que nos ayudan un poco más. Por ejemplo, en el tema 6, se pueden observar los términos “jugadores” o “entrenamiento”, lo que nos puede dar a entender que podría ser la Titulación de Ciencias de la Actividad Física y del Deporte. También, el tema 4 podría ser la Titulación de Fundamentos de la Arquitectura, ya que solo es en este tema donde aparece la palabra “arquitectura”.

5.4. Modelos de clasificación en cada clase

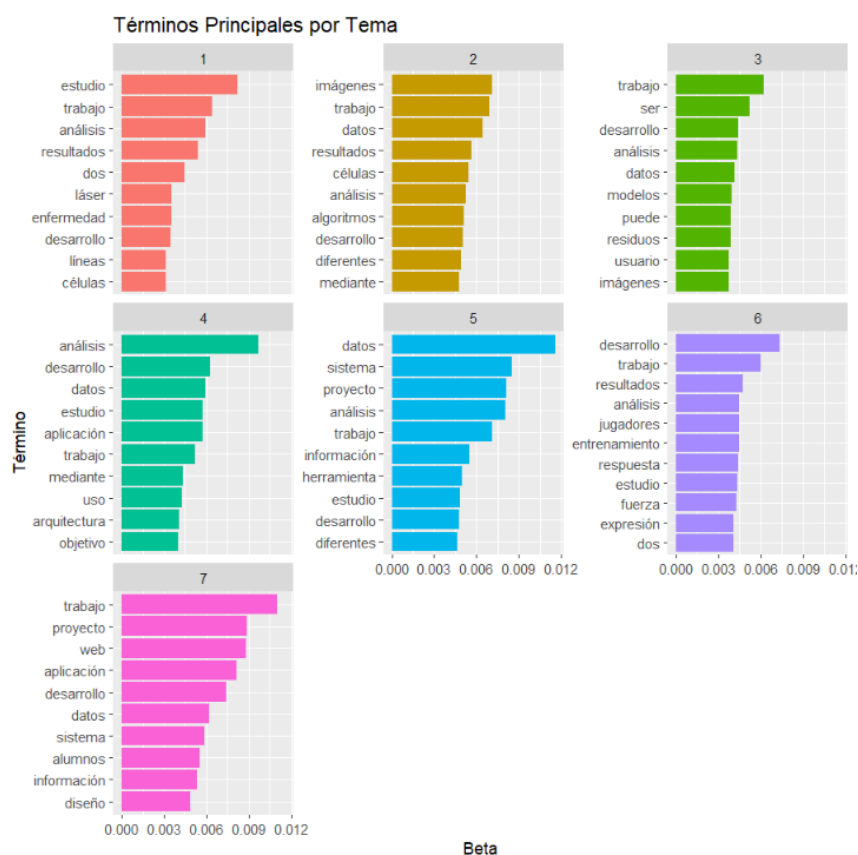


Figura 5.8: Términos principales por Tema

5.4. Modelos de clasificación en cada clase

A continuación, se muestran las matrices de confusión y las métricas obtenidas de cada cluster después de aplicar el modelo Naive Bayes:

Clase 1

Según la matriz de confusión y las métricas de esta primera clase, veáse el cuadro 5.6 y la figura 5.9, podemos ver como el único documento que había del Grado en Ciencias de la Actividad Física y del Deporte se ha evaluado de forma correcta, mientras que el Grado en Fundamentos de la Arquitectura no ha obtenido buenos resultados. El Grado en Ingeniería en Tecnologías Industriales, junto al de Matemáticas e Informática mejoran significativamente.

Capítulo 5. Construcción del clasificador y análisis de rendimiento

	Deportes	F. Arq	I. Ind	Mates
Bio	0	1	0	0
Deportes	1	0	0	0
F. Arq	0	0	1	1
I. Soft	0	0	0	0
I. Ind	0	0	1	0
I. Inform	0	0	0	0
Mates	1	0	1	2

Cuadro 5.6: Matriz de confusión Clase 1

1	Grado en Biotecnología	NA	NA	NA	NA
2	Grado en Ciencias de la Actividad Física y del Deporte	0.8889	0.5000	1.0000	1.000
3	Grado en Fundamentos de la Arquitectura	0.6667	0.0000	0.7500	0.000
4	Grado en Ingeniería del Software	NA	NA	NA	NA
5	Grado en Ingeniería en Tecnologías Industriales	0.7778	0.3333	1.0000	1.000
6	Grado en Ingeniería Informática	NA	NA	NA	NA
7	Grado en Matemáticas e Informática	0.6667	0.6667	0.6667	0.500
8	GENERAL	0.7500	0.3750	0.8542	0.625

Figura 5.9: Métricas Clase 1

Clase 2

En cuanto a la segunda clase, según podemos ver en el cuadro 5.7 y la figura 5.10, los documentos del Grado en Ciencias de la Actividad Física y del Deporte se han clasificado correctamente. Además, el Grado de Biotecnología ha obtenido resultados buenos aunque la sensibilidad no es muy alta (0.33). Sin embargo, los Grados en Ingeniería del Software, Ingeniería en Tecnologías Industriales, Ingeniería Informática y Matemáticas e Informática han obtenido resultados negativos, ya que no se ha clasificado ningún documento de manera positiva correctamente, excepto el perteneciente al Grado en Matemáticas e Informática, que si lo ha hecho pero con una precisión muy baja (0.14).

	Bio	Deportes	I. Soft	I. Ind	I. Inform	Mates
Bio	1	0	0	0	0	0
Deportes	0	2	0	0	0	0
F. Arq	0	0	0	0	0	0
I. Soft	0	0	0	0	0	0
I. Ind	0	0	0	0	0	0
I. Inform	0	0	0	0	0	0
Mates	2	0	2	1	1	1

Cuadro 5.7: Matriz de confusión Clase 2

5.4. Modelos de clasificación en cada clase

	Class	Accuracy	Sensitivity	Specificity	Precision
1	Grado en Biotecnología	0.8	0.3333	1.0000	1.0000
2	Grado en Ciencias de la Actividad Física y del Deporte	1.0	1.0000	1.0000	1.0000
3	Grado en Fundamentos de la Arquitectura	NA	NA	NA	NA
4	Grado en Ingeniería del Software	0.8	0.0000	1.0000	NA
5	Grado en Ingeniería en Tecnologías Industriales	0.9	0.0000	1.0000	NA
6	Grado en Ingeniería Informática	0.9	0.0000	1.0000	NA
7	Grado en Matemáticas e Informática	0.4	1.0000	0.3333	0.1429
8	GENERAL	0.8	0.3889	0.8889	0.7143

Figura 5.10: Métricas Clase 2

Clase 3

En la clase 3, podemos observar en la matriz de confusión, cuadro 5.8, y en las métricas obtenidas a partir de esta, figura 5.11, como los resultados cambian notoriamente respecto a la anterior clase. En esta, los documentos de los Grados en Ingeniería del Software e Ingeniería Informática se han clasificado correctamente. El Grado en Ingeniería en Tecnologías Industriales ha tenido una mejora sustancial, como se puede comprobar en su precisión (0.4). Por último, el Grado de Matemáticas e Informática, al igual que en la clase anterior, no ha obtenido buenos resultados, pues el único documento que pertenecía a esta clase no se ha clasificado correctamente.

	Bio	F. Arq	I. Soft	I. Ind	I. Inform	Mates
Deportes	0	0	0	0	0	0
I. Soft	0	0	1	0	0	0
I. Ind	1	1	0	2	0	1
I. Inform	0	0	0	0	1	0
Mates	0	0	0	0	0	0

Cuadro 5.8: Matriz de confusión Clase 3

Capítulo 5. Construcción del clasificador y análisis de rendimiento

	Class	Accuracy	Sensitivity	Specificity	Precision
1	Grado en Biotecnología	NA	NA	NA	NA
2	Grado en Ciencias de la Actividad Física y del Deporte	NA	NA	NA	NA
3	Grado en Fundamentos de la Arquitectura	NA	NA	NA	NA
4	Grado en Ingeniería del Software	1.0000	1.00	1.00	1.0
5	Grado en Ingeniería en Tecnologías Industriales	0.5714	1.00	0.40	0.4
6	Grado en Ingeniería Informática	1.0000	1.00	1.00	1.0
7	Grado en Matemáticas e Informática	0.8571	0.00	1.00	NA
8	GENERAL	0.8571	0.75	0.85	0.8

Figura 5.11: Métricas Clase 3

Clase 4

Según la matriz de confusión y las métricas de esta primera clase, véase el cuadro 5.9 y la figura 5.12, se puede observar como en esta se han obtenido muy buenos resultado los Grados en Biotecnología, Fundamentos de la Arquitectura e Ingeniería en Tecnologías Industriales. Por el contrario, los Grados en Ciencias de la Actividad Física y del Deporte e Ingeniería Informática no han sido bien clasificados.

	Bio	Deportes	F. Arq	I. Soft	I. Ind	I. Inform
Bio	1	0	0	0	0	0
Deportes	0	0	0	0	0	0
F. Arq	0	0	1	0	0	0
I. Soft	0	1	0	1	0	3
I. Ind	0	0	0	0	2	0
I. Inform	0	0	1	1	1	0
Mates	0	0	0	0	0	0

Cuadro 5.9: Matriz de confusión Clase 4

	Class	Accuracy	Sensitivity	Specificity	Precision
1	Grado en Biotecnología	1.0000	1.0000	1.0000	1.00
2	Grado en Ciencias de la Actividad Física y del Deporte	0.9167	0.0000	1.0000	NA
3	Grado en Fundamentos de la Arquitectura	0.9167	0.5000	1.0000	1.00
4	Grado en Ingeniería del Software	0.5833	0.5000	0.6000	0.20
5	Grado en Ingeniería en Tecnologías Industriales	0.9167	0.6667	1.0000	1.00
6	Grado en Ingeniería Informática	0.5000	0.0000	0.6667	0.00
7	Grado en Matemáticas e Informática	NA	NA	NA	NA
8	GENERAL	0.8056	0.4444	0.8778	0.64

Figura 5.12: Métricas Clase 4

5.4. Modelos de clasificación en cada clase

Clase 5

En esta quinta clase, la matriz de confusión y las métricas, véase el cuadro 5.10 y la figura 5.13, nos muestran como el documento del Grado en Ingeniería en Tecnologías Industriales se ha clasificado correctamente, mientras que los Grados en Ingeniería del Software, Biología y Ciencias de la Actividad Física y del Deporte y Matemáticas e Informática les ha pasado lo contrario. En cuanto a el Grado en Ingeniería Informática, su clasificación no ha sido mala, obteniendo una sensibilidad y una precisión más alta de lo normal (0.5).

	Bio	Deportes	I. Soft	I. Ind	I. Inform	Mates
Bio	0	1	0	0	0	0
Deportes	1	0	0	0	0	0
F. Arq	0	0	0	0	0	0
I. Soft	0	0	1	0	1	4
I. Ind	0	0	0	1	0	0
I. Inform	0	0	0	0	1	0
Mates	0	0	0	0	0	0

Cuadro 5.10: Matriz de confusión Clase 5

	Class	Accuracy	Sensitivity	Specificity	Precision
1	Grado en Biología	0.8000	0.0000	0.8889	0.0000
2	Grado en Ciencias de la Actividad Física y del Deporte	0.8000	0.0000	0.8889	0.0000
3	Grado en Fundamentos de la Arquitectura	NA	NA	NA	NA
4	Grado en Ingeniería del Software	0.5000	0.0000	0.5556	0.0000
5	Grado en Ingeniería en Tecnologías Industriales	1.0000	1.0000	1.0000	1.0000
6	Grado en Ingeniería Informática	0.8000	0.5000	0.8750	0.5000
7	Grado en Matemáticas e Informática	0.7000	0.2500	1.0000	1.0000
8	GENERAL	0.7667	0.2917	0.8681	0.4167

Figura 5.13: Métricas Clase 5

Capítulo 5. Construcción del clasificador y análisis de rendimiento

Clase 6

Esta clase, como podemos observar en el cuadro 5.11 y la figura 5.14, es la que está compuesta por menos Grados. De estos, el Grado en Biotecnología y el de Ciencias de la Actividad Física y del Deporte han obtenido buenos resultados, mientras que el restante, Grado en Fundamentos de la Arquitectura, no ha conseguido clasificar correctamente su documento.

	Bio	Deportes	F. Arq
Bio	2	1	0
Deportes	1	4	1
F. Arq	0	0	0
I. Ind	0	0	0

Cuadro 5.11: Matriz de confusión Clase 6

	Class	Accuracy	Sensitivity	Specificity	Precision
1	Grado en Biotecnología	0.7778	0.6667	0.8333	0.6667
2	Grado en Ciencias de la Actividad Física y del Deporte	0.6667	0.8000	0.5000	0.6667
3	Grado en Fundamentos de la Arquitectura	0.8889	0.0000	1.0000	NA
4	Grado en Ingeniería del Software	NA	NA	NA	NA
5	Grado en Ingeniería en Tecnologías Industriales	NA	NA	NA	NA
6	Grado en Ingeniería Informática	NA	NA	NA	NA
7	Grado en Matemáticas e Informática	NA	NA	NA	NA
8	GENERAL	0.7778	0.4889	0.7778	0.6667

Figura 5.14: Métricas Clase 6

5.4. Modelos de clasificación en cada clase

Clase 7

En esta última clase, la matriz de confusión y las métricas, véase el cuadro 5.12 y la figura 5.15, nos muestran algo similar a lo que ha ocurrido en otras clases, donde los documentos pertenecientes a los Grados en Ingeniería del Software e Ingeniería Informática no se han clasificado correctamente. Sin embargo, los documentos de los Grados en Fundamentos de la Arquitectura, Ingeniería en Tecnologías Industriales y Matemáticas e Informática han obtenido buenos resultados.

	Deportes	F. Arq	I. Soft	I. Ind	I. Inform	Mates
Bio	0	0	0	0	0	0
F. Arq	0	2	0	0	0	0
I. Soft	1	0	0	0	2	1
I. Ind	0	0	0	2	0	0
I. Inform	0	0	1	0	0	1
Mates	0	0	1	0	0	3

Cuadro 5.12: Matriz de confusión Clase 7

	Class	Accuracy	Sensitivity	Specificity	Precision
1	Grado en Biotecnología	NA	NA	NA	NA
2	Grado en Ciencias de la Actividad Física y del Deporte	NA	NA	NA	NA
3	Grado en Fundamentos de la Arquitectura	1.0000	1.00	1.0000	1.00
4	Grado en Ingeniería del Software	0.5714	0.00	0.6667	0.00
5	Grado en Ingeniería en Tecnologías Industriales	1.0000	1.00	1.0000	1.00
6	Grado en Ingeniería Informática	0.7143	0.00	0.8333	0.00
7	Grado en Matemáticas e Informática	0.7857	0.60	0.8889	0.75
8	GENERAL	0.8143	0.52	0.8778	0.55

Figura 5.15: Métricas Clase 7

De todas estas gráficas se puede decir que los Grados que peor rendimiento tienen son los de Ingeniería Informática e Ingeniería del Software, resultado que no sorprende pues son dos Grados muy similares, por lo que era esperable que el modelo no consiguiese diferenciarlos de manera correcta. En el otro lado, los que mejor se han clasificado son aquellos que más se diferencian con las demás titulaciones, los Grados en Biotecnología y Ciencias de la Actividad Física y del Deporte, este último obteniendo un rendimiento muy superior al de la primera clasificación con los medoides.

En general, se han mejorado los resultados obtenidos de la primera clasificación, sobre todo en cuanto a la sensibilidad y a la precisión, lo que indica que los positivos reales han aumentado, es decir, que aquellos documentos que se han clasificado como pertenecientes a un Grado han tenido un porcentaje mayor de acierto.

Esta mejora se ha podido deber a que al dividir los datos en clusters, se está

Capítulo 5. Construcción del clasificador y análisis de rendimiento

agrupando documentos con características similares, lo que ha permitido que el modelo trabaje con datos más homogéneos. Así, cada cluster representa un subconjunto con características únicas y el modelo entrenado en estos clusters está optimizado para esas particularidades, mejorando la precisión general. Además, los clusters pueden eliminar parte del ruido que podría confundir a un modelo entrenado sobre todos los datos, lo que se traduce en mejores métricas de rendimiento.

Capítulo 6

Conclusiones y líneas futuras

6.1. Evaluación de los objetivos

En este trabajo se ha construido un clasificador de documentos de TFGs centrado en la variable Titulación de Grado desde 0, explicando inicialmente el contexto de la clasificación de textos para finalmente aplicar modelos de clasificación sobre los datos recopilados. Se comenzó creando la base de datos y obteniendo los datos de la fuente el Archivo Digital de la Universidad Politécnica de Madrid. Tras ello se llevó a cabo la preparación y transformación de datos textuales utilizando las técnicas de minería de datos, en este caso TF-IDF. Además, se explican los modelos de clasificación de textos seleccionados para resolver el problema, que posteriormente se aplicarían, incluyendo los resultados obtenidos, análisis de errores y comparaciones de desempeño entre distintos modelos.

Con todo ello, se han cumplido todos los objetivos que se indicaron al principio del proyecto, desde el análisis exploratorio de los datos disponibles en el Archivo Digital de la UPM, hasta el desarrollo y evaluación de los distintos modelos de clasificación de textos.

En cuanto a los resultados obtenidos, podemos decir que la viabilidad de la clasificación automática de los TFGs no resulta posible, ya que los modelos sobre todo fallan a la hora de clasificar positivamente los documentos. Esto podía ser esperable, ya que hablamos de titulaciones muy similares, como Ingeniería del Software e Ingeniería Informática, lo que hace difícil para el modelo distinguir entre ellas. La naturaleza del problema está en parte relacionada con la falta de una frontera clara que separe las clases, ya que estas comparten muchos puntos en común.

Aún así, que el clasificador no haya obtenido tan buenos rendimientos puede deberse a otros factores más allá de la naturaleza de las clases. Estos factores podrían ser la técnica de minería de datos seleccionada para extraer información de los títulos y resúmenes de los TFGs, ya que el cálculo del TF-IDF, si bien es efectiva para identificar términos importantes, no captura el contexto semántico de las palabras. Otras técnicas, como Word Embeddings, que permiten a las máquinas entender las relaciones semánticas entre las palabras además de re-

ducir la dimensionalidad, podría haber llevado a mejores resultados. También, podría ser útil explorar modelos más complejos, como máquinas de soporte vectorial (SVM) o redes neuronales, que pueden manejar mejor la clasificación en espacios de alta dimensionalidad y donde las clases sean más similares.

En cuanto al rendimiento de los 3 modelos que se han utilizado en el trabajo, hemos visto como el KNN ha tenido un rendimiento menor que los otros dos. Esto ha podido ser por la elección de la métrica de distancia (euclidiana, manhattan, etc.) o de un valor adecuado para K, que puede afectar significativamente el rendimiento. Sin embargo, también ha podido ser por la naturaleza del modelo, ya que este no suele funcionar bien con conjuntos de datos grandes y de alta dimensionalidad o cuando las clases no están bien separadas.

6.2. Líneas futuras

En cuanto a líneas futuras de trabajo, y relacionado con el anterior apartado tendríamos:

- **Experimentación con diferentes algoritmos.** Probar otros algoritmos de clasificación, como Support Vector Machines (SVM) o redes neuronales más profundas, y ver el rendimiento que se obtiene con ellos.
- **Técnicas de preprocesamiento más avanzadas.** Como ya hemos mencionado, probar con otras técnicas como Word Embeddings, pues pueden ayudar a reducir la dimensionalidad de los datos y mejorar la calidad de la representación vectorial.
- **Mayor diversidad de titulaciones.** Ampliar el conjunto de titulaciones a clasificar. Esto podría permitir evaluar la robustez de los modelos en un espectro más amplio y analizar si existen patrones de lenguaje específicos para ciertas áreas de conocimiento.
- **Incorporación de variables.** Añadir el estudio de variables como año o autor para ver si estos proporcionan información contextual valiosa y mejoran la precisión de la clasificación.
- **Idioma.** Una opción a futuro podría ser la opción de poder clasificar documentos que estén en distintos idiomas.

6.3. Evaluación personal del proceso de realización del TFG

Estoy muy satisfecho con todo el trabajo realizado durante el desarrollo de este TFG, ya que he logrado alcanzar los objetivos que se marcaron al principio de este, consiguiendo así mejorar mi conocimiento en el análisis de datos, técnicas de machine learning y el uso de todas las herramientas utilizadas en RStudio. Todas estas habilidades adquiridas serán útiles en mi desarrollo profesional y académico.

6.4. Análisis de impacto

Son varias las áreas en las que este trabajo ha tenido y puede tener impacto potencial según los resultados obtenidos:

- Impacto económico: La automatización del proceso de clasificación puede disminuir costes asociados a la gestión de grandes volúmenes de información, como en la clasificación o en la recuperación de información, en organizaciones tanto del sector público, como bibliotecas o centros de investigación, como privado, por ejemplo consultoras y empresas de Análisis de Datos.
- Impacto social: Una mejor organización de los TFG permite optimizar el tiempo y los recursos de estudiantes y profesores a la hora de explorar trabajos previos, identificar áreas de interés, por lo que, además, puede evitar duplicar investigaciones.

6.4.1. Alineamiento con las ODS

La Agenda 2030 y los Objetivos de Desarrollo Sostenible (ODS) ofrecen una hoja de ruta hacia un futuro más sostenible. Al alinear nuestras actividades con estos objetivos, estamos más cerca de poner fin a la pobreza, proteger el planeta y asegurar que todas las personas gocen de paz y prosperidad para el año 2030.

Este proyecto se alinea con varios ODS de la Agenda 2030:

- ODS 4: Educación de calidad: Mejora el acceso y la organización del conocimiento académico.
- ODS 9: Industria, innovación e infraestructura: Propone soluciones tecnológicas para optimizar sistemas de información académica.

Bibliografía

- [1] I. H. Witten, *Text Mining*. Springer, 2004.
- [2] IBM. «Five machine learning types to know». (2023), dirección: <https://www.ibm.com/think/topics/machine-learning-types>.
- [3] P. Dialani. «The Future of Data Revolution will be Unstructured Data». (2020), dirección: <https://www.analyticsinsight.net/insights/the-future-of-data-revolution-will-be-unstructured-data>.
- [4] A. K. Uysal y S. Gunal. «The impact of preprocessing on text classification». (2014), dirección: <https://www.sciencedirect.com/science/article/abs/pii/S0306457313000964?via%3Dihub>.
- [5] S. L. Martin Emms, *Machine Learning for Natural Language Processing*. 2007.
- [6] G. Cloud. «What is Supervised Learning?» (2024), dirección: <https://cloud.google.com/discover/what-is-supervised-learning>.
- [7] G. Cloud. «What is unsupervised learning?» (2024), dirección: <https://cloud.google.com/discover/what-is-unsupervised-learning>.
- [8] IBM. «Supervised versus unsupervised learning: What's the difference?» (2021), dirección: <https://www.ibm.com/think/topics/supervised-vs-unsupervised-learning>.
- [9] S. T. A. Sreedhar Kumar Seetharaman, *A Generalized Study on Data Mining and Clustering Algorithm*. Lecture Notes in Computational Science y Engineering, 2018.
- [10] C. Z. Charu C. Aggarwal, *Mining Text Data*. Springer, 2012.
- [11] A. K. M. L Douglas Baker, *Distributional Clustering of Words for Text Classification*. ACM SIGIR Conference, 1998.
- [12] Elastic. «¿Qué es la clasificación de texto?» (2024), dirección: <https://www.elastic.co/es/what-is/text-classification>.
- [13] A. W. Services. «What is Text Classification?» (2024), dirección: <https://aws.amazon.com/what-is/text-classification/>.
- [14] J. H. M. Daniel Jurafsky, *Speech and Language Processing*. Prentice Hall, 2021.
- [15] W. W. Cohen, *Learning Rules that Classify Emails*. AAAI Conference, 1996.

BIBLIOGRAFÍA

- [16] F. Parra. «Estadística y Machine Learning con R». (2019), dirección: <https://bookdown.org/content/2274/metodos-de-clasificacion.html>.
- [17] IBM. «¿Qué es el algoritmo KNN?» (2025), dirección: <https://www.ibm.com/es-es/topics/knn>.
- [18] R. Díaz. «Algoritmo KNN – cómo funciona y ejemplos en Python». (2024), dirección: <https://www.themachinelearners.com/algoritmo-knn/>.
- [19] A. W. Services. «¿Qué es la regresión logística?» (2024), dirección: <https://aws.amazon.com/es/what-is/logistic-regression/>.
- [20] R. Foundation. «The R Project». (2024), dirección: <https://www.r-project.org/>.
- [21] Posit. «The most trusted IDE for open source data science». (2024), dirección: <https://posit.co/products/open-source/rstudio/>.
- [22] P. Software. «Using DBI». (2024), dirección: <https://solutions.posit.co/connections/db/r-packages/dbi>.
- [23] K. Müller. «Package 'RSQLite'». (2024), dirección: <https://cran.r-project.org/web/packages/RSQLite/RSQLite.pdf>.
- [24] H. Wickham. «Package 'dplyr'». (2023), dirección: <https://cran.r-project.org/web/packages/dplyr/dplyr.pdf>.
- [25] D. R. Julia Silge, *Text Mining with R*. Oreilly, 2017.
- [26] H. Wickham, *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016.
- [27] S. L. Lohr, *Sampling: Design and Analysis*. Duxbury Press, 1999.
- [28] A. B. Jacobo. «Introducción a la teoría del muestreo». (2021), dirección: https://sivea.uson.mx/docentes/tareas/Material_didactico_semestre_2021-2_1.pdf.
- [29] K. Benoit. «stopwords: Multilingual Stopword Lists». (2021), dirección: <https://cran.r-project.org/web/packages/stopwords/index.html>.
- [30] Y. Zhao, *R and Data Mining: Examples and Case Studies*. Elsevier, 2015.
- [31] G. L. Marina Sokolova, *Information Processing and Management*. Elsevier, 2009.

Anexos

Apéndice A

Anexo

A.1. Anexo de script en R

Aquí se mostrará el código utilizado a lo largo del trabajo.

Base de Datos

```
1  # #Se importan librerías que se utilizaran a lo largo de todo el
   # trabajo
2  library(dplyr)
3  library(DBI)
4  library(tidytext)
5  library(RSQLite)
6  library(ggplot2)
7  library(stopwords)
8  library(cluster)
9  library(factoextra)
10 library(class)
11 library(e1071)
12 library(nnet)
13 library(ROCR)
14 library(topicmodels)
15
16
17 # Crear/conectar una base de datos SQLite
18 con <- dbConnect(RSQLite::SQLite(), "tfg_database.sqlite")
19
20 # Crear un data frame con información de los TFGs
21 tfg_data <- data.frame(
22   Grado = "Grado en Biotecnología",
23   Resumen = "El polen de abedul es una de las principales causas
               de enfermedades mediadas por la inmunoglobulina E en paí
               ses del norte y centro de Europa y en Norteamérica. Los
               pacientes con alergia al polen de abedul suelen desarrollar
               alergia alimentaria, especialmente...",
```

Capítulo A. Anexo

```
24  Título = "Aislamiento y caracterización de anticuerpos con
      potencial terapéutico frente a la alergia al polen de
      abedul y a la manzana",
25  Autor = Castillo Marcos, Natalia
26  )
27
28  # Escribir el data frame en la tabla "data" en la base de datos
29  dbWriteTable(con, "data", tfg_data, overwrite = TRUE)
30
31  # Verificar que la tabla fue creada
32  dbListTables(con)
33
34  # Cerrar la conexión a la base de datos
35  dbDisconnect(con)
```

Tabla TF-IDF

```
1  # Conectar con la base de datos
2  con <- dbConnect(RSQLite::SQLite(), dbname = "tfg_database.
      sqlite")
3
4  # Guardar la tabla con los TFGs en la variable data
5  data <- dbReadTable(con, "data")
6
7  # Cerrar la conexión a la base de datos
8  dbDisconnect(con)
9
10 # Seleccionar y guardar tanto el título como el resumen y
    juntarlos
11 tfg_data <- dbGetQuery(con, "
12     SELECT Resumen, Título
13     FROM data
14 ")
15
16 tfg_data <- tfg_data %>%
17   mutate(contenido = paste(Título, Resumen, sep = " "))
18
19 tfg_data <- tfg_data %>%
20   mutate(tfg_id = row_number())
21
22
23 # Obtener los tokens del título y resumen
24 tokens <- tfg_data %>%
25   unnest_tokens(word, contenido)
26
27 stopwords_es <- stopwords("es")
28 stopwords_df <- data.frame(word = stopwords_es, stringsAsFactors
    = FALSE)
```

```

29
30 # Eliminar las stopwords
31 clean_tokens <- tokens %>%
32   anti_join(stopwords_df, by = "word")
33
34 # Calcular TF
35 tf <- clean_tokens %>%
36   count(tfg_id, word, name = "n") %>% # Contar las palabras por
37     documento (tfg_id)
38   group_by(tfg_id) %>%
39   mutate(tf = round(n / sum(n), 4)) %>% # Frecuencia relativa
40     con 4 decimales
41   ungroup()
42
43 total_docs <- n_distinct(clean_tokens$tfg_id)
44
45 # Calcular IDF
46 idf <- clean_tokens %>%
47   count(word, name = "doc_count") %>% # Contar documentos donde
48     aparece cada palabra
49   mutate(idf = round(log(total_docs / (1 + doc_count)), 4))
50
51 # Calcular TF-IDF
52 tf_idf <- tf %>%
53   inner_join(idf, by = "word") %>%
54   mutate(tf_idf = round(tf * idf, 4)) %>%
55   arrange(desc(tf_idf)) # Ordenar por relevancia
56 tf_idf %>%
57   top_n(15, tf_idf) %>% # Seleccionar las 20 palabras con mayor
58     TF-IDF
59   mutate(word = reorder(word, tf_idf)) %>%
60   ggplot(aes(x = word, y = tf_idf)) +
61   geom_col(fill = "steelblue") +
62   coord_flip() +
63   labs(title = "Palabras más relevantes según TF-IDF",
64        x = "Palabras",
65        y = "TF-IDF")

```

Tabla documento-termino

```

1 # Crear una tabla inicial con solo tfg_id
2 tabla <- data.frame(tfg_id = data$tfg_id)
3
4 # Obtener diccionario a partir de los tokens
5 diccionario <- clean_tokens$word
6
7 # Eliminar duplicados, si existen
8 diccionario <- unique(diccionario)

```

Capítulo A. Anexo

```
9
10 # Ordenar alfabéticamente
11 diccionario <- sort(diccionario)
12
13
14 # Asignar tf_idf si aparece word en el tfg
15 for (word in termino) {
16   for (i in 1:240) {
17     valor <- tf_idf[tf_idf$tfg_id == i & tf_idf$word == word, "
18       tf_idf"]
19     if (length(valor$tf_idf) > 0) {
20       tabla[i, word] <- valor
21     } else {
22       tabla[i, word] <- 0
23     }
24   }
25 }
```

Primer Clustering

```
1 # Eliminar el tfg_id
2 tabla_sin_id <- tabla[, -1]
3
4 # Transponer la tabla para que los términos sean filas
5 tabla_transpuesta <- t(tabla_sin_id)
6
7 # Calcular numero de clusters óptimo
8 g <- fviz_nbclust(tabla_transpuesta, clara, method = "silhouette
9   ", k.max = 30)
10
11 # Obtener el número óptimo de clusters entre 20 y 30
12 num_clusters <- as.numeric(g$data[g$data$y == max(g$data$y
13   [20:30]), "clusters"])
14
15 # Calcular los clusters con el algoritmo clara
16 cl.pam <- clara(tabla_transpuesta, k=num_clusters)
17
18 ## taxonomia del clustering particional previo
19 cl.h <- hclust(dist(cl.pam$medoids), method = "ward.D")
20 plot(cl.h)
```

Modelo KNN

```
1 # Obtener los medoides del método K-medoids
2 medoids <- as.data.frame(t(cl.pam$medoids))
3
```

```

4 # Incluir la columna Grado
5 medoids$Grado <- data$Grado
6
7
8 # 1. Preparar los datos
9 # La última columna (Grado) es la variable objetivo y las demás
  son las predictoras
10 predictoras <- medoids[, -ncol(medoids)]
11 objetivo <- medoids[, ncol(medoids)]
12
13 # Dividir en conjuntos de train y test
14 set.seed(222) # Asegurar reproducibilidad
15 train <- sample(nrow(medoids), ceiling(nrow(medoids) * 0.7))
16 test <- (1:nrow(medoids))[-train]
17
18 # Evaluar distintos valores de k
19 k_values <- seq(1, 25)
20 precision <- sapply(k_values, function(k) {
21   knn.pred <- knn(predictoras[train, ],
22                   predictoras[test, ],
23                   objetivo[train],
24                   k = k)
25   mean(knn.pred == objetivo[test])
26 })
27
28
29
30 # 2. Aplicar el modelo KNN
31 # Ajustar el modelo y predecir en los datos de test
32 predicciones <- knn(predictoras[train, ],
33                     predictoras[test, ],
34                     objetivo[train],
35                     k = which.max(precision))
36
37 # 3. Evaluar el modelo
38 # Crear una tabla de confusión
39 confusion_matrix <- table(Predicted = predicciones, Actual =
  objetivo[test])

```

Modelo Naive Bayes

```

1 # 1. Aplicar el modelo Naive Bayes
2 modelo_nb <- naiveBayes(Grado ~ ., data = train)
3
4 # Aplica el modelo a los datos de test
5 predicciones <- predict(modelo_nb, test)
6
7

```

Capítulo A. Anexo

```
8 # 2. Evaluar el modelo
9 # Crear una tabla de confusión
10 confusion_matrix <- table(Predicted = predicciones, Actual =
    test$Grado)
```

Modelo de Regresión Logística

```
1 # 1. Aplicar el modelo Regresión Logística
2 modelo_logistico <- multinom(Grado ~ ., data = train)
3
4 # Aplica el modelo a los datos de test
5 predicciones <- predict(modelo_logistico, test, type = "class")
6
7
8 # 2. Evaluar el modelo
9 # Crear una tabla de confusión
10 confusion_matrix <- table(Predicted = predicciones, Actual =
    test$Grado)
```

Cálculo de métricas

```
1 # Calcular métricas
2 classes <- c("Grado en Biotecnología", "Grado en Ciencias de la
    Actividad Física y del Deporte", "Grado en Fundamentos de la
    Arquitectura", "Grado en Ingeniería del Software", "Grado en
    Ingeniería en Tecnologías Industriales", "Grado en Ingeniería
    Informática", "Grado en Matemáticas e Informática")
3 metrics <- data.frame(Class = classes, Accuracy = NA,
    Sensitivity = NA, Specificity = NA, Precision = NA)
4
5 for (i in 1:length(classes)) {
6     class <- classes[i]
7
8     # Comprobar si 'class' está en las columnas de la matriz
9     if (!(class %in% colnames(confusion_matrix)) || !(class %in%
        rownames(confusion_matrix))) {
10         next # Salta a la siguiente iteración si no está en las
            columnas
11     }
12
13     # Extraer TP, FN, FP, TN
14     TP <- confusion_matrix[class, class]
15     FN <- sum(confusion_matrix[, class]) - TP
16     FP <- sum(confusion_matrix[class, ]) - TP
17     TN <- sum(confusion_matrix) - (TP + FN + FP)
18
19     # Calcular métricas
20     accuracy <- round((TP + TN) / sum(confusion_matrix), 4)
```

```

21 sensitivity <- ifelse((TP + FN) == 0, NA, round(TP / (TP + FN)
    , 4)) # Evitar divisiones por 0
22 specificity <- ifelse((TN + FP) == 0, NA, round(TN / (TN + FP)
    , 4)) # Evitar divisiones por 0
23 precision <- ifelse((TP + FP) == 0, NA, round(TP / (TP + FP),
    4)) # Evitar divisiones por 0
24
25 # Guardar métricas
26 metrics[i, "Accuracy"] <- accuracy
27 metrics[i, "Sensitivity"] <- sensitivity
28 metrics[i, "Specificity"] <- specificity
29 metrics[i, "Precision"] <- precision
30 }
31
32 # Calcular la media de cada métrica numérica ignorando valores
    NA
33 avg <- round(colMeans(metrics[, -1], na.rm = TRUE), 4)
34
35 # Incluir la fila TOTAL al dataframe
36 metrics <- rbind(metrics, c("GENERAL", avg))

```

Curva ROC

```

1 # Para el modelo Naive Bayes
2 probabilidades <- predict(modelo_nb, test, type = "raw")
3
4 # Para el modelo de Regresión Logística
5 probabilidades <- predict(modelo_logistico, test, type = "probs"
    )
6
7 # Configurar el gráfico
8 plot(0, 0, type = "n", xlim = c(0, 1), ylim = c(0, 1),
9     xlab = "False Positive Rate (FPR)", ylab = "True Positive
    Rate (TPR)",
10     main = "Curvas ROC")
11 abline(a = 0, b = 1, col = "red", lty = 2) # Línea diagonal de
    referencia
12
13 # Colores para las curvas
14 colores <- rainbow(length(classes))
15
16 # Iterar sobre cada clase para calcular y dibujar su curva ROC
17 for (i in 1:length(classes)) {
18     clase_objetivo <- classes[i]
19
20     # Crear etiquetas binarias para la clase actual
21     etiquetas_binarias <- ifelse(test$Grado == clase_objetivo, 1,
        0)

```

Capítulo A. Anexo

```
22
23 # Probabilidades de predicción para la clase actual
24 probabilidades_clase <- probabilidades[, clase_objetivo]
25
26 # Crear el objeto prediction y calcular performance
27 pred <- prediction(probabilidades_clase, etiquetas_binarias)
28 perf <- performance(pred, "tpr", "fpr")
29
30 # Dibujar la curva ROC
31 plot(perf, col = colores[i], lwd = 2, add = TRUE)
32 }
33
34 # Incluir la leyenda
35 legend("bottomright", legend = classes, col = colores, lty = 1,
      lwd = 2, cex = 0.4)
```

Clustering jerárquico sobre documentos

```
1 # Clustering jerárquico
2 cluster <- hclust(dist(tabla_sin_id, method = "euclidean"),
3   method = "ward.D2")
4
5 grupos_terminos <- cutree(cluster, k = 7)
6
7 grupos <- data.frame(Termino = rownames(tabla), Grupo = grupos_terminos)
8
9 tabla_frecuencias <- table(grupos$Grupo)
```

Topic Modeling

```
1 # Contar las ocurrencias de cada palabra dentro de cada
2   documento (tfg_id)
3 word_count <- clean_tokens %>%
4   count(tfg_id, word, sort = TRUE) %>%
5   ungroup()
6
7 # Crear una Matriz Documento-Término (DTM) , esta matriz
8   representa la frecuencia de cada palabra en cada documento
9 dtm <- word_count %>%
10   cast_dtm(tfg_id, word, n)
11
12 # Ajustar el Modelo LDA
13 # 'k = 7' especifica el número de temas
14 # 'control = list(seed = 1234)' establece la semilla aleatoria
15   para reproducibilidad
16 lda_model <- LDA(dtm, k = 7, control = list(seed = 1234))
```

```

15
16 # Extraer Distribuciones Tema-Término
17 topic_terms <- tidy(lda_model, matrix = "beta")
18
19 # Identificar los 10 Términos Principales para Cada Tema
20 top_terms <- topic_terms %>%
21   group_by(topic) %>%
22   top_n(10, beta) %>%
23   ungroup() %>%
24   arrange(topic, -beta)
25
26
27 # Visualizar los Términos Principales
28 ggplot(top_terms, aes(x = reorder(term, beta), y = beta, fill =
29   factor(topic))) +
30   geom_col(show.legend = FALSE) +
31   facet_wrap(~ topic, scales = "free_y") +
32   coord_flip() +
33   labs(title = "Términos Principales por Tema", x = "Término", y
34     = "Beta")
35
36 # Extraer las probabilidades documento-tópico
37 document_topics <- tidy(lda_model, matrix = "gamma")
38
39 # Asignar el tópico dominante
40 dominant_topics <- document_topics %>%
41   group_by(document) %>%
42   slice_max(gamma) %>%
43   ungroup()
44
45 tabla_frecuencias <- table(dominant_topics$topic)

```

Modelo Naive Bayes sobre cada cluster

```

1 # Clase que se quiere obtener
2 k <- 1
3
4 # Obtener documentos del topic k
5 topic <- unlist(dominant_topics[dominant_topics$topic == k, "
6   document"])
7
8 topic <- sort(as.numeric(topic))
9
10 # Obtener la tabla de TF-IDF de los documentos de topic
11 tabla_topic <- tabla_sin_id[(rownames(tabla_sin_id) %in% topic),

```

```
12 # Eliminar aquellas que tengan columnas enteras de 0
13 tabla_topic <- tabla_topic[, colSums(tabla_topic != 0) > 0]
14
15 # Incluir la columna Grado
16 tabla_topic$Grado <- data[data$tfq_id %in% topic, "Grado"]
17
18
19 # 1. Preparar los datos
20 # Dividir en conjuntos de train y test
21 set.seed(111) # Para reproducibilidad
22 train <- sample_frac(tabla_topic, .7)
23 test <- setdiff(tabla_topic, train)
24
25
26 # 2. Aplicar el modelo Naive Bayes
27 modelo_nb <- naiveBayes(Grado ~ ., data = train)
28 #modelo_logistico <- multinom(Grado ~ ., data = train)
29
30 # Aplica el modelo a los datos de test
31 predicciones <- predict(modelo_nb, test)
32
33
34 # 3. Evaluar el modelo
35 # Crear una tabla de confusión
36 confusion_matrix <- table(Predicted = predicciones, Actual =
    test$Grado)
```

Este documento esta firmado por



Firmante	CN=tfgm.fi.upm.es, OU=CCFI, O=ETS Ingenieros Informaticos - UPM, C=ES
Fecha/Hora	Tue Jan 14 16:33:16 CET 2025
Emisor del Certificado	EMAILADDRESS=camanager@etsiinf.upm.es, CN=CA ETS Ingenieros Informaticos, O=ETS Ingenieros Informaticos - UPM, C=ES
Numero de Serie	561
Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)