UNIVERSIDAD
DE GRANADA

Facultad de Ciencias. Escuela Técnica Superior de Ingenierías
Informática y de Telecomunicación

Grado en Ingeniería Informática y Matemáticas

TRABAJO DE FIN DE GRADO

Minería de bases de conocimiento

Presentado por:
Mónica Calzado Granados

Curso académico 2023-2024

Minería de bases de conocimiento

Mónica Calzado Granados

Mónica Calzado Granados *Minería de bases de conocimiento*.
Trabajo de fin de Grado. Curso académico 2023-2024.

Responsable de tutorización

Nombre del tutor 1
Departamento del tutor 1

Nombre del tutor 2
Departamento del tutor 2

Grado en Ingeniería Informática y Matemáticas

Facultad de Ciencias.
Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

Universidad de Granada

Declaración de originalidad

D./Dña. Mónica Calzado Granados

Declaro explícitamente que el trabajo presentado como Trabajo de Fin de Grado (TFG), correspondiente al curso académico 2023-2024, es original, entendido esto en el sentido de que no he utilizado para la elaboración del trabajo fuentes sin citarlas debidamente.

En Granada a 11 de febrero de 2024

Fdo: Mónica Calzado Granados

Dedicatoria (opcional)
Ver archivo `preliminares/dedicatoria.tex`

Índice general

Agradecimientos

Agradecimientos (opcional, ver archivo `preliminares/agradecimiento.tex`).

Summary

An english summary of the project (around 800 and 1500 words are recommended).
File: `preliminares/summary.tex`

Introducción

De acuerdo con la comisión de grado, el TFG debe incluir una introducción en la que se describan claramente los objetivos previstos inicialmente en la propuesta de TFG, indicando si han sido o no alcanzados, los antecedentes importantes para el desarrollo, los resultados obtenidos, en su caso y las principales fuentes consultadas.

Ver archivo `preliminares/introduccion.tex`

Parte I.

Primera parte

1. Primer capítulo

1.1. Introducción

Este documento es una plantilla para la elaboración de un trabajo fin de Grado siguiendo los requisitos de la comisión de Grado en Matemáticas de la Universidad de Granada que, a fecha de junio de 2023, son las siguientes:

- La memoria debe realizarse con un procesador de texto científico, preferiblemente (La)TeX.
- La portada debe contener el logo de la UGR, incluir el título del TFG, el nombre del estudiante y especificar el grado, la facultad y el curso actual.
- La contraportada contendrá además el nombre del tutor o tutores.
- La memoria debe necesariamente incluir:
  - Declaración explícita firmada en la que se asume la originalidad del trabajo, entendida en el sentido de que no ha utilizado fuentes sin citarlas debidamente. Esta declaración se puede descargar en la web del Grado.
  - un índice detallado de capítulos y secciones,
  - un resumen amplio en inglés del trabajo realizado (se recomienda entre 800 y 1500 palabras),
  - una introducción en la que se describan claramente los objetivos previstos inicialmente en la propuesta de TFG, indicando si han sido o no alcanzados, los antecedentes importantes para el desarrollo, los resultados obtenidos, en su caso y las principales fuentes consultadas,
  - una bibliografía final que incluya todas las referencias utilizadas.
- Se recomienda que la extensión de la memoria sea de unas 50 páginas, sin incluir posibles apéndices.

Para generar el pdf a partir de la plantilla basta compilar el fichero `tfg.tex`. Es conveniente leer los comentarios contenidos en dicho fichero pues ayudarán a entender mejor como funciona la plantilla.

La estructura de la plantilla es la siguiente[1]:

- Carpeta **preliminares**: contiene los siguientes archivos

  `dedicatoria.tex` Para la dedicatoria del trabajo (opcional)

  `agradecimientos.tex` Para los agradecimientos del trabajo (opcional)

  `introduccion.tex` Para la introducción (obligatorio)

[1] Los nombres de las carpetas no se han acentuado para evitar problemas en sistemas con Windows

summary.tex Para el resumen en inglés (obligatorio)

tablacontenidos.tex Genera de forma automática la tabla de contenidos, el índice de figuras y el índice de tablas. Si bien la tabla de contenidos es conveniente incluirla, el índice de figuras y tablas es opcional. Por defecto está desactivado. Para mostrar dichos índices hay que editar este fichero y quitar el comentario a `\listoffigures` o `\listoftables` según queramos uno de los índices o los dos. En este archivo también es posible habilitar la inclusión de un índice de listados de código (si estos han sido incluidos con el paquete `listings`)

El resto de archivos de dicha carpeta no es necesario editarlos pues su contenido se generará automáticamente a partir de los metadatos que agreguemos en `tfg.tex`

- Carpeta **capitulos**: contiene los archivos de los capítulos del TFG. Añadir tantos archivos como sean necesarios. Este capítulo es `capitulo01.tex`.
- Carpeta **apendices**: Para los apéndices (opcional)
- Carpeta **img**: Para incluir los ficheros de imagen que se usarán en el documento.
- Fichero `library.bib`: Para incluir las referencias bibliográficas en formato `bibtex`. Es útil la herramienta doi2bib para generar de forma automática el código bibtex de una referencia a partir de su DOI así como la base de datos bibliográfica MathSciNet. Para que una referencia aparezca en el pdf no basta con incluirla en el fichero `library.bib`, es necesario además *citarla* en el documento usando el comando `\cite`. Si queremos mostrar todos las referencias incluidas en el fichero `library.bib` podemos usar `\cite{*}` aunque esta opción no es la más adecuada. Se aconseja que los elementos de la bibliografía estén citados al menos una vez en el documento (y de esa forma aparecerán de forma automática en la lista de referencias).
- Fichero `glosario.tex`: Para incluir un glosario en el trabajo (opcional). Si no queremos incluir un glosario deberemos borrar el comando `\input{glosario.tex}` del fichero `tfg.tex` y posteriormente borrar el fichero `glosario.tex`
- Fichero `tfg.tex`: El documento maestro del TFG que hay que compilar con LaTeX para obtener el pdf. En dicho documento hay que cambiar la *información del título del* TFG *y el autor así como los tutores*.

1.2. Elementos del texto

En esta sección presentaremos diferentes ejemplos de los elementos de texto básico. Conviene consultar el contenido de `capitulos/capitulo01.tex` para ver cómo se han incluido.

1.2.1. Listas

En LaTeX tenemos disponibles los siguientes tipos de listas:
Listas enumeradas:

1. item 1
2. item 2

3. item 3

Listas no enumeradas

- item 1
- item 2
- item 3

Listas descriptivas

termino1 descripción 1

termino2 descripción 2

1.2.2. Tablas y figuras

En la Tabla 1.1 o la Figura 1.1 podemos ver...

Agrupados		
cabecera	cabecera	cabecera
elemento	elemento	elemento
elemento	elemento	elemento
elemento	elemento	elemento

Tabla 1.1.: Ejemplo de tabla

Figura 1.1.: Logotipo de la Universidad de Granada

1.3. Entornos matemáticos

La plantilla tiene definidos varios entornos matemáticos cuyo nombre es el mismo omitiendo los acentos. Así, para incluir una *proposición* usaríamos:

```
\begin{proposicion}
texto de la proposición
\end{proposicion}
```

Ver el código fuente del archivo `capitulo01.tex` para el resto de ejemplos.

Teorema 1.1. *Esto es un ejemplo de teorema.*

Proposición 1.1. *Ejemplo de proposición*

Lema 1.1. *Ejemplo de lema*

Corolario 1.1. *Ejemplo de corolario*

Definición 1.1. Ejemplo de definición

Observación 1.1. Ejemplo de observación

Adicionalmente está definido el entorno `teorema*` que permite incluir un teorema sin numeración:

Teorema (Fórmula de Gauß-Bonnet)**.** *Sea S una superficie compacta y K su curvatura de Gauß. Entonces*

$$\int_S K = 2\pi\chi(S)$$

donde $\chi(S)$ es la característica de Euler de S.

Las fórmulas matemáticas se escriben entre símbolos de dólar $ si van en línea con el texto o bien usando el entorno[2] `equation` cuando queremos que se impriman centradas en una línea propia, como el siguiente ejemplo

$$\cos^2 x + \sin^2 x = 1. \tag{1.1}$$

Gracias al paquete `mathtools`, las ecuaciones escritas dentro del entorno `equation` llevarán numeración de forma automática si son referenciadas en cualquier parte del documento (por ejemplo la identidad Pitagórica (1.1), ver el código de los dos anteriores ejemplos y la Sección 1.4 para más información sobre referencias cruzadas en el documento).

1.4. Referencias a elementos del texto

Para las referencias a los elementos del texto (secciones, capítulos, teoremas,...) se procede de la siguiente manera:

- Se *marca* el elemento (justo antes del mismo si se trata de un capítulo o sección o en el interior del *entorno* en otro caso), mediante el comando `\label{`*etiqueta*`}`, donde *etiqueta* debe ser un identificador único.
- Para crear una referencia al elemento en cualquier otra parte del texto se usa el comando `\ref{`*etiqueta*`}` (únicamente imprime la numeración asociada a dicho elemento, por ejemplo 1 o 1.1) o bien `\autoref{`*etiqueta*`}` (imprime la numeración del elemento así como un texto previo indicando su tipo, por ejemplo Capítulo 1 o Teorema 1.1)

1.5. Bibliografía e índice

Esto es un ejemplo de texto en un capítulo. Incluye varias citas tanto a libros [1], artículos de investigación [2], recursos online [3] (páginas web), tesis [4], trabajo fin de máster [5], trabajo

[2] También es posible delimitar una ecuación mediante los comandos `\[` y `\]` pero éstas nunca llevarán numeración aunque añadamos una etiqueta y las referenciemos (ver Sección 1.4).

fin de grado [6] así como artículos sin publicar (preprints) [7] (en estos últimos usar el campo `note` para añadir la información relevante).

Ver el fichero `library.bib` para las distintas plantillas. Cada nueva referencia debe añadirse en dicho fichero siguiendo el estilo del código `bibtex` según el tipo de referencia (página web, tesis, trabajo fin de grado o máster, artículo de investigación, libro,...). Alternativamente se puede usar la web https://zbib.org para generar automáticamente el código `bibtex`.

Citamos a MCJT [8]

Parte II.

Segunda parte

2. Ejemplo de capítulo

2.1. Primera sección

Este fichero `capitulo-ejemplo.tex` es una plantilla para añadir capítulos al TFG. Para ello, es necesario:

- Crear una copia de este fichero `capitulo-ejemplo.tex` en la carpeta `capitulos` con un nombre apropiado (p.e. `capitulo01.tex`).
- Añadir el comando `\input{capitulos/capitulo01}` en el fichero principal `tfg.tex` donde queremos que aparezca dicho capítulo.

3. Experimentación

Tras haber estudiado las bases matemáticas de diversos algoritmos de aprendizaje supervisado, vamos a aplicar dichas técnicas a un caso real.

3.1. Dataset

Para la fase de experimentación he decidido tomar como corpus una base de datos de Kaggle [9]. En esta se combinan mensajes de correo electrónico de tres fuentes distintas:

- **`LingSpam.csv`**: este dataset es uno de los más antiguos conocidos a nuestra disposición y está constituido por 2591 mensajes de spam y ham recuperados de Linguist List [10]. Estos mensajes se centran en ofertas de trabajo, oportunidades de investigación y discusiones sobre software.
- **`EnronSpam.csv`**: el dataset está formado por 9687 correos de varios trabajadores de la empresa Enron Corporation. El dataset original contiene aproximadamente 500000 emails y se puede encontrar en [11]
- **`SpamAssassin.csv`**: este dataset recoge 6045 mensajes donados por varios usuarios de un foro público. Al tratarse de usuarios variados, estos mensajes son de temas menos específicos que aquellos mensajes que podrían provenir de un mismo usuario. El dataset original se puede encontrar en [12]

Los datasets que vamos a utilizar son simplificaciones de los originales, proporcionados por el dataset de Kaggle. Esta decisión de coger mensajes de distintas fuentes se ha tomado para fomentar la diversidad de contenido y que el modelo esté entrenado sobre una mayor variedad de escenarios. Por ejemplo, si sólo empleáramos la base de datos de EnronSpam, estaríamos limitando el espacio de detección del modelo al del ámbito empresarial.

Los tres archivos .csv constan de los siguientes atributos:

- **`id:`** identificador de cada instancia o mensaje.
- **`Body:`** cuerpo o contenido del mensaje.
- **`Label:`** clase a la que pertenece la instancia (toma el valor 1 si es spam y 0 si no es spam).

3.2. Definición del problema?

Nos encontramos pues ante un problema de clasificación binaria, al cual vamos a aplicar varios algoritmos de aprendizaje supervisado para discernir qué técnica es capaz de identificar mejor las relaciones entre los contenidos de los mensajes.
Vamos a dividir el Dataset en un conjunto de entrenamiento para entrenar el modelo y un

conjunto de prueba para constatar la eficacia del modelo. El objetivo de esta fase va a ser entrenar varios modelos que puedan discernir con gran exactitud entre correo Spam y Ham, tanto para el conjunto de prueba como para un nuevo conjunto creado por nosotros y ajeno a las tres fuentes expuestas en 3.1.
La división del Dataset la vamos a hacer mediante la función `train_test_split` de Skicit-learn (ver Apéndice B.2). El conjunto de entrenamiento contendrá el 80 % de los datos y el conjunto de test el 20 % de los datos.

3.3. Preprocesado de los datos

Antes de entrenar el clasificador, es crucial realizar el preprocesado de los datos para garantizar la máxima eficacia del algoritmo de aprendizaje automático seleccionado. Se han empleado las siguientes técnicas de preprocesado:

- **Eliminación de valores nulos y duplicados:** Haciendo uso de los métodos `drop_duplicates()` y `dropna()` de la biblioteca `Pandas` (Apéndice B.1), podemos eliminar las filas con valores nulos (campos vacíos) y las filas duplicadas en un DataFrame.

- **Tokenización:** utilizamos la función `CountVectorizer()` de `sklearn` (Apéndice B.2) para convertir el cuerpo de cada correo electrónico en una representación numérica, donde cada palabra se convierte en una característica y se cuenta su frecuencia en cada correo electrónico.

  Realizamos este proceso porque el clasificador de Naive Bayes multinomial está pensado para aplicarlo en clasificación con características discretas, como recuentos de palabras [13]. Los datos de entrenamiento y de prueba se deben pasar al clasificador bien como una matriz densa (array-like) o una matriz dispersa (sparse matrix).

  La función de tokenización `CountVectorizer()` de `sklearn` precisamente devuelve una matriz dispersa, que ocupa menos espacio en memoria que una matriz densa o array-like. Hemos elegido esta función porque el conjunto de datos, al ser texto, es muy disperso, lo que significa que la mayoría de entradas en la matriz serán ceros.

- **Filtrado de palabras poco frecuentes:** eliminamos aquellas palabras que aparecen solo una vez en todo el corpus. Para ello vamos a usar el parámetro `min_df` en la configuración de `CountVectorizer()`. Este parámetro nos permite establecer un umbral mínimo del número de documentos en los que tiene que aparecer una palabra para poder formar parte del vocabulario. Al establecer `min_df=2`, estamos eliminando todas las palabras que aparezcan sólo una vez en todo el corpus. Con ello conseguimos reducir notablemente la matriz dispersa y quitar palabras irrelevantes, a la vez que mejoramos la eficiencia.

- **Eliminación de palabras vacías:** Las palabras vacías, tales como artículos, pronombres y preposiciones, se deben eliminar del texto antes de la tokenización. Esto se realiza con la intención de reducir el ruido en los datos y mejorar la calidad de las características tomadas. Para ello, scikit-learn proporciona una lista predeterminada de palabras vacías en inglés, denominada `ENGLISH_STOP_WORDS`, la cual podemos actualizar con palabras adicionales según sea necesario. Para especificar esta lista de palabras vacías en la configuración de `CountVectorizer()`, podemos utilizar el parámetro `stop_words=stop_words_list`.

Además, al analizar los correos electrónicos, se observó que la palabra *Subject* aparece al principio del cuerpo en todos los correos, lo cual es redundante y puede eliminarse.

3.4. Visualización y análisis (exploratorio) de los datos

-selección de características (mapa de palabras dese)

3.5. Algoritmos elegidos

-Naive Bayes Multinomial

A. Ejemplo de apéndice

Los apéndices son opcionales.

Este fichero `apendice-ejemplo.tex` es una plantilla para añadir apéndices al TFG. Para ello, es necesario:

- Crear una copia de este fichero `apendice-ejemplo.tex` en la carpeta `apendices` con un nombre apropiado (p.e. `apendice01.tex`).
- Añadir el comando `\input{apendices/apendice01}` en el fichero principal `tfg.tex` donde queremos que aparezca dicho apéndice (debe de ser después del comando `\appendix`).

B. Bibliotecas usadas

B.1. Pandas

Pandas [14] es una biblioteca de Python utilizada principalmente para manipulación y análisis de datos. Ofrece estructuras de datos flexibles y herramientas para trabajar con datos tabulares y de series temporales. Estas funciones son fundamentales para realizar tareas comunes de limpieza y manipulación de datos. Algunas de las funciones que hemos empleado son:

- `read_csv`: Una función utilizada para leer datos desde archivos CSV y cargarlos en un DataFrame de Pandas. Esto permite la manipulación y análisis de datos de manera conveniente en Python.
- `drop_duplicates`: Una función utilizada para eliminar filas duplicadas de un DataFrame. Esto puede ser útil cuando se trabaja con conjuntos de datos que pueden contener duplicados y se desea mantener solo una instancia de cada fila única.
- `dropna`: Una función utilizada para eliminar filas o columnas que contienen valores nulos (NaN) en un DataFrame. Esto es útil para limpiar los datos y prepararlos para su análisis, eliminando observaciones incompletas o no válidas.

B.2. Scikit-learn

Scikit-learn [15] es una biblioteca de aprendizaje automático de Python que ofrece una amplia gama de herramientas para tareas como clasificación, regresión, clustering y más. A continuación, enumeramos algunas de las funciones que hemos empleado en este trabajo:

- `train_test_split`: Una función utilizada para dividir los datos en conjuntos de entrenamiento y prueba.
- `CountVectorizer`: Una función utilizada para convertir una colección de documentos de texto en una matriz de recuentos de términos/palabras.
- `MultinomialNB`: Una implementación del clasificador Naive Bayes multinomial en scikit-learn, adecuado para clasificación de textos con características discretas como recuentos de palabras.
- `ENGLISH_STOP_WORDS`: Una lista predeterminada de palabras vacías en inglés, como artículos, pronombres y preposiciones, utilizada para la eliminación de palabras vacías en el procesamiento de texto.

Glosario

La inclusión de un glosario es opcional.
Archivo: `glosario.tex`

$\mathbb{R}$ Conjunto de números reales.

$\mathbb{C}$ Conjunto de números complejos.

$\mathbb{Z}$ Conjunto de números enteros.

Bibliografía

[1] Martin Aigner and Günter M. Ziegler. *Proofs from The Book*. Springer-Verlag, Berlin, fifth edition, 2014. Including illustrations by Karl H. Hofmann.

[2] Leonhard Euler. An essay on continued fractions. *Math. Systems Theory*, 18(4):295–328, 1985. Translated from the Latin by B. F. Wyman and M. F. Wyman.

[3] Wikipedia. Leonhard Euler — Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Leonhard_Euler, 2023. [Recurso online, accedido el 27 de julio de 2023].

[4] Robert Charles Rempel. *Relaxation Effects for Coupled Nuclear Spins*. PhD thesis, Stanford University, Stanford, CA, June 1956.

[5] Jian Tang. Spin structure of the nucleon in the asymptotic limit. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, September 1996.

[6] John Doe. Are we living in a simulation?, July 2003. Bacherlo's Thesis, Massachusetts Institute of Technology, Cambridge, MA.

[7] Jesús Castro-Infantes, José M. Manzano, and Francisco Torralbo. Conjugate plateau constructions in product spaces, 2022. Preprint. arXiv: 2203.13162 [math.DG].

[8] María del Consuelo Justicia de la Torre. Nuevas técnicas de minería de textos: Aplicaciones, 2017.

[9] User: nitishabharathi. Email spam dataset. https://www.kaggle.com/datasets/nitishabharathi/email-spam-dataset, 2020. Kaggle Dataset. Accessed: 2024-1-30.

[10] Ion Androutsopoulos, John Koutsias, Konstantinos Chandrinos, Georgios Paliouras, and Costantine Spyropoulos. An evaluation of naive bayesian anti-spam filtering. *CoRR*, cs.CL/0006013, 01 2000.

[11] Leslie Kaelbling and Melinda Gervasio. Enron email dataset. https://www.cs.cmu.edu/~./enron/. Accessed: 2024-1-30.

[12] Spam assassin dataset. https://spamassassin.apache.org/old/publiccorpus/. Accessed: 2024-1-30.

[13] Sklearn.Naive_bayes.MultinomialNB. https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html. Accessed: 2024-2-9.

[14] DataFrame — pandas 2.2.0 documentation. https://pandas.pydata.org/docs/reference/frame.html. Accessed: 2024-2-11.

[15] Scikit-learn. https://scikit-learn.org/stable/index.html. Accessed: 2024-2-9.